GENE EXPRESSION ANALYSIS FOR TIME-COURSE MICROARRAY DATA

GENE EXPRESSION ANALYSIS FOR TIME-COURSE MICROARRAY DATA

By

FANG LI, B.Sc.

A Project

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Master of Science

McMaster University

MASTER OF SCIENCE (2005) McMaster University
(Statistics) Hamilton, Ontario

TITLE:	Gene Expression Analysis for Time-Course Microarray Data
AUTHOR:	Fang Li , B.Sc. (Brock University, Canada)
SUPERVISOR:	Dr. Sylvia Esterby
NUMBER OF PAGES:	xi, 73

Abstract

DNA microarray technology makes it possible to analyze the expression levels of many thousands of genes simultaneously. One of the goals of microarray data analysis is to understand the multiple biological roles of genes and their interactions in complex biological processes. Genes with similar expression patterns are likely to share similar functions or biological processes. Therefore, analysis of changes in gene expression of a certain biological processes over time is of particular interest. Unsupervised clustering methods provide an efficient way of finding overall patterns and tendencies by clustering microarray gene expression data. The genes in the same cluster are regulated in a similar manner based on the assumption above. But traditional unsupervised clustering methods usually end up with clusters of genes with similar expression patterns but without interpretations describing the clusters in terms of gene functions or processes involved.

In this project, some statistical techniques are applied to analyze the data set from microarray experiments of sporulation in yeast. These techniques include LOWESS data normalization, which is intended to remove the systematic variations from the data; a partitional clustering method, K-means, is used with initial centroids obtained from hierarchical clustering method of DIANA; the "gap statistic" technique is implemented to estimate the "optimal" number of clusters in the data set; and finally multiple hypothesis testing is used to determine whether biologically related genes are statistically over-represented in the gene clusters using the web query tool FatiGO. These methods are combined with graphical representation of cluster profile shape and colour maps of up and down regulation via heat maps. Application of these methods to a yeast sporulation time-course data set [Chu *et al.* 1998] demonstrates the utility of cluster analysis to such data sets and provides an automated method for including biological information about gene function and characteristics.

Acknowledgements

I would like to give my sincere thanks to my supervisor, Dr. Sylvia Esterby, for any suggestions, advice, help, encouragement, and patience she has given to me. I deeply appreciate her excellent guidance.

It is my pleasure to have Dr. Canty and Dr. Viveros as my committee members. Thanks for all their expertise, time, patience and excellent advice on my project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The DNA microarray (also known as biochip, DNA chip, or gene chip) technology has had a significant impact on genome studies, making it possible to study the activities (changes in gene transcription) of many thousands of genes simultaneously instead of working on a gene-by-gene basis [Leung and Cavalieri 2003]. This technology is being broadly used in identification and diagnosis of complex genetic diseases, new drug discovery, etc.

DNA molecules on a chromosome within a cell usually contain all the genes of the organism. Each gene is a segment or region of the long double-stranded DNA which has specific function(s) to control the cell activities [Calladine and Drew 1997]. The central dogma of molecular biology is that DNA is transcribed into mRNA and then mRNA is translated into protein. A gene is expressed or active if its DNA has been transcribed to single-stranded mRNA. Gene expression is the level (amount/abundance of the mRNA) of transcription of the DNA of the gene [Nguyen *et al.* 2002]. Changes to the cell's internal or external environment can lead to changes in gene expression levels.

Finding gene groups with similar gene expression patterns may help in understanding a gene's functional behaviour over multiple conditions or experimental samples. For example, many human diseases (e.g. cancer) may be diagnosed by identifying certain changes in gene expression under some conditions.

The fundamental basis of DNA microarray is the measurement of the gene expression levels during biological processes. Microarray technology provides a means to compare gene expression levels in particular cell samples over a particular time or under multiple experimental conditions across different tissues or disease states, such as, diseased versus healthy tissue, or tissue untreated versus treated by a drug.

The commonly used statistical approaches for microarray data analysis include inferential methods, which identify significantly differentially expressed genes, and exploratory data analysis, such as clustering methods [Leung and Cavalieri 2003; Kaminski and Friedman 2002]. The replication of a microarray experiment is essential for applying inferential statistics. Exploratory methods, such as cluster analysis may be applied to find gene groups with similar expression profiles.

The data used in the project are taken from the work of Chu *et al.* (1998). The DNA microarray data contains nearly every yeast gene. Samples were taken at seven time points during the biological process of sporulation to assay changes in gene expression during sporulation. In the project, one of the objectives is the detection of gene groups where genes within a group have similar changing patterns of gene expression which are related to sporulation. The second objective is the comparison of genes with unknown functions to genes with known functions in the same cluster. Thus clues may be obtained to help predict the functions of the former genes. The third objective is the interpretation of co-expression, which shows the similar expression profile or behavior, in a gene group derived from cluster analysis in terms of biological knowledge (e.g. biological process, molecular function, cellular component).

The project is organized as follows. Chapter 2 describes the basic concepts of the microarray and the procedures of a microarray experiment. Chapter 3 gives the background of sporulation data and some characteristics different from other microarray data. Chapter 4 introduces the techniques used to remove the experimental noise in the data. Chapter 5 explains details about clustering methodology and some problems when applying it. Chapter 6 describes how to interpret the gene list in the clusters using standard biological terms. Chapter 7 discusses the results obtained from the analysis of gene expression data of sporulation in yeast by our methodology. Finally, some conclusions are drawn in Chapter 8.

Chapter 2

cDNA Microarray

Affymetrix and cDNA microarrays are two major microarray technologies used to measure the gene expression levels. cDNA microarray, one of the most popular microarray platforms, allows the comparison of gene expression levels in two different samples, e.g., the same cell type in a healthy and diseased state. The main difference between these two applications is that in the cDNA array the treatment and control are hybridized onto the same array using two different fluorescent dyes, whereas the Affymetrix chip uses only one fluorescent dye so two Affymetrix chips are needed to compare treatment and control [Knudsen 2004]. The advantages of cDNA array are its flexibility and lower cost whereas the advantages of Affymetrix array are its reliability, reproducibility, precise measurements and high density per array. In this project, the data set was obtained from cDNA microarray.

There are two parts to the experimental procedures which precede measurement of dye intensities. The first is the fabrication of the array on which the set of selected genes, called probes, are placed (top 2 steps on the left of Figure 2.1), and the second is the preparation of the sample of interest and a reference sample (top 2 steps in the middle of Figure 2.1). Typically, a microarray is a chemically coated glass slide, onto which single stranded DNA molecules of genes of interest are attached in tiny quantities on the surface of the glass at fixed positions, called probe. The step of polymerase chain reaction (PCR) involves making many copies of the gene for transfer to a single spot on the array (Nguyen *et al.* 2002). There may be thousands of spots on the glass slide, so the power of a microarray is that it is possible to measure the expression levels of thousands of genes simultaneously [Knudsen 2004].

The steps involved in making the cDNA of the test and reference sample, hybridization and measurement can be summarized as follows (Figure 2.1):

- mRNA is extracted from a test sample and a reference sample separately.
- The mRNA is converted to cDNA and then labelled with two fluorescent dyes where red fluorescent (Cy5) is used for test sample (treatment) and green fluorescent (Cy3) for reference sample (control).
- Both labelled cDNA samples are mixed and then are hybridized to the glass slide where the probes are spotted for genes of interest. Hybridization refers to binding two DNA strands together and here it will occur between the sample and/or reference cDNA and the probe, if there is matching.
- The slide is scanned by a laser microscope scanner and the intensities of the two dyes are measured for each spot.
- The scanned image is processed and the ratio of intensities of red and green is calculated as the ratio of expression level of each gene in treatment and control.

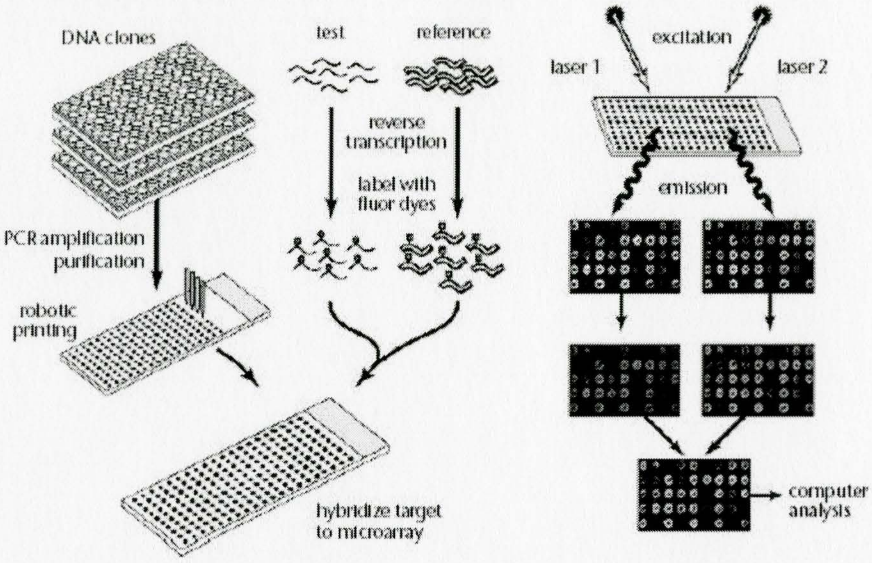


Figure 2.1: Procedures of making cDNA array [Duggan *et al.* 1999]. Used with Nature's copyright permission.

Thus, if the intensity of the red dye is high at the spot of a particular gene of interest, this implies that the gene is more active in the test sample than in the reference sample.

Chapter 3

Description of Data Set and Data Selection

3.1 Description of Data Set

The data used to study the transcriptional program of sporulation in budding yeast were collected and analyzed by Chu *et al.* (1998). The data set and additional information were obtained from http://cmgm.stanford.edu/pbrown/sporulation/additional/. There were 6118 genes in total and 41% of the genes had unknown functions. The mRNA levels (expression level) of 6118 genes in the yeast genome were measured during the course of meiosis and spore formation.

Changes in the concentrations of the mRNA transcripts from each gene were measured at seven successive intervals after transfer of wild-type (strain SK1) diploid yeast cells to a nitrogen-deficient medium that induces sporulation [Chu *et al.* 1998]. Yeast cells were transferred to sporulation-inducing medium and samples were taken after

0, 0.5, 2, 5, 7, 9 and 11.5 hours. These times correspond to different stages of meiosis and spore formation in yeast. A control sample was prepared at the initial time point (0 hour). For the sample taken at each time point, transcript abundances were determined relative to the control by hybridization of sample and control to a cDNA microarray, as described in Chapter 2 (Figure 2.1). Therefore, 7 arrays were obtained, including an array in which time 0 is compared with itself.

For each array, the data set used in the project consisted of 4 dye intensities for each gene: green at spot, green background, red at spot and red background, where red measures the sample intensity and green measures the control intensity. These four intensities for each gene were measured at each of the seven time points.

Regardless of what microarray technology is used, the data generated by microarray experiments can be viewed as a matrix of expression measurements, which is organized by N genes versus M experiments/conditions/time points/samples once the data preprocessing is complete. Initially, the data set consisted of a matrix of 6118 rows (genes) by 28 columns (dye intensities). This was reduced to a matrix of 6118 rows and 7 columns by calculating the background corrected intensity ratio at each time point, defined as:

$$\text{background corrected intensity ratio} = \frac{\text{intensity of Cy5} - \text{background intensity of Cy5}}{\text{intensity of Cy3} - \text{background intensity of Cy3}}$$

where Cy5 and Cy3 represent the red dye and the green dye respectively. The region of the slide around the spot (Figure 2.1) provides a measure of background intensity. Various methods are used to define the areas corresponding to the spot and the background and determine intensity at the spot and background [Nguyen *et al.* 2002]. It does not make any sense if the corrected intensities are zero or negative. Therefore, the genes with zero or negative ratio must be removed from the data set, but no such ratios were found in the 6118 genes.

3.2 Data Selection

In most gene expression studies there will typically only be a reasonably small percentage of the total number of genes that will show any significant change over time. For each gene, a criterion of significant change was determined by a threshold level of 1.13 of RMS (Root Mean Square) [Note 20 of Chu *et al.* 1998], where

$$\mathrm{RMS} = \sqrt{\frac{\sum_{i=1}^{n}(\log_2 x_i)^2}{n}}$$

where x_i is background corrected ratio at time i $(i = 1, 2, \ldots, 7)$. The criterion value of 1.13 is essentially equivalent to a 3-fold change of expression ratio for a single time point or an average 2.2-fold change of expression ratio across the entire time course [Chu *et al.* 1998]. Therefore, the genes (1148) whose $RMS > 1.13$ are identified and used for further analysis. The genes with $RMS \leq 1.13$ were interpreted as being genes for which variations of expression are due to measurement errors rather than biological changes, and thus should be removed. To permit comparison with results of Chu *et al.*, much of the analysis was performed on the set of 1148 genes obtained from RMS criterion, however some analyses have been done on all 6118 genes and on a subset of 1000 genes.

Chapter 4

Transformation and Normalization

4.1 Transformation

Before further analysis, the raw intensity ratios need to be transformed and normalized. The intensity ratio is also defined as a fold change. For an unchanged expression, the ratio is equal to 1; for a down-regulated gene, the ratio is less than 1 and for an up-regulated gene, the ratio is larger than 1. The problem with the raw intensity ratios is that the scale is highly asymmetric. Up-regulations will have values between 1 and infinity while down-regulations will have values between 0 and 1. Therefore, the intensity ratios should be log-transformed so that upregulated and downregulated values are of the same scale and comparable [Leung and Cavalieri 2003]. The most common function used for this purpose is the log-transformation with base 2 (i.e. $\log_2(ratio)$). It is intuitive in terms of gene fold change since it treats the up or down fold change of identical magnitude as equal but with opposite sign.

4.2 Normalization

The non-biological variation in the measured gene expression data needs to be removed or minimized so that biological differences can be more easily detected and the samples made comparable. Some possible sources of non-biological variation can be categorized as follows [Pevsner 2003]:

- Variations in the labelling efficiencies due to different physical properties of red/green fluorescence, known as dye biases.
- Variations in the sample preparation. For example, variations of mRNA purity and quantity among the biological samples being studied.
- Variations in the hybridization. For example, the reaction of hybridization is influenced by temperature, time, and the overall amount of probe molecules on the slide used for the hybridization.
- Variations in the performance of the fluorescence scanner used to detect and quantify the intensities of the fluorescent dyes.

There are a number of approaches to normalization, such as global normalization, intensity-dependent normalization, within-print tip group normalization and scale normalization [Yang *et al.* 2002]. The intensity-dependent normalization approach is used in the project.

Normalization is based on the assumption that most of the genes do not change in expression level in the samples being tested [Pevsner 2003]. Therefore the average of intensity ratio with logarithm (i.e. $base = 2$) transformation should be around zero and the mean of log ratio within each array should be zero. A side-by-side boxplot can show the systematic variation across time points before normalization (Figure 4.1). Dye bias is a common phenomenon in the raw microarray data set. The dyes Cy3 and Cy5 have different physical properties that can be checked by labelling the same sample with both dyes and then plotting intensities against each other. If the dyes of Cy3 and Cy5 were behaving similarly, then the linear regression line through the data should have a gradient of 1 and an intercept of 0. In fact, it is rare to have the dye intensities across all spots equal for two samples [Yang *et al.* 2002]. Dye bias is not linear and this can easily be seen in an MA-plot, which characterizes the difference in log intensities versus average log intensities. In a MA-plot, M is the log ratio of intensities defined as $M = \log_2(R/G)$, A is the average log ratio defined as $A = (1/2) * \log_2(R * G)$ and M is plotted versus A. To remove intensity-dependent dye bias, Yang, *et al.* (2002) recommended "LOWESS" normalization.

LOWESS (locally weighted polynomial regression) [Cleveland 1979] works by doing local regression on subsets along the length of whole data set. Then the points from the local regression are joined to form a smooth curve across the whole data set. A user-defined parameter span is used to split the whole data set into the subsets. A span of 0.1 uses 10% of the data points for each subset. The span influences the smoothness of the LOWESS curve. Larger span gives more smoothness but ineffective fit. If the span is too small, the curve will be too sensitive to local points, increasing the risk of overfitting. For each subset, the polynomial (linear or quadratic) is fitted using weighted least squares (WLS) regression. The regression weights for each data point in a subset are computed by the function below:

$$w_i = (1 - |\frac{x - x_i}{d}|^3)^3$$

where x is the predictor value associated with the response value to be fitted, x_i are the neighbours of x in the subset determined by the span, and d is the distance from x to the most distant predictor value within the subset. Therefore, more weight is given to the points near the point whose response is being estimated and less weight to points further away.

The weighted least squares regression [Hamilton 2002] employs an $n \times n$ weighting matrix W, where n is the number of the points in subset, with the weights w_i on the diagonal of the matrix and zero elsewhere. The weighted least squares regression model is defined as:

$$\mathbf{W}^{\frac{1}{2}}\mathbf{Y} = \mathbf{W}^{\frac{1}{2}}\mathbf{X}\boldsymbol{\beta} + \mathbf{W}^{\frac{1}{2}}\boldsymbol{\epsilon}$$

and the estimated coefficients $\boldsymbol{\beta}$ is

$$\hat{\boldsymbol{\beta}} = ((\mathbf{W}^{\frac{1}{2}}\mathbf{X})^{'}(\mathbf{W}^{\frac{1}{2}}\mathbf{X}))^{-1}(\mathbf{W}^{\frac{1}{2}}\mathbf{X})^{'}(\mathbf{W}^{\frac{1}{2}}\mathbf{Y})$$

The fitted value for the point is then obtained by evaluating the local polynomial using the predictor value for that data point:

$$\hat{\boldsymbol{Y}} = \mathbf{X}\hat{\boldsymbol{\beta}}$$

LOWESS works well to reduce intensity-dependent curvature in the measured expression levels.

$$\log_2 \frac{R}{G} \rightarrow \log_2 \frac{R}{G} - c(A)$$

where $c(A)$ is the LOWESS fit to the MA-plot and is computed separately for each array. In particular, it will not be affected by a small percentage of differentially expressed genes [Yang *et al.* 2002]. Comparison with the fit, the span parameter, $span = 0.88$ was selected for the LOWESS normalization. As noted above, the Figure 4.2 shows the Box-plot and MA-plot after LOWESS normalization on 1148 filtered genes over seven time points. The data set used for most of the calculations consists of 1148×7 matrix of normalized log ratios, ie. the data retained by the RMS procedure.

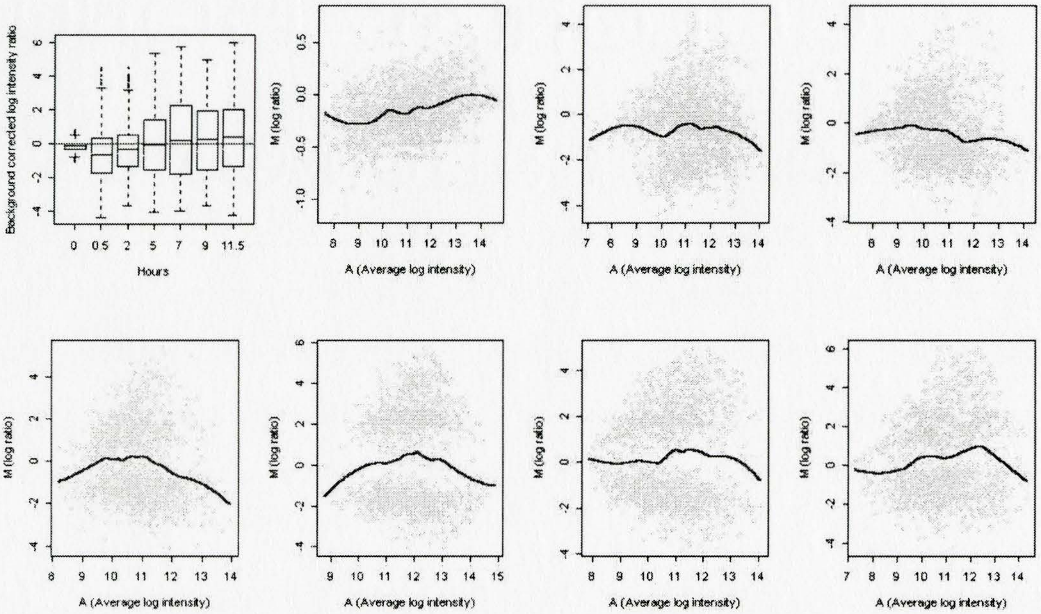


Figure 4.1: The boxplot and MA-plots of 1148 filtered genes over seven time points in order from 0 to 11.5 hours (before normalization).

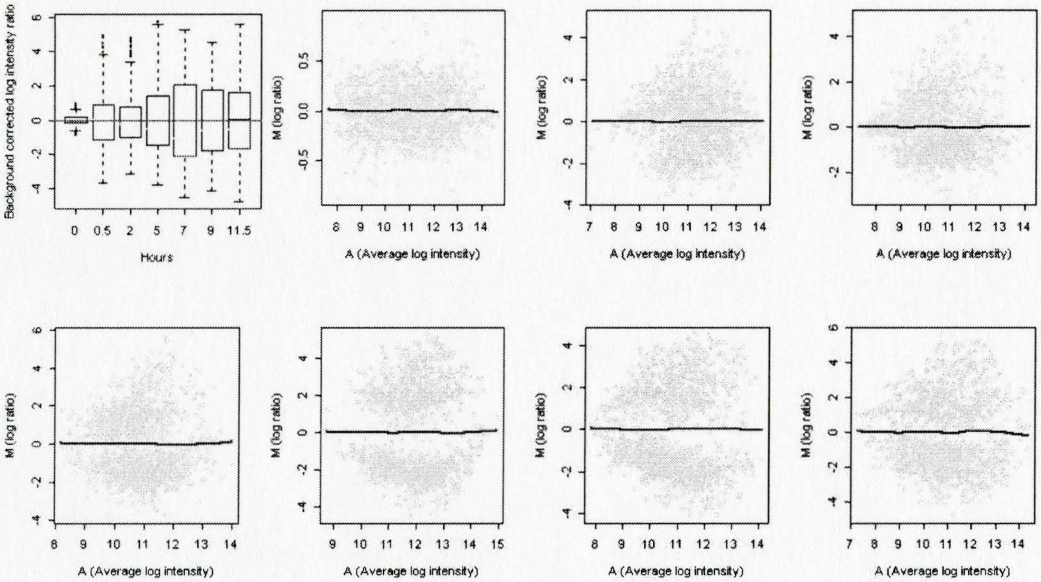


Figure 4.2: The boxplot and MA-plots of 1148 filtered genes over seven time points in order from 0 to 11.5 hours (after normalization).

Chapter 5

Cluster Analysis

Most gene expression data come from microarray experiments with low number of replicates or without replicates because of high cost. It is a shortcoming for applying the inferential statistical methods, such as t-test or ANOVA to find the differentially expressed genes. However, in the time-course micro-array experiments, the objectives are different. We are not testing whether the expression level of a gene in the disease equals that in the normal state, but asking whether profiles over time are similar and we are comparing these time-courses from gene to gene. For example, consider a cluster of genes involved in the cell division cycle, for each individual gene in the cluster, the measurement of expression level might fluctuate due to noise at a given time point so that it might fall out of the cluster. However, if the measurements for the gene at all time points are considered together, the cluster will become robust and thus overcome the noise in individual gene measurement [Baldi *et al.* 2002].

Cluster analysis [Hartigan 1975; Kaufman and Rousseeuw 1990; Everitt 1974; Struyf *et al.* 1996] is a simple but proven method for analyzing gene expression data

[Sherlock 2001]. It is used here as a grouping technique to find genes with similar expression profiles. Generally clustering methods can be divided into two basic types: hierarchical and partitional clustering.

Hierarchical clustering proceeds successively either by merging smaller clusters, initially starting with singleton, into larger ones (agglomerative methods), or by splitting larger clusters, initially starting with one cluster of all objects, until each object is separate (divisive methods). These hierarchical clustering methods differ in the rule of how small clusters are merged or how large clusters are split. The end result of the algorithm is a tree of clusters, called a dendrogram, which shows how the clusters are related. The disjoint clusters are obtained by cutting the dendrogram at a desired level. Partitional clustering attempts to decompose the data set into a set of disjoint clusters by minimizing the measure of dissimilarity within each cluster, while maximizing the dissimilarity of different clusters.

An important component of a clustering algorithm is the distance measure (metric) between data points to assess the dissimilarity or similarity among the objects of a data set. Applying the same clustering algorithm but with a different metric might result in different clusters. Two most commonly used distance measures are the Euclidean distance d, which is scale-dependent and takes into account the magnitude of the variables, and 1-Pearson's correlation coefficient, $(1-r)$, which is scale/location-invariant and insensitive to the magnitude of variables.

5.1 K-means Clustering Method

The specific clustering algorithm and metric (distance measure) need to be selected as well as the appropriate number of clusters within the data. The K-means algorithm is known as a partitioning clustering method which partitions the N objects iteratively into the predefined number of clusters ($K \ll N$).

Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ denote the p-dimensional objects to be clustered. For a given K, the K-means algorithm attempts to minimize the sum of the squared distance of an object within the cluster to the cluster center (centroid):

$$W = \sum_{j=1}^{K} \sum_{i \in C_j} ||\mathbf{x}_i - \bar{\mathbf{x}}_{C_j}||^2,$$

where C_j is j^{th} cluster and $\bar{\mathbf{x}}_{C_j}$ is its center, $\bar{\mathbf{x}}_{C_j} = \frac{1}{N_{C_j}} \sum_{i \in C_j} \mathbf{x}_i$
The basic algorithm of K-means [Hartigan, 1975] is:

1. Specify initial set of K centroids or randomly select K objects from the given data set to be clustered.

2. Assign each object to the closest cluster center such that K clusters are constructed.

3. Calculate the new centers of K clusters (the average of all the members within a certain cluster).

4. Calculate the sums of within cluster sum of squares, W and for iteration i, denote this by W_i. For $i = 1$, go to step 2. For $i \geq 2$, compare W_i with W_{i-1} according to some criterion (e.g. $|W_i - W_{i-1}| < 0.001$). If this criterion is satisfied or

this criterion can not be satisfied but the number of iterations exceeds a certain number of iterations (e.g. 100), exit the algorithm, otherwise go back to step 2.

The K-means is sensitive to the selection of the initial partition. It often terminates at a local minimum and is not suitable to discover clusters with non-convex shapes. Randomly chosen initial centroids can lead to empty cluster(s) and result in the algorithm failure. A more satisfactory alternative [Costa *et al.* 2004] is to use the hierarchical clustering method to provide initial centroids to the K-means. The hierarchical tree can be cut to produce K clusters and set the K mean vectors as the initial centroids for K-means algorithm.

In the recent comparative studies of clustering methods, Costa *et al.* (2004) suggested that as a whole, K-means achieves high accuracies in all experiments. Gibbons and Roth (2002) studied two ratio-based and two Affymetrix-based microarray data sets and concluded that hierarchical clustering tends to produce worse-than-random results.

Chen *et al.* (2002) applied 4 indices to evaluate the performance of clustering algorithms. Let $g_i^{(j)}$ be the member of cluster C_j and $\bar{C}_j$ be the center of the cluster. N is the total number of genes and N_{C_j} is the number of genes in cluster C_j. D is the Euclidean distance function. Then index of homogeneity and separation scores is defined as

$$H\text{average} = \frac{1}{N}\sum_{j=1}^{N_{C_i}} D(g_j^{(i)}, \bar{C}_i)$$

and

$$S\text{average} = \frac{1}{\sum_{i\neq j} N_{C_i}N_{C_j}}\sum_{i\neq j} N_{C_i}N_{C_j}D(\bar{C}_i, \bar{C}_j)$$

H_{average} reflects the compactness of the clusters while S_{average} reflects the overall distance between clusters. Decreasing H_{average} or increasing S_{average} suggests an improvement in the clustering results. Silhouette width proposed by Rousseeuw (1987) is a composite index reflecting the compactness and separation of the clusters. A larger averaged silhouette width indicates a better overall quality of the clustering result [Rousseeuw 1987]. WADP (weighted average discrepant pairs) was proposed by Bittner *et al.* (2000) to test the robustness of clustering results after small perturbation. This is important in microarray expression data analysis because there is always experimental noise in the data. A good clustering result should be insensitive to the noise and able to capture the real structure in the data, reflecting the biological processes under investigation. WADP equals zero when two clustering results match perfectly. In the worst case, WADP is close to one. After evaluating these indices, Chen *et al.* (2002) suggested that K-means generated clusters with slightly better structural quality than others.

The distance metric used with the clustering algorithm will affect the clustering results as well. Gibbons and Roth (2002) demonstrated that for ratio-style data, Euclidean distance is better than or equal to the other measures (e.g. Pearson correlation distance, Manhattan distance, etc.). Therefore, Euclidean distance metric was selected for K-means cluster analysis of yeast gene expression data.

5.2 Estimating the Number of Clusters in Data

One of the problems in cluster analysis is how to estimate the appropriate number of clusters in the data set. For most clustering methods (e.g. K-means), the user must specify the number of clusters. Therefore, estimating the "true" number of clusters in the data set must be done in an iterative fashion, running the cluster algorithm for a set of plausible values of K. Because most microarray data sets have higher dimensions, visualization tools are difficult to use in finding a reasonable number of clusters in a high dimensional data set, if we visualize clusters for each variable. A good clustering algorithm should yield clusters which have high intra-class similarity and low inter-class similarity. Therefore, there is an intuitive way to find the appropriate number of clusters. First we apply the clustering algorithm with different number of clusters ($k = 1, 2, \ldots, K$) to the data set. For each k, we calculate the ratio of average euclidian distance within and between clusters. The appropriate number of clusters should occur at lowest ratio. However, numerous other approaches to this issue have been proposed from many studies. A comprehensive survey of methods for estimating the number of clusters is given by Milligan and Cooper (1985). Tibshirani *et al.* (2001) suggested a statistical approach with some theoretical development and involving what was called the gap statistic. The method is applicable to virtually any clustering algorithm (A implementation of the gap statistic using R package was written for the project and is given in the Appendix A.2).

Let $d_{ii'}$ be the distance between observations i and i'. Suppose a clustering algorithm has generated m clusters, $C_1, C_2, \ldots, C_m$, with C_r denoting the indices of the observations in cluster r and $n_r = |C_r|$ the cluster size. Let $D_r = \sum_{ii' \in C_r} d_{ii'}$ and $W_k = \sum_{r=1}^{k} \frac{D_r}{2n_r}$. If $d_{ii'}$ is squared Euclidean distance then W_k is the pooled within cluster sum of squares around cluster means. The basic idea of the gap static is to compare $log(W_k)$ to its expectation under an appropriate null reference distribution.

The gap statistic is defined as

$$Gap_n(k) = E_n^*(log(W_k)) - log(W_k)$$

where E^* denotes expectation under a sample of size n from the reference distribution. The estimate $\hat{k}$ will be the value maximizing $Gap_n(k)$ after taking the sampling distribution into account. The expectation of the reference distribution $E_n^*(log(W_k))$ can be estimated as $(1/B)\sum_{b=1}^{B}(log(W_{kb}^*)$ where W_{kb}^* is the within-cluster sum of squares of the bth Monte Carlo replicate of W_k^*.

From a graphic view, within-cluster sum of squares W_k is a decreasing function of k. we look for a turning point of elbow-shape to identify the number of clusters, $\hat{k}$.

5.3 Gap Statistic Algorithm

The notations and the steps follow Tibshirani *et al.* (2001).

Notation:

$\{x_{ij}\}$	$n \times p$ normalized log intensities ratio matrix
$d_{ii'}$	the Euclidean distance between point i and i'
C_r	the indices of points in cluster r and $n_r = \|C_r\|$
D_r	the sum of pairwise distances for all points in cluster r

Steps:

1. cluster the intensities ratio matrix at the number of clusters $k = 1, 2, \ldots, K$.

2. for $k = 1, 2, \ldots, K$ compute the sum of pairwise distances for all points in cluster

r and the within cluster dispersion measures W_k:

$$\begin{aligned} D_r = & \quad \textstyle\sum_{i,i' \in C_r} d_{ii'} \\ = & \textstyle\sum_{i \in C_r} \sum_{i' \in C_r} \|\mathbf{x}_i - \mathbf{x}_{i'}\|^2 \\ = & \quad 2n_r \textstyle\sum_{i \in C_r} \|\mathbf{x}_i - \bar{\mathbf{x}}\|^2 \end{aligned}$$

and

$$W_k = \sum_{r=1}^{K} \frac{1}{2n_r} D_r$$

.

3. generate B reference matrices using one of methods below:

   (a) generate each reference feature uniformly over the range of the observed values for that feature.

   (b) Generate the reference features from a uniform distribution over a box aligned with the principal components of the data.

   Note: Because we have normalized the data matrix X, the columns have mean zero. Compute the singular value decomposition $X = UDV^T$ and transform via $X' = XV$ and then draw uniform features Z' over the ranges of the columns of X', as in method a above. Back transform via $Z = Z'V^T$ to give reference data Z.

4. for each reference matrix, compute the within cluster dispersion measures $W^*_{kb}, b = 1, 2, \ldots, B$ and $k = 1, 2, \ldots, K$ as in step 2.

5. compute the estimated gap statistic,

$$Gap(k) = (1/B)\sum_{b} log(W^*_{kb}) - log(W_k)$$

.

6. compute standard deviation sd_k,

$$sd_k = [(1/B)\sum_{b}(log(W^*_{kb}) - \bar{l})^2]^{1/2}$$

where $\bar{l} = (1/B)\sum_b log(W^*_{kb})$.

7. define the simulation error $s_k = sd_k\sqrt{1 + 1/B}$.

8. find the optimal number of clusters by $\hat{k}$ = smallest k such that

$$Gap(k) \geq Gap(k+1) - s_{k+1}.$$

5.4 Fitting Polynomial Curves For Clusters

In the comparative studies of clustering algorithms, it was also shown that no single clustering algorithm is the best approach [Chen *et al.* 2002]. Combining these methods we may succeed in getting more meaningful clustering results. R source code is provided to fit a fourth-order polynomial curve, expected to be adequate with only 7 time points, to each cluster to summarize the pattern of change in expression over time for all genes in a cluster. The fourth-order polynomial model is defined as:

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 x_i^4 + \varepsilon_i$$

where y represents log intensity ratio in a cluster (n_k=the number of genes in cluster k) and x_i represents 7 time points. In our case, the matrix X is

$$X_{7n_k \times 5}= \begin{array}{c} \begin{array}{ccccc} 1 & \mathrm{x}_i & \mathrm{x}_i^2 & \mathrm{x}_i^3 & \mathrm{x}_i^4 \end{array} \\ \left(\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \\ 1 & 0.5 & 0.25 & 0.125 & 0.0625 \\ 1 & 2 & 4 & 8 & 16 \\ 1 & 5 & 25 & 125 & 625 \\ 1 & 7 & 49 & 343 & 2401 \\ 1 & 9 & 81 & 648 & 5832 \\ 1 & 11.5 & 132.25 & 1520.875 & 17490.0625 \\ \vdots & & & & \vdots \\ & & & & \\ 1 & 7 & 49 & 343 & 2401 \\ 1 & 9 & 81 & 648 & 5832 \\ 1 & 11.5 & 132.25 & 1520.875 & 17490.0625 \end{array} \right) \end{array} \begin{array}{l} \\ \\ \\ \\ \\ \text{gene } 1 \\ \\ \\ \vdots \\ \\ \text{gene } n_k \\ \\ \\ \end{array}$$

Therefore, the parameters β_i can be estimated by $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y}$. The fit of the polynomial curve can be measured by the coefficient of determination, R^2. When the mean is contained in the model, the coefficient of determination corrected for the mean, R_m^2, is defined as the ratio of the sum of squares model corrected for the mean to the sum squares total corrected for the mean [Goldsmith 1974],

$$R_m^2 = \frac{\mathrm{SSM}_m}{\mathrm{SST}_m} = \frac{\mathrm{SSM} - \mathrm{MSS}}{\mathrm{SST} - \mathrm{MSS}},$$

where the SSM is the sum of squares model, the MSS is the sum of squares due to mean and the SST is the sum of squares total.

5.5 Clustering of Yeast Data

Initial centroids for K-means were obtained using `diana(X)`[Kaufman and Rousseeuw 1990]. The K-means clustering was performed with improvement using R function `Cluster(X,k,` hc) in Appendix A.5. Gap statistic implementation was written in R and given in Appendix A.2. R source code for plots of profiles of genes within cluster and fitted polynomial curves was given in Appendix A.3.

Initially the shape of profiles for genes in a cluster and number of genes in cluster was explored for $K = 20, 25, 30, 40$ using the first 1000 genes of 6118 genes in the data set. This was done to understand the nature of the profiles and it was found that some large clusters showed essentially no change of expression over time, others with various shapes stayed fairly stable with change of K and there were some small clusters with atypical profiles. It's also done for all 6118 genes. This provides support for the use of criterion of Chu, since a large number of genes (3391 when clustering done on all 6118 genes) fall into clusters with constant expression over time.

The gap statistic with K-means and initial centroids obtained from hierarchical clustering was applied to the 1148 genes meeting RMS criterion. The maximum number of clusters was set at 30 and the reference matrices were generated at step 3 of gap statistic algorithm using a uniform distribution over the range of observed values (option a), and the number of clusters was determined to be 9. The results of gap statistic are given in Figure 5.1 and Figure 5.2.

```
> gapstat(newdat3[,2:8], K=30, B=50)
$Gapk
 [1] 0.6930711 1.8283406 1.9632833 1.9878576 2.1212461 2.1671136 2.2256047
 [8] 2.2399751 2.2782241 2.2843147 2.3134124 2.3169155 2.3122075 2.3167583
[15] 2.3037693 2.2868366 2.2999302 2.3037831 2.3056700 2.2893878 2.3029118
[22] 2.3146344 2.3375083 2.3416686 2.3438345 2.3404221 2.3485463 2.3437400
[29] 2.3375174 2.3356186

$Sk
 [1] 0.01082040 0.01181728 0.01189092 0.01106499 0.01143219 0.01137687
 [7] 0.01145138 0.01235576 0.01201024 0.01403692 0.01335720 0.01213183
[13] 0.01276264 0.01100956 0.01134031 0.01147210 0.01248416 0.01323230
[19] 0.01363560 0.01227602 0.01197334 0.01130182 0.01148524 0.01125200
[25] 0.01263769 0.01287899 0.01229486 0.01202497 0.01101910 0.01110992

$Sdk
 [1] 0.01071379 0.01170085 0.01177376 0.01095597 0.01131956 0.01126478
 [7] 0.01133856 0.01223403 0.01189191 0.01389863 0.01322560 0.01201231
[13] 0.01263690 0.01090109 0.01122858 0.01135907 0.01236116 0.01310193
[19] 0.01350126 0.01215507 0.01185537 0.01119047 0.01137208 0.01114114
[25] 0.01251318 0.01275210 0.01217372 0.01190650 0.01091053 0.01100046

$Diff
 [1] -1.1234522156 -0.1230517764 -0.0135093180 -0.1219563227 -0.0344906474
 [6] -0.0470397419 -0.0020145691 -0.0262388341  0.0079463461 -0.0157405043
[11]  0.0086287507  0.0174706714  0.0064587250  0.0243293216  0.0284047391
[16] -0.0006093439  0.0093793830  0.0117486228  0.0285582745 -0.0015506225
[21] -0.0004208016 -0.0113886906  0.0070917386  0.0104717889  0.0162914084
[26]  0.0041706051  0.0168312385  0.0172417361  0.0130086828

$Khat
[1] 9
```

Figure 5.1: The results were obtained from 1148 filtered genes by the gap statistic algorithm with the K-means clustering method used to generate the clusters, DIANA used to compute the initial centers for the K-means, The maximum number of clusters K=30, and the Monte Carlo replicate B=50.

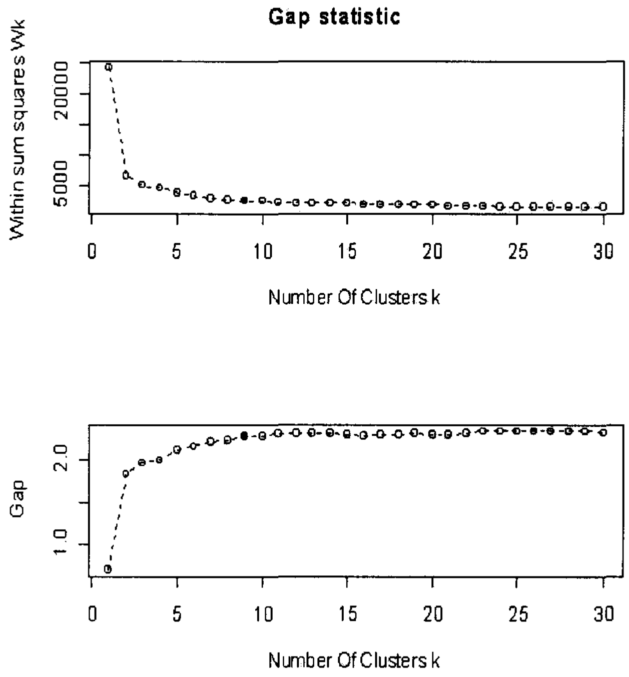


Figure 5.2: The plot of gap statistic with $\hat{k} = 9$ shown as solid dot. The results were obtained from 1148 filtered genes and the gap statistic algorithm.

The changing pattern in expression level over time for each cluster can be dynamically shown by the plot of profiles of all genes in a cluster, with the polynomial fitted to the log ratios for all genes in the cluster. This is shown for the 9 clusters on the selected set of 1148 genes in Figure 5.3. Clusters 1, 2, 5, 7 and 9 have mean profiles that show up regulation over the entire period of observation, but with some markedly different patterns. For example, the cluster 1 mean profile shows increased expression in the early part of the period of observation, whereas, that of cluster 5 shows sharp increase from $t = 0$ to a maximum at about time 7 and then it levels off. Clusters 3, 4, 6 and 8 show down regulation over the entire period of observation, with cluster 8 showing a steady decline.

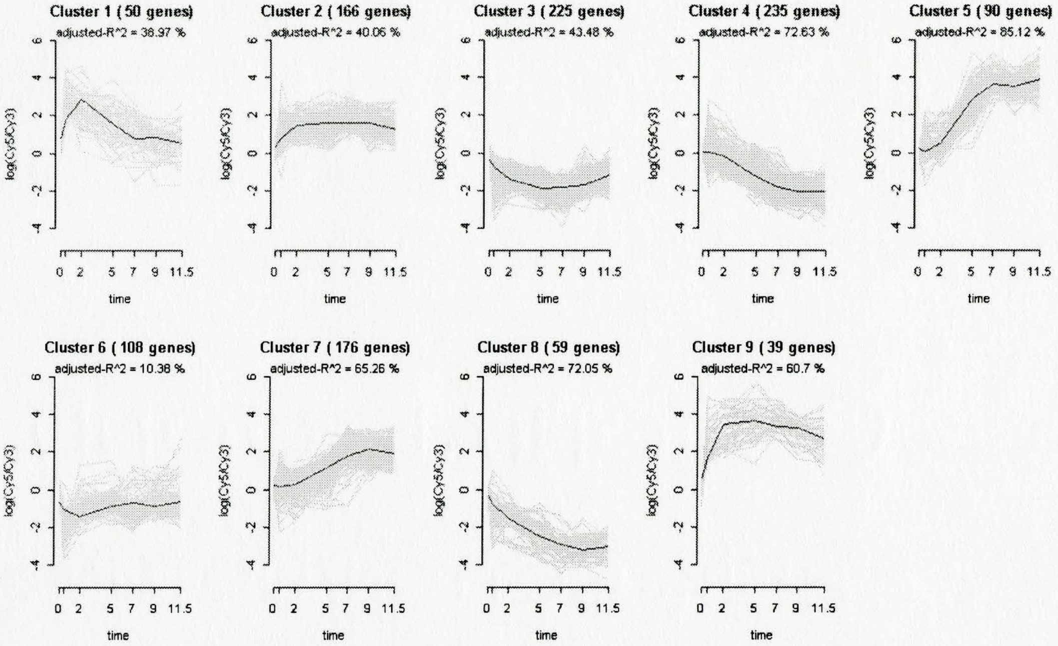


Figure 5.3: The gene patterns for 9 clusters, where clusters were obtained from 1148 filtered genes by K-means clustering algorithm and the gap statistic to determine the number of clusters. Each grey line consists of the 7 points (time, expression level) for one gene joined by lines. The black curve is the polynomial fitted to the data for all genes in the cluster.

The value of R^2 associated with the 4th degree polynomial fitted to the data in a cluster gives an indication of which clusters have genes with more similar profiles. There is less variability around the mean profile in clusters 4, 5 and 8, for which R2 is 0.72, 0.73 and 0.85 respectively. The variability that is present at each time point is also informative and is shown in the Figure 5.4. In each cluster, time 0 expression is very muck less variable than at other points because array t0 was compared with itself.

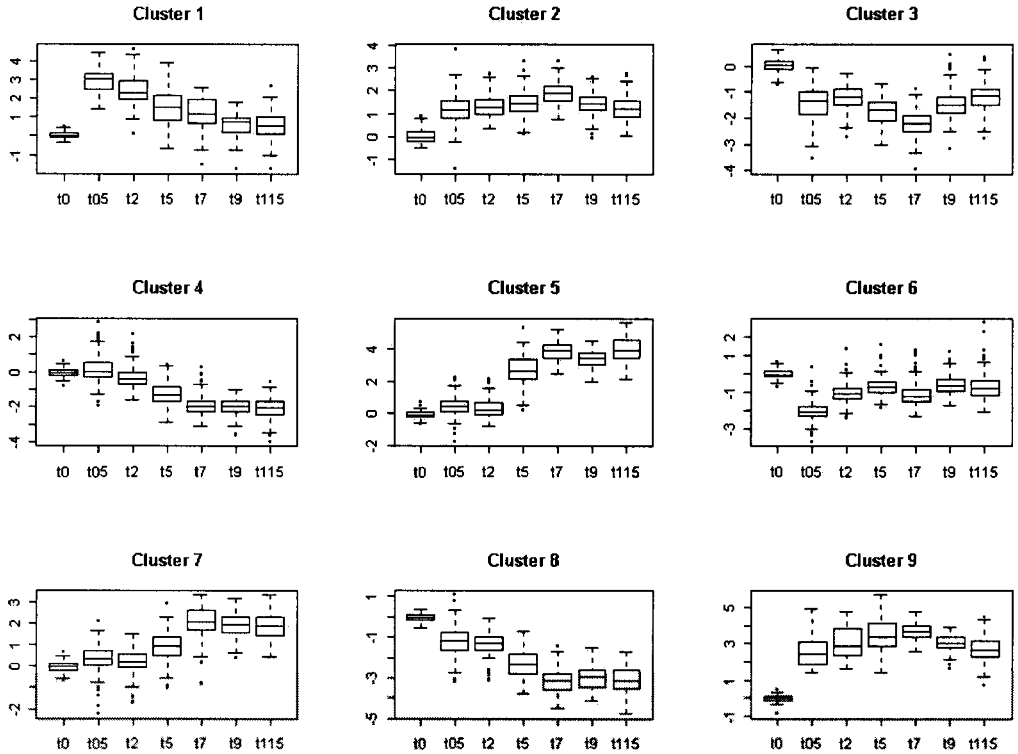


Figure 5.4: Box plots of expression levels by time point within cluster for the same clusters as shown in Figure 5.3.

Chapter 6

Interpretation of Clustering Results Using Gene Ontology

Interpretation of clustering results using Gene Ontology is the final step of microarray data analysis. Some clusters of genes that share similar patterns of expression have been identified. But to provide a biological interpretation, similarity of the genes in a cluster needs to be summarized with respect to biological features, such as their molecular functions, their roles in biological processes, and their presence in cellular components.

An ontology is a description of concepts. Gene Ontology [Ashburner *et al.* 2000; http:// www.geneontology.org] provides the standard terms with consistent biological descriptions for the gene annotation of different organisms (e.g. yeast). GO is a biological knowledge database about genes which is organized in three independent ontologies: molecular function (e.g. catalytic activity) which refers to the tasks performed by individual genes; biological process (e.g. pyrimidine metabolism) which refers to the biological function with which a gene is associated; and cellular component (e.g. ribosome) which refers to the subcellular structure locations. Each gene is annotated with GO terms which are structured in a hierarchy, ranging from more general (higher level) to more specific (lower level). Therefore, a term at a lower level has one (or more) parent term(s) at the upper level. It is possible that one gene is annotated with more than one GO term. The level 3 is the best compromise between quantity and quality of GO information [Conde *et al.* 2002; Mateos *et al.* 2002]. The systematic gene names (e.g. YBR166C) are used to map the relationship between the gene and its GO terms and extract the biologically common characteristics in the groups of genes under study.

FatiGO [Al-Shahrour *et al.* 2004; http://www.fatigo.org/] is a web tool for finding the most characteristic Gene Ontology terms for each cluster or comparing two groups of genes to give those GO terms which are statistically significant in two groups using multiple testing.

Given a fixed level and one of three ontologies, for each GO term, FatiGo counts the number of genes in the group with the term at the given level. The percentage for the GO term is computed by

$$\text{percentage} = \frac{\text{the number of genes in the group with the term at the given level}}{\text{the total number of genes in the group}} \times 100\%$$

or

$$\text{percentage} = \frac{\text{the number of genes in the group with the term at the given level}}{\text{the total number of genes in the group} - \text{the number of genes without GO annotated}} \times 100\%$$

The percentages are ordered from higher to lower. Therefore, the dominant GO terms can be found in the group of genes.

For comparing one group of genes (typically a cluster) with the reference gene group (typically all genes except group to be compared), a Fisher's exact test for

2x2 contingency tables is applied for each GO term, with the null hypothesis of no difference in the frequency of the given term in each group.

	Compared group	Reference group	
Given GO term present	n_{11}	n_{12}	$n_{1.}$
Given GO term absent	n_{21}	n_{22}	$n_{2.}$
	$n_{.1}$	$n_{.2}$	N

The p-value from Fisher's exact test for the given GO term is computed by using hypergeometric distribution:

$$\text{p-value} = \sum_{k=n_{11}}^{n_{.1}} \frac{\begin{pmatrix} n_{1.} \\ k \end{pmatrix} \begin{pmatrix} N - n_{1.} \\ n_{.1} - k \end{pmatrix}}{\begin{pmatrix} N \\ n_{.1} \end{pmatrix}}$$

where N is the number of genes in the genome (6118 for yeast), $n_{.1}$ is the number of genes in the cluster, $n_{1.}$ is the number of the given GO terms and n_{11} is the number of this GO term in the cluster.

For each test, given a significant level α, the chance of making a type I error is just α. When n independent tests are carried, the chance of making at least one type I error in the n test is at most $1-(1-\alpha)^n$ (e.g. $1-(1-0.05)^{10} = 0.40$). Since it is likely to get a number of false rejections just by chance, the individual p-value can not be directly used to check whether the corresponding GO term is statistically significant.

FatiGO returns one unadjusted p-value from Fisher's exact test and three adjusted p-values based on three different ways of accounting for multiple testing: step-down minP method [Westfall and Young 1993] which provides control of the family wise error rate; False Discovery Rate (FDR) method [Benjamini and Hochberg 1995] which provides control of the FDR only under independence and some specific type of positive dependence of tests statistics. and False Discovery Rate method [Benjamini and Yekutieli 2001] which provides strong control under arbitrary dependency of test statistics. The details for adjusted p-values are described as follows (e.g. FDR of Benjamini and Hochberg 1995): Suppose that $p_1, p_2, \ldots, p_n$ are n observed p-values for n GO terms from Fisher's exact test. Order them from the smallest to the largest as $p_{(1)}, p_{(2)}, \ldots, p_{(n)}$, then the i^{th} adjusted p-value [Dudoit *et al.* 2003] is:

$$p_{(i)}^{adj} = \min_{k \in \{i, \ldots, n\}} \{\min(\frac{n}{k} * p_{(k)}, 1)\}, i = 1, \ldots, n.$$

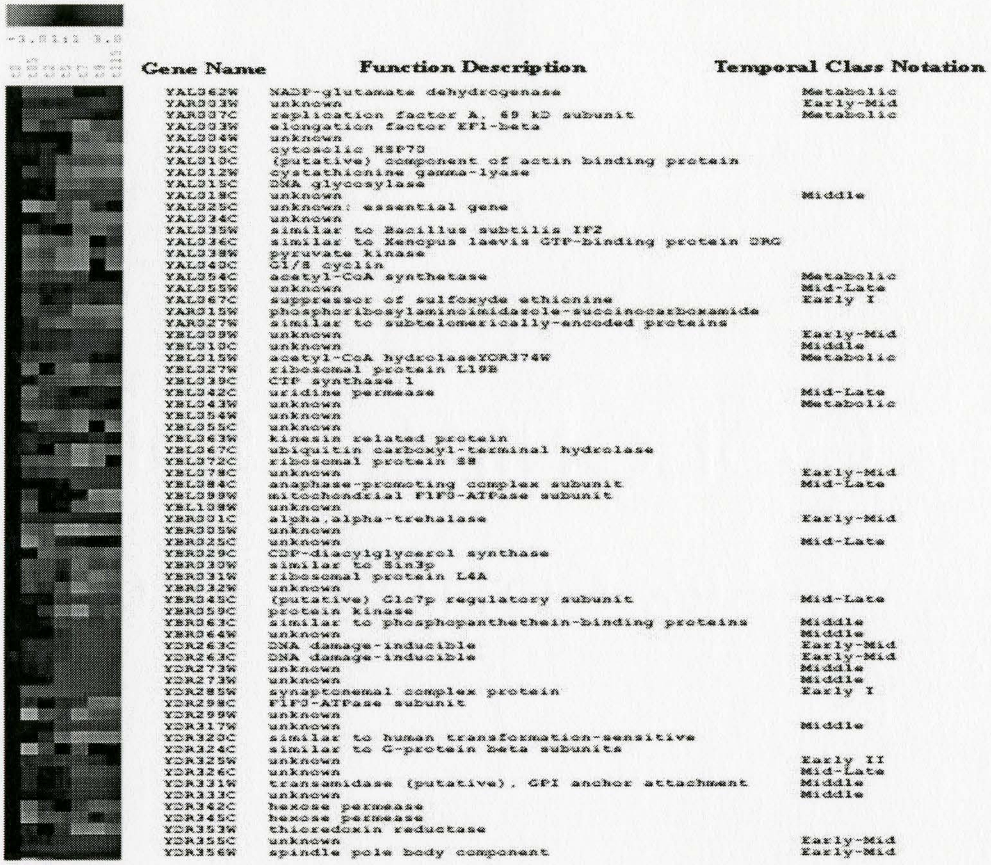
.

Chapter 7

Interpretation of Yeast Cluster Results

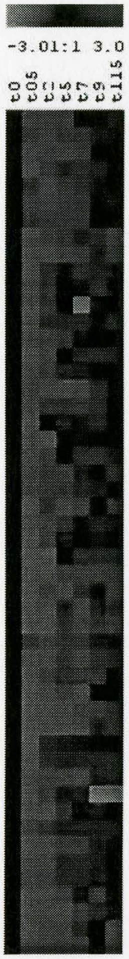
In the analysis of the sporulation data using the method of Chu *et al.* (1998) given in Section 3.2 above, 1148 genes were retained from 6118 genes by RMS criterion (RMS $\geq$ 1.13) and the rest of the genes which did not show significant changes were removed. Chu *et al.* (1998) noted that about half of 1148 genes were induced (log ratio of expression greater than 0 and the spot color is red), and half were repressed (log ratio of expression less than 0 and the spot color is green). For the analysis performed here, the changing pattern of expression level during sporulation was shown in Figure 5.3.

The amount/abundance of the mRNA levels (expression levels) in the two biological samples can be indirectly measured by the intensities of the two dyes. The entire set of gene expressions over 7 time points can be visualized as a heatmap image [Shannon *et al.* 2003; Saeed *et al.* 2003]. The heatmap presents a grid of colored points where each color represents a ratio of gene expression value in the samples, where colors of red, green and black represent up-regulated, down-regulated, and unchanged genes respectively. The heatmap is used to display the expression patterns of a group of genes in a graphical format. The color intensity of each spot in the heatmap is proportional to the gene expression ratio at this spot. Red and green represent positive and negative value respectively. The brightness represents relatively higher positive or negative values. Comparison of Figure 7.1, genes which have not been clustered, with Figures 7.2, 7.3 and 7.4 provide another means of seeing the pattern of expression represented by genes in a cluster. These can be compared with the plots for corresponding clusters in Figure 5.3. These heatmaps have been produced using the free Java package of TM4 (Saeed *et al.* 2003) and uploading clustering results (the data format in Appendix A.7) into TM4.



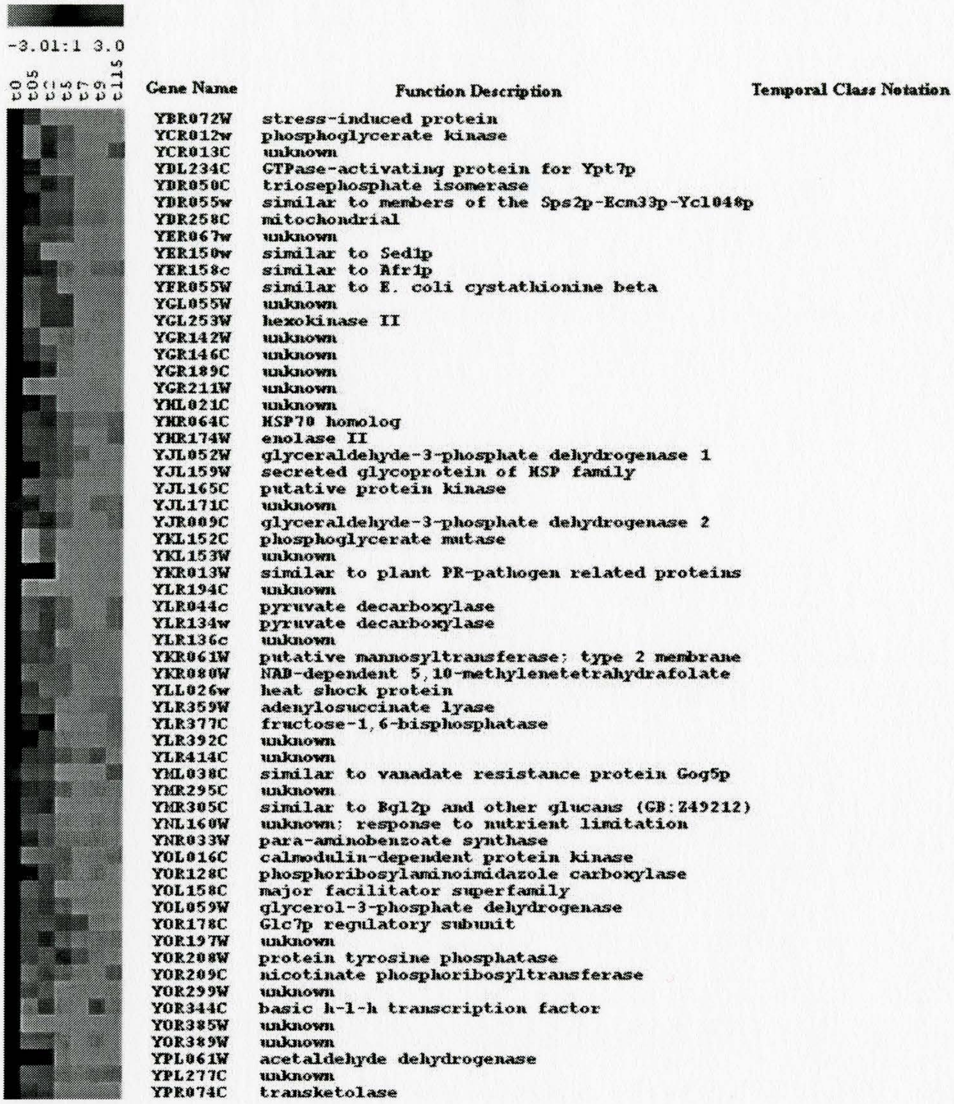
Gene Name	Function Description	Temporal Class Notation
YAL062W	NADP-glutamate dehydrogenase	Metabolic
YAR003W	unknown	Early-Mid
YAR007C	replication factor A, 69 kD subunit	Metabolic
YAL003W	elongation factor EF1-beta	
YAL004W	unknown	
YAL005C	cytosolic HSP70	
YAL010C	(putative) component of actin binding protein	
YAL012W	cystathionine gamma-lyase	
YAL015C	DNA glycosylase	
YAL018C	unknown	Middle
YAL025C	unknown: essential gene	
YAL034C	unknown	
YAL035W	similar to Bacillus subtilis IF2	
YAL036C	similar to Xenopus laevis GTP-binding protein DRG	
YAL038W	pyruvate kinase	
YAL040C	G1/S cyclin	
YAL054C	acetyl-CoA synthetase	Metabolic
YAL055W	unknown	Mid-Late
YAL067C	suppressor of sulfoxyde ethionine	Early I
YAR015W	phosphoribosylaminoimidazole-succinocarboxamide	
YAR027W	similar to subtelomerically-encoded proteins	
YBL009W	unknown	Early-Mid
YBL010C	unknown	Middle
YBL015W	acetyl-CoA hydrolaseYOR374W	Metabolic
YBL027W	ribosomal protein L19B	
YBL039C	CTP synthase 1	
YBL042C	uridine permease	Mid-Late
YBL043W	unknown	Metabolic
YBL054W	unknown	
YBL055C	unknown	
YBL063W	kinesin related protein	
YBL067C	ubiquitin carboxyl-terminal hydrolase	
YBL072C	ribosomal protein S8	
YBL078C	unknown	Early-Mid
YBL084C	anaphase-promoting complex subunit	Mid-Late
YBL099W	mitochondrial F1F0-ATPase subunit	
YBL108W	unknown	
YBR001C	alpha,alpha-trehalase	Early-Mid
YBR005W	unknown	
YBR025C	unknown	Mid-Late
YBR029C	CDP-diacylglycerol synthase	
YBR030W	similar to Sin3p	
YBR031W	ribosomal protein L4A	
YBR032W	unknown	
YBR045C	(putative) Glc7p regulatory subunit	Mid-Late
YBR059C	protein kinase	
YBR063C	similar to phosphopanthethein-binding proteins	Middle
YBR064W	unknown	Middle
YDR263C	DNA damage-inducible	Early-Mid
YDR263C	DNA damage-inducible	Early-Mid
YDR273W	unknown	Middle
YDR273W	unknown	Middle
YDR285W	synaptonemal complex protein	Early I
YDR298C	F1F0-ATPase subunit	
YDR299W	unknown	
YDR317W	unknown	Middle
YDR320C	similar to human transformation-sensitive	
YDR324C	similar to G-protein beta subunits	
YDR325W	unknown	Early II
YDR326C	unknown	Mid-Late
YDR331W	transamidase (putative), GPI anchor attachment	Middle
YDR333C	unknown	Middle
YDR342C	hexose permease	
YDR345C	hexose permease	
YDR353W	thioredoxin reductase	
YDR355C	unknown	Early-Mid
YDR356W	spindle pole body component	Early-Mid

Figure 7.1: The heatmap represents the first 67 of the 1148 filtered genes data set before clustering. The colour patterns which represent genes profiles can not be recognized easily. The columns to the right of the heatmap are that systematic gene name, the description of gene function and temporal class notations. Seven temporal class notations associated with some genes means that these genes were induced at the seven transcription stages. The temporal class notations were given in the original data set.



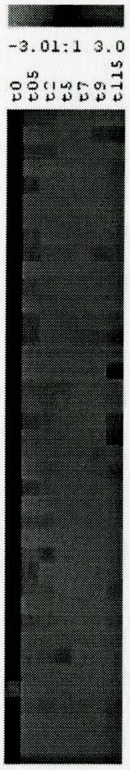
Gene Name	Function Description	Temporal Class Notation
YAL062W	NADP-glutamate dehydrogenase	Metabolic
YAR007C	replication factor A, 69 kD subunit	Metabolic
YAL054C	acetyl-CoA synthetase	Metabolic
YBL043W	unknown	Metabolic
YBR088C	DNA polymerase processivity factor	Metabolic
YCR005c	peroxisomal citrate synthase	Early I
YDR180W	unknown; binds chromosomes	Early I
YDR256C	catalase A	Metabolic
YER024w	similar to Yat1p	Metabolic
YER055c	ATP phosphoribosyltransferase	Metabolic
YER069w	acetylglutamate kinase and	Metabolic
YER091c	homocysteine methyltransferase	Metabolic
YFR030W	sulfite reductase subunit	Metabolic
YGL062W	pyruvate carboxylase 1	Metabolic
YGR067C	unknown	Metabolic
YGR239C	unknown	Metabolic
YHL024W	similar to RNA-binding proteins in the N-terminal	Early I
YHL030W	unknown	Metabolic
YHR053C	metallothionein	Metabolic
YIR029W	allantoicase	Metabolic
YIR042C	unknown	Metabolic
YJL045W	similar to succinate dehydrogenase flavoprotein	Early I
YJL089W	transcription factor	Metabolic
YJL153C	peripheral membrane protein	Metabolic
YJR016C	dihydroxyacid dehydratase	Metabolic
YJR109C	carbamyl phosphate synthetase	Metabolic
YJR152W	allantoate permease	Metabolic
YKL120W	similar to members of the mitochondrial carrier	Early I
YKR009C	peroxisomal beta-oxidation protein	Metabolic
YKR033C	similar to Gat1p	Metabolic
YKR034W	transcription factor	Metabolic
YKR071C	unknown	
YLL027W	unknown	Early I
YLR303W	O-acetylhomoserine sulfhydrylase	Metabolic
YLR304C	aconitase	Metabolic
YLR438W	ornithine aminotransferase	Metabolic
YML042W	carnitine O-acetyltransferase	Metabolic
YMR018W	similar to Pex5p/Pas10p (GB:Z49211)	Metabolic
YMR095C	unknown; induced in stationary phase	Metabolic
YMR147W	unknown	Early I
YNL117W	malate synthase	Metabolic
YNL142W	ammonia permease	Metabolic
YNL202W	peroxisomal 2,4-dienoyl-CoA reductase	Early I
YOR100C	similar to members of the mitochondrial carrier	Metabolic
YOL125W	unknown	Metabolic
YOR225W	unknown	Metabolic
YOR375C	glutamate dehydrogenase	Metabolic
YPL111W	arginase	Metabolic
YPR002W	similar to Bacillus subtilis MMGE protein	Metabolic
YPR006C	isocitrate lyase, nonfunctional	Metabolic

Figure 7.2: The heatmap of cluster 1 which contains 50 up-regulated genes with higher expression level at 0.5 hour. The cluster is generated from 1148 filtered genes by K-means clustering algorithm.



Gene Name	Function Description	Temporal Class Notation
YBR072W	stress-induced protein	
YCR012w	phosphoglycerate kinase	
YCR013C	unknown	
YDL234C	GTPase-activating protein for Ypt7p	
YDR050C	triosephosphate isomerase	
YDR055w	similar to members of the Sps2p-Ecm33p-Ycl048p	
YDR258C	mitochondrial	
YER067w	unknown	
YER150w	similar to Sed1p	
YER158c	similar to Afr1p	
YFR055W	similar to E. coli cystathionine beta	
YGL055W	unknown	
YGL253W	hexokinase II	
YGR142W	unknown	
YGR146C	unknown	
YGR189C	unknown	
YGR211W	unknown	
YHL021C	unknown	
YHR064C	HSP70 homolog	
YHR174W	enolase II	
YJL052W	glyceraldehyde-3-phosphate dehydrogenase 1	
YJL159W	secreted glycoprotein of HSP family	
YJL165C	putative protein kinase	
YJL171C	unknown	
YJR009C	glyceraldehyde-3-phosphate dehydrogenase 2	
YKL152C	phosphoglycerate mutase	
YKL153W	unknown	
YKR013W	similar to plant PR-pathogen related proteins	
YLR194C	unknown	
YLR044c	pyruvate decarboxylase	
YLR134w	pyruvate decarboxylase	
YLR136c	unknown	
YKR061W	putative mannosyltransferase; type 2 membrane	
YKR080W	NAD-dependent 5,10-methylenetetrahydrafolate	
YLL026w	heat shock protein	
YLR359W	adenylosuccinate lyase	
YLR377C	fructose-1,6-bisphosphatase	
YLR392C	unknown	
YLR414C	unknown	
YML038C	similar to vanadate resistance protein Gog5p	
YMR295C	unknown	
YMR305C	similar to Bgl2p and other glucans (GB:Z49212)	
YNL160W	unknown; response to nutrient limitation	
YNR033W	para-aminobenzoate synthase	
YOL016C	calmodulin-dependent protein kinase	
YOR128C	phosphoribosylaminoimidazole carboxylase	
YOL158C	major facilitator superfamily	
YOL059W	glycerol-3-phosphate dehydrogenase	
YOR178C	Glc7p regulatory subunit	
YOR197W	unknown	
YOR208W	protein tyrosine phosphatase	
YOR209C	nicotinate phosphoribosyltransferase	
YOR299W	unknown	
YOR344C	basic h-l-h transcription factor	
YOR385W	unknown	
YOR389W	unknown	
YPL061W	acetaldehyde dehydrogenase	
YPL277C	unknown	
YPR074C	transketolase	

Figure 7.3: The heatmap of cluster 8 which contains 59 down-regulated genes with higher negative expression level after 5 hour. The cluster is generated from 1148 filtered genes by K-means clustering algorithm.



Gene Name	Function Description	Temporal Class Notation
YBR184W	alpha-galactosidase	Early-Mid
YCR010C	unknown	Early I
YDL239c	unknown	Early-Mid
YDR015c	unknown	Early I
YDR113C	anaphase inhibitor (putative)	Early II
YDR285W	synaptonemal complex protein	Early I
YDR374C	unknown	Early I
YDR446W	unknown	Early II
YER044c-a	unknown	
YER179w	unknown	Early I
YFL003C	MutS homolog	Early II
YGL009C	3-isopropylmalate dehydratase	Early I
YGL033W	unknown	Early II
YGL081W	unknown	Early II
YGL183C	unknown	Early I
YGL251C	DNA/RNA helicase, putative	Early I
YGR108W	G2/M cyclin	Early-Mid
YGR226C	unknown	Early-Mid
YHL022C	ds break formation complex catalytic subunit	Early II
YHR014W	negative regulator of M phase (putative)	Early I
YHR208W	transaminase	Early I
YIL031W	suppresses mif2 mutation	Early I
YIL057C	unknown	Early-Mid
YIL072W	DNA binding protein	Early I
YIL144W	unknown	Early-Mid
YJL106W	protein kinase	Early I
YJR036C	putative ubiquitin-protein ligase	Early-Mid
YLL046c	ribonucleoprotein	Early II
YLL047W	unknown	Early II
YLR263W	synaptonemal complex component (putative)	Early I
YMR133W	ds break formation complex subunit	Early II
YMR198W	spindle pole body associated protein	Early I
YNL155W	unknown	Mid-Late
YNL196C	unknown	Early II
YOL104C	unknown	Early I
YOR177C	unknown	Early I
YOR351C	protein kinase	Early I
YPL121C	unknown	Early II
YPR007C	isocitrate lyase, nonfunctional	Early I

Figure 7.4: The heatmap of cluster 9 contains 39 up-regulated genes with higher expression level after 0 hour. The cluster is generated from 1148 filtered genes by K-means clustering algorithm.

Genes with similar functions are grouped into 9 clusters with good separation (comparing Figure 7.1 with Figures 7.2, 7.3, and 7.4). Five clusters (1, 2, 5, 7, and 9) have up-regulated genes. The majority of genes in these 5 clusters have the temporal class notation (e.g. Metabolic, Early I, Early II, Early-Mid, Middle, Mid-Late, and Late). Four clusters (3, 4, 6, and 8) have down-regulated genes without temporal class notation, which can be verified by their heatmaps. Interestingly it also can be seen in heatmaps of these 9 clusters that genes within the same cluster are either up-regulated or down-regulated over time. This result is consistent with the findings in the paper of Chu *et al.* (1998).

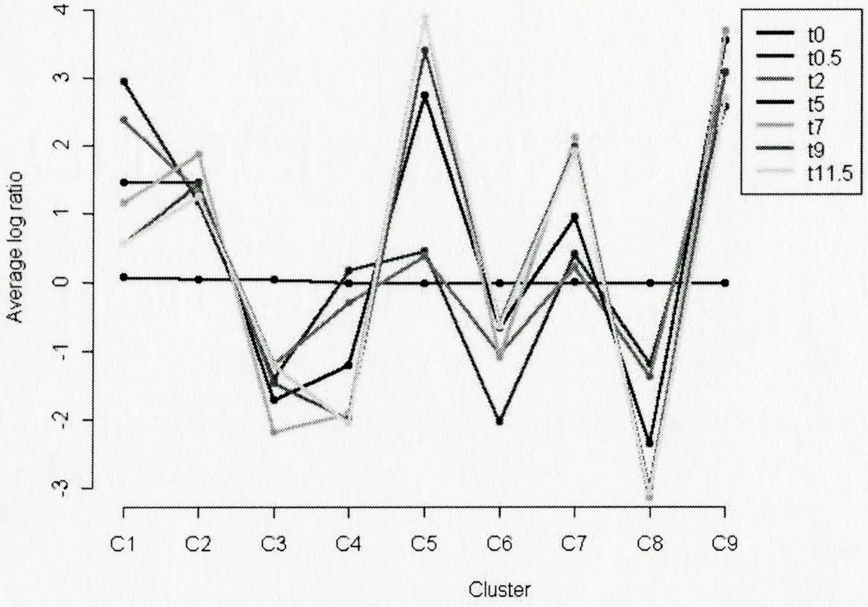


Figure 7.5: The average of gene expression versus clusters.

Some properties of gene clusters can be explored by Figure 7.5. Array t0 was self-hybridized and compared with itself. Therefore, the average of gene expression of each cluster at 0 hour is around zero. The average of gene expression of Cluster 5 reaches the highest level at 11.5 hours. However, the average of gene expression of Cluster 8 reaches the highest level at 7 hours.

In the research of Chu *et al.* (1998), a small, representative set of genes was hand-picked for each of these 7 temporal classes (Table 7.1). All hand-picked genes with the temporal class of Metabolic, Early I, and Early II are grouped into cluster 1, 9, and 2

Metabolic	Early I	Early II	Early-Mid	Middle	Mid-Late	Late
ACS1 (1)	ZIP1 (9)	KGD2 (2)	YBL078C (2)	YSW1 (5)	CDC27 (7)	YMR322C (7)
PYC1 (1)	YDR374 (9)	AGA2 (2)	QRI1 (2)	SPR28 (5)	DIT2 (7)	YOR391C (7)
SIP4 (1)	DMC1 (9)	YPT32 (2)	YNL013C (2)	SPS2 (5)	YKL050C (7)	SPS100 (6)
CAT2 (1)	HOP1 (9)	SPO16 (2)	APC4 (5)	YLL012W (5)	DIT1 (5)	
YOR100C (1)	IME2 (9)	YPR192W (2)	STU2 (5)	YLR277C (7)		
CAR1 (1)			PDS1 (9)	ORC3 (7)		
				YLL005C (7)		

Table 7.1: A representative set of genes were hand-picked for 7 temporal classes [Chu *et al.* 1998]. The number in the bracket beside the gene name is the cluster which the gene belongs to.

respectively. Those hand-picked genes with the temporal class of Early-Mid, Middle, Mid-Late, and Late are grouped into the different clusters but some of them are still in the same cluster. Therefore, more time points might be needed to sharpen these boundaries and reveal more classes or more clusters may be needed.

The genes with significant GO terms in the different clusters can be explored by FatiGO. For example, 44.62% of the genes in cluster 6 function as the GO term of nucleic acid binding. The function is significantly different from the rest of the genes with p-value less than $1e^{-5}$ (Table 7.2). In cluster 3, 57.86% of the genes are functionally annotated in GO term as structural constituent of ribosome. This percentage is clearly higher than the 3.11% observed for the distribution of this GO term in the rest of the genes (Figure 7.6). Similarly, the significant component and process annotations are summarized in Table 7.3 and Table 7.4 respectively. For example, 88.14% of the genes in cluster 3 are involved in metabolism process and the annotation is significantly different from the rest of the genes with p-value less than $1e^{-5}$ (Table 7.4). 53.33% of the genes in cluster 9 were annotated as the component of non-membrane-bound organelle and the annotation is significantly different from the rest of the genes with p-value less than 0.005 (Table 7.3).

Cluster	Function(s)	Percentage	p-values		
2	structural constituent of cytoskeleton	8.70%	0.00002	0.00194	0.00984
3	structural constituent of ribosome	57.86%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
	transferase activity	8.12%	0.00032	0.00666	0.03147
5	structural constituent of cytoskeleton	17.65%	$< 1e^{-5}$	0.00037	0.00187
6	nucleic acid binding	44.62%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
	helicase activity	12.31%	0.00011	0.00502	0.02548

Table 7.2: Significant function annotations in the different clusters. The p-values from left to right: unadjusted p-value, adjusted p-value of step-down minP and FDR [Benjamini and Hochberg 1995].

Cluster	Component(s)	Percentage	p-values		
3	ribonucleoprotein complex	55.21%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
	non-membrane-bound organelle	64.06%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
	eukaryotic 43S preinitiation complex	21.35%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
	eukaryotic 48S initiation complex	19.79%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
	intracellular organelle	87.50%	0.00002	0.00032	0.00173
	intracellular	97.92%	0.00031	0.00481	0.02599
4	proton-transporting ATP synthase complex	3.33%	$< 1e^{-5}$	0.00002	0.00013
	proton-transporting two-sector ATPase complex	3.33%	$< 1e^{-5}$	0.00002	0.00013
	external encapsulating structure	7.14%	$< 1e^{-5}$	0.00030	0.00146
5	immature spore	16.33%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
	external encapsulating structure	20.41%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
7	ubiquitin ligase complex	6.56%	$< 1e^{-5}$	0.00018	0.00099
8	external encapsulating structure	20.00%	$< 1e^{-5}$	$< 1e^{-5}$	0.00001
9	non-membrane-bound organelle	53.33%	$< 1e^{-5}$	0.00066	0.00354

Table 7.3: Significant component annotations in the different clusters. The p-values from left to right: unadjusted p-value, adjusted p-value of step-down minP and FDR [Benjamini and Hochberg 1995].

Cluster	Process(es)	Percentage	p-values		
2	regulation of gene expression, epignetic	8.04%	0.00024	0.00577	0.02179
3	metabolism	88.14%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
5	cell differentiation	48.21%	$< 1e^{-5}$	$< 1e^{-5}$	$< 1e^{-5}$
7	cell differentiation	11.71%	$< 1e^{-5}$	0.00005	0.00019

Table 7.4: Significant process annotations in the different clusters. The p-values from left to right: unadjusted p-value, adjusted p-value of step-down minP and FDR [Benjamini and Hochberg 1995].

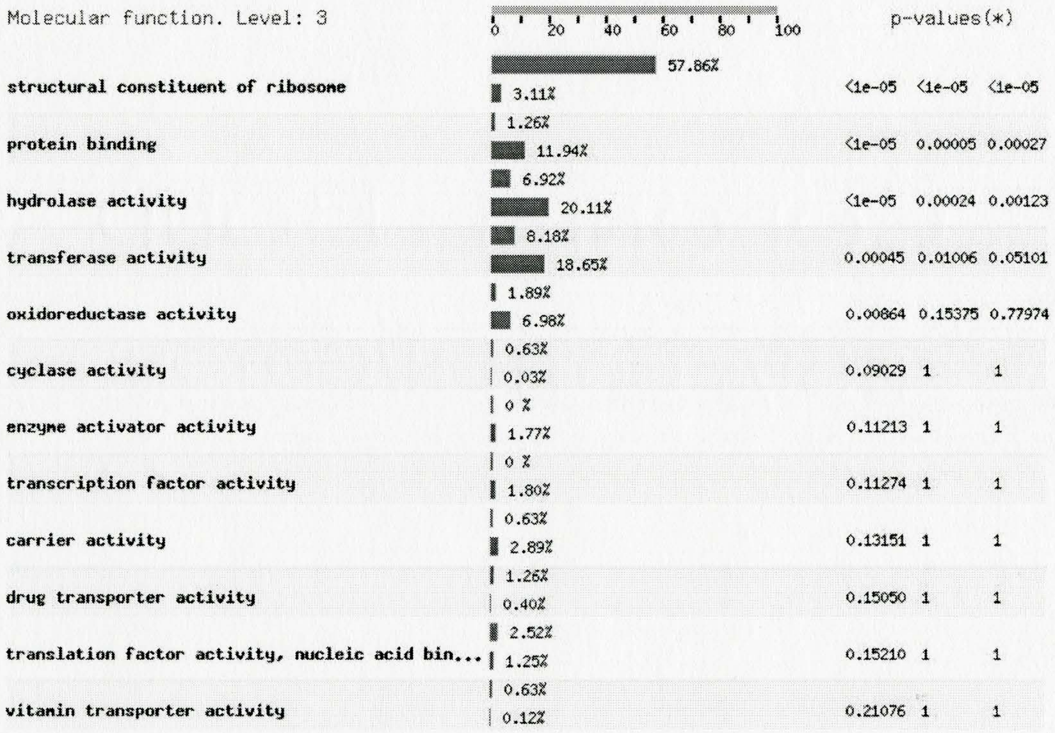


Figure 7.6: Exploration with FatiGO to show that the percentage of a GO term in Cluster 3 is different from the distribution of this term in the rest of the genes. The p-values from left to right: unadjusted p-value, adjusted p-value of step-down minP and FDR [Benjamini and Hochberg 1995].

In each cluster, there exist many genes with unknown functions. Because genes with related functions tend to be expressed in similar patterns, the possible roles and functions of these unknown genes can be inferred by the roles and functions of the well known genes in the same cluster. These hypothesis about the gene function can be verified by further studies.

In cluster analysis, usually the number of clusters must be decided in advance and arbitrarily selected. A statistical method (gap statistic) can be applied to find the number, as done here, especially when the underlying biological knowledge is unavailable.

The biological interpretation of gene clusters is the key step of the whole analysis. Because there exist many gene functional annotation databases, without a standard term to describe the gene's function(s), the interpretation of the clustering results might be misunderstood. The GO provides structured, controlled vocabularies (ontologies) that describe gene products in terms of their associated biological processes, cellular components and molecular functions in a species-independent manner. FatiGO is applied to compare two groups of genes and extract a list of GO terms distribution among the two groups which is significantly different by using multiple testing.

To validate the clustering results above, the whole data set without RMS filtering was analyzed using the same method above. The number of cluster was chosen to be 11 as identified by the gap statistic. The changing patterns of gene expression for these 11 clusters is very similar to what we found above (compare Figure 5.3 and Figure 7.7). Cluster 1, 2 and 3 (total of 3391 genes) from the complete data set contain those genes with smaller changes during sporulation process and the tendency is for constant gene expression in clusters. Analysis using the whole data set and 1148 filtered genes generally identify the same tendencies for genes either upregulated or downregulated

Cluster from 1148 filtered genes			Cluster from 6118 genes			Number of the common genes in two clusters
Cluster	1*	(50)	Cluster	8	(345)	44
Cluster	2	(166)	Cluster	8	(345)	73
Cluster	3	(225)	Cluster	10	(204)	125
Cluster	4*	(235)	Cluster	5	(491)	213
Cluster	5*	(90)	Cluster	9	(100)	89
Cluster	6*	(108)	Cluster	4	(412)	101
Cluster	7*	(176)	Cluster	7	(273)	154
Cluster	8*	(59)	Cluster	10	(204)	59
Cluster	9*	(39)	Cluster	11	(46)	39
			Cluster	1	(1173)	0
			Cluster	2	(708)	0
			Cluster	3	(1510)	0
			Cluster	6	(856)	0
Total		(1148)	Total		(6118)	

Table 7.5: Comparison of clustering results from the whole genes data set and the filtered genes data set. the number in the bracket is the number of genes within the cluster. The most genes within the cluster with asterisk are well separated into the different clusters even when the whole data set is used.

over entire time period, but of course with a different number of clusters. Comparison of clustering results from the whole genes data set and the filtered genes data set is summarized in Table 7.5. Clusters were matched by the shape of the mean profile and the number of genes in common for matched clusters was counted. There is reasonable consistency in cluster membership between the 1148 filtered genes data set and the all 6118 genes data set. For example, 90 genes from the filtered gene data set are grouped into cluster 5. When 6118 genes are used, 89 of 90 genes are still grouped together into cluster 9 from the whole data set. This means that these 89 genes might have a strong relationship. Clusters 1, 2, 3 and 6 of 6118 genes contains genes with smaller expression changes. These genes were removed when RMS filter criterion of 1.13 was applied.

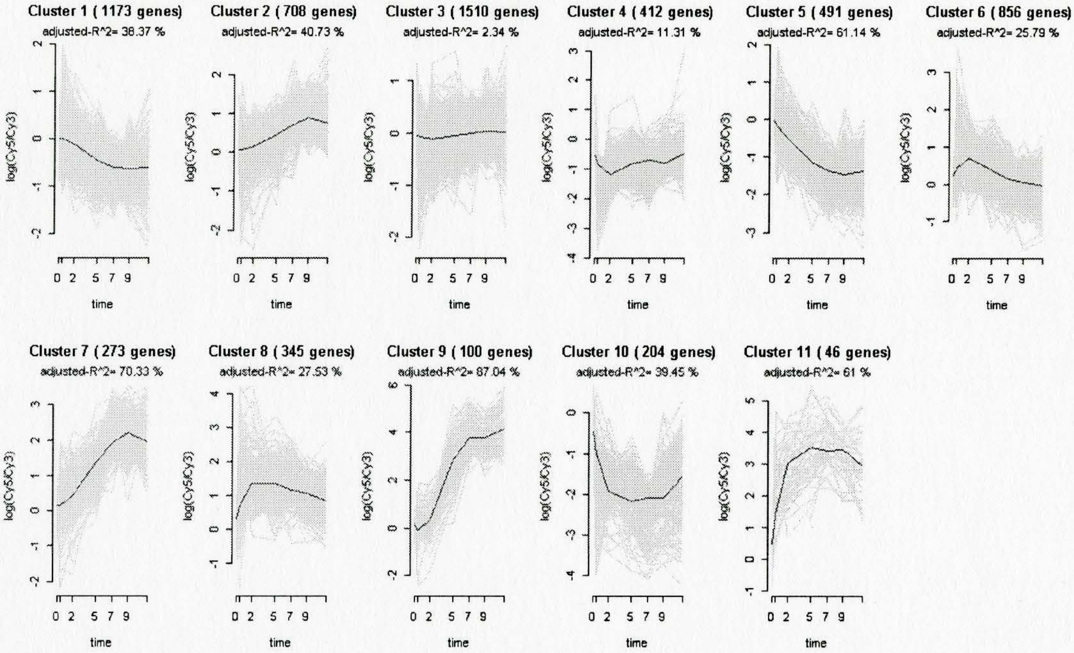


Figure 7.7: The gene patterns for 9 clusters are obtained from all of 6118 genes by K-means clustering algorithm.

Chapter 8

Conclusions

DNA microarray technology has now made it possible to measure simultaneously the gene expression levels for thousands of genes during certain biological processes (e.g. sporulation) or the response to changes in the environment (e.g. drug treatment) across samples.

The application of cluster analysis to gene expression data is based on the assumption that genes with similar expression profiles share similar functions or involve similar biological processes [Eisen *et al.* 1998; Gibbons and Roth 2002]. The co-expressed genes with unknown functions or poorly characterized genes in the same cluster can be predicted by genes with known functions or characters. One biological process may involve hundreds of genes. One gene may also function in many biological processes. The genes with similar expression patterns in the same cluster can help in understanding how the genes interact with changes in the environment/conditions. This is very useful in new drug discovery.

Cluster analysis has proven to be useful to group genes together with similar functions based on gene expression patterns under various conditions or across different tissue samples [Eisen *et al.* 1998] and has demonstrated genes with similar expression patterns contribute to common function and are likely co-regulated. However, clustering is an unsupervised method. This means that no previous knowledge of the number and characteristics of the clusters in the data is used in determining the clusters. There exist different techniques for identifying the number of clusters. A recent approach to identify the number of clusters is the use of gap statistic. For unsupervised cluster analysis, GO is an excellent biological knowledge database for gene function prediction and biological interpretation of clusters. The software FatiGo, a query tool, is used to integrate GO annotation into cluster analysis.

In this project, similarity patterns and significant GO annotations in each cluster are assessed. With our methodology, the biological knowledge is integrated into cluster analysis and makes the clustering results more meaningful. The gap statistic is used to determine the number of the clusters based on the nature of the data, and thus the number of clusters is not arbitrarily selected. The gene expression patterns are displayed by fitting a polynomial curve for each cluster and by displaying the heat maps for each cluster. The two types of displays provide complementary information since the heat map is a depiction of the state of expression, as upregulation, downregulation or no change relative to the control, at each time point for all genes in a cluster. The use of these methods in the project shows that they can be easily implemented and that they automate the microarray data analysis.

The final results depend on what clustering method and similarity metric is employed and what technique is used to find the "optimal" number of clusters in the given data set. However, if used in an iterative fashion, the procedures and the methodology used here are generally applicable and provide knowledge-driven cluster analysis for gene expression data.

Most clustering algorithms require that a gene reside within exactly one cluster. But genes usually play multiple roles in sporulation process. From the view of gene functional category, it is reasonable that some genes may overlap between categories. It means that the same genes might function differently at different stages or conditions (see Appendix A.1). Many genes can be seen in multiple categories. Therefore, it is reasonable to generate "fuzzy" clusters, or leave some genes unclustered [Chen *et al.* 2002]. Application of such methods to the present data set is an area for further investigation.

Usually a microarray data set contains a huge amount of gene expression values. It is necessary to draw attention to the time complexity of the clustering algorithms when using some clustering algorithms. K-means algorithm is relatively efficient. The time complexity is $O(tkn)$, where n is the number of objects, k is the number of clusters, and t is the number of iterations $(k, t \ll n)$. Comparing with other common used clustering algorithms, PAM has time complexity $O(k(n-k)^2)$. DIANA has worst-case time complexity $O(n^2 logn)$. The different versions of AGNES differ in how they compute cluster similarity. The most common versions of AGNES are single-link, complete-link and average-link clustering. The complexity of these algorithms is $O(n^2 logn)$. [Pantel 2003; Jain *et al.* 1999].

References

1. Al-Shahrour, F., Díaz-Uriarte, R., Dopazo, J. (2004) FatiGO: a web tool for finding significant associations of Gene Ontology terms with groups of genes. *Bioinformatics*, **20**, 578-580.

2. Al-Shahrour, F., Herrero, J., Mateos, Á., Santoyo, J., Díaz-Uriarte, R. and Dopazo, J. (2003) Using Gene Ontology on genome-scale studies to find significant associations of biologically relevant terms to group of genes. Neural Networks for Signal Processing XIII. IEEE Press.

3. Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000) Gene ontology: tool for the unification of biology. *Nature Genetics*, **25**, 25-29.

4. Baldi, P., Hatfield, G. W. (2002) DNA microarrays and gene expression from experiments to data analysis and modeling. Cambridge University Press.

5. Benjamini, Y., Hochberg, Y. (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal Royal Statistical Society*,

B. **57**, 289-300.

6. Benjamini, Y., Yekutieli, D. (2001) The control of the false discovery rate in multiple testing under dependency. *Annals of Statistics*, **29**, 1165-1188.

7. Bittner, M., Meltzer, P., Chen, Y., Jiang, Y., Seftor, E., Hendrix, M., Radmacher, M., Simon, R., Yakhini, Z., Ben-Dor, A., Sampas, N., Dougherty, E., Wang, E., Marincola, F., Gooden, C., Lueders, J., Glatfelter, A., Pollock, P., Carpten, J., Gillanders, E., Leja, D., Dietrich, K., Beaudry, C., Berens, M., Alberts, D., Sondak, V., Hayward, N. and Trent, J. (2000) Molecular classi.cation of cutaneous malignant melanoma by gene expression pro.ling. *Nature*, **406**, 536-540.

8. Calladine, C. R., Drew H. R. (1997) *Understanding DNA : the molecule and how it works.* Academic Press.

9. Chen, G., Jaradat, S. A., Banerjee, N., Tanaka, T. S., Ko, M. S. H., and Zhang, M. Q. (2002) Evaluation and Comparison of Clustering Algorithms in Analyzing ES Cell Gene Expression Data. *Statistic sinica*, **12**, 241-262.

10. Chu, S., DeRisi, J., Eisen, M., Mulholland, J., Botstein, D., Brown, P. O., Herskowitz, I. (1998) The Transcriptional Program of Sporulation in Budding Yeast. *Science*, **282**, 699-705

11. Cleveland, W. S. (1979) Robust locally weighted regression and smoothing scatterplots. *J. Amer. Statist. Assoc.* **74**, 829-836.

12. Conde, L., Mateos, A., Herrero, J. and Dopazo, J. (2002) Unsupervised reduction of the dimensionality followed by supervised learning with a perceptron improves

the classification of conditions in DNA microarray gene expression data. *Neural Networks for Signal Processing XII*, (IEEE Press) eds. Boulard, Adali, Bengio, Larsen, Douglas. 77-86.

13. Costa, I.G., Carvalho, F., Souto, M. (2004) Comparative analysis of clustering methods for gene expression time course data. *Genetics and Molecular Biology*, **27**, 4, 623-631.

14. Duggan, D. J., Bittner, M., Chen, Y., Meltzer, P., Trent, J. M. (1999) Expression profiling using cDNA microarrays. *Nature Genetics* **21**, 10-14.

15. Dudoit, S., Shaffer, J. P., Boldrick, J. C. (2003) Multiple Hypothesis Testing in Microarray Experiments. *Statistical Science*, **18**(1), 71C103.

16. Eisen, M. B., Spellman, P. T., Brown, P. O., Botstein D. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci.*, **95**(25), 14863-14868.

17. Everitt, B. (1974) *Cluster Analysis.* Heinemann, London.

18. Gibbons, F. D., Roth, F.P. (2002) Judging the Quality of Gene Expression-Based Clustering Methods Using Gene Annotation. *Genome Research* **12**(10), 1574-1581.

19. Goldsmith, C. H. (1974) Creation of Pure Error in Regression Problems. *Biometrics.* **30**(3), 561.

20. Hamilton, L. C. (1992) *Regression with graphics: a second course in applied statistics.* Duxbury Press.

21. Hartigan, J. (1975) *Clustering Algorithms.* John Wiley & Sons, New York.

22. Jain, A. K., Murty, M. N., and Flynn, P. J. (1999) Data clustering: a review. ACM Computing Surveys, **31**(3), 264C323.

23. Kaminski, N., Friedman N. (2002) Practical Approaches to analyzing Results of microarray Experiments. *Am. J. Respir. Cell Mol. Biol.*, **27**, 125-132.

24. Kaufman, L., Rousseeuw, P. J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis.* Wiley, New York.

25. Knudsen, S. (2004) *Guide To Analysis of DNA Microarray Data.* Wiley.

26. Leung, F. Y., Cavalieri, D. (2003) Fundamentals of cDNA microarray data analysis. *Trends in Genetics.*, **19**(11), 649-659.

27. Mateos, A., Herrero, J., Tamames, J., Dopazo, J. (2002) Supervised neural networks for clustering conditions in DNA array data after reducing noise by clustering gene expression profiles. In *Methods of Microarray Data Analysis II*, ed. Lin SM, Johnson KF, Kluwer Academic. Publ., pp. 91-103.

28. Milligan, G.W., Cooper, M.C. (1985) An examination of procedures for determining the number of clusters in a data set. *Psychomatrika*, **50**, 159-179.

29. Nguyen, D.V., Arpat, A.B., Wang N, Carroll RJ (2002) DNA Microarray Experiments: Biological and Technological Aspects. *Biometrics*, **58**, 701-717.

30. Pantel, P. (2003) Clustering by Committee. Ph.D. Dissertation. Department of Computing Science, University of Alberta.

31. Pevsner, J. (2003) *Bioinformatics and Functional Genomics.* Wiley.

32. Rousseeuw, P.J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Computat. Appl. Math.*, **20**, 53-65.

33. Saeed, A.I., Sharov, V., White, J., Li, J., Liang, W., Bhagabati, N., Braisted, J., Klapa, M., Currier, T., Thiagarajan, M., Sturn, A., Snuffin, M., Rezantsev, A., Popov, D., Ryltsov, A., Kostukovich, E., Borisovsky, I., Liu, Z., Vinsavich, A., Trush, V., Quackenbush, J. (2003) TM4: a free open-source system for microarray data management and analysis. *Biotechniques*, **34**(2), 374-378

34. Shannon, W.,1Culverhouse, R., Duncan, J. (2003) Analyzing microarray data using cluster analysis. *Pharmacogenomics* **4**(1), 41C51.

35. Sherlock, G. (2001) Analysis of large-scale gene expression data. *Briefings in Bioinformatics.* **2**, 350-362

36. Struyf, A., Hubert, M., Rousseeuw, P.(1996) Clustering in an Object-Oriented Environment, *Journal of Statistical Software*, **1**. http://www.jstatsoft.org/index.php?vol=1.

37. Tibshirani, R., Walter, G., Hastie, T. (2001) Estimating the number of clusters in a data set via the gap statistic. *J. R. Statist. Soc. B.*, **63**, 411-423.

38. Westfall, P.H., Young, S.S. (1993) *Resampling-based multiple testing: examples and methods for p-value adjustment.* John Wiley & Sons.

39. Yang, Y.H., Dudoit, S., Luu, P., Speed, T.P. (2002) Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Research*, **30**, 4e15.

Additional References

1. Bolshakova, N., Azuaje, F. (2005) Estimating the number of clusters in DNA microarray data. *Method of Information in Medicine*, (to appear) http://www.cs.tcd.ie/research_groups/mlg/kdp/Bolshakova_IMIA04.pdf (abstract)

2. Brazma, A., Vilo, J. (2000) Minireview: Gene expression data analysis. *Federation of European Biochemical Societies*, **480**, 17-24

3. Causton, H., Quackenbush, J., Brazma, A. (2003) *Microarray Gene Expression Data Analysis.* Blackwell Publishing

4. Cherry J. M., Adler C., Ball C., Chervitz S. A., Dwight S. S., Hester E. T., Jia Y., Juvik G., Roe T., Schroeder M., Weng S., Botstein D. (1998) SGD: Saccharomyces Genome Database, *Nucleic Acids Research*, **26**(1), 73-79.

5. Lægreid, A., Hvidsten, T. R., Midelfart, H., Komorowski, J., Sandvik, A. K. (2003) Predicting gene ontology biological process from temporal gene expression patterns. *Genome Res.*, **13**(5), 965-79.

6. McLachlan, G.J., Do, K., Ambroise, C. (2004) *Analyzing microarray gene expression data.* Wiley.

7. Mendenhall, W. (1968) *Introduction to Linear Models and The Design and Analysis of Experiments.* Duxbury Press

8. munich information center for protin sequences (mips), http://mips.gsf.de/

9. Saccharomyces Genome Database (SGD), http://www.yeastgenome.org/

10. Schena, M., (2003) *Microarray analysis.* Wiley.

11. Speed, T. (2003) *Statistical analysis of gene expression microarray data.* Chapman.

12. Yeung, K.Y., Haynor, D.R., Ruzzo, W.L. (2001) Validating Clustering for Gene Expression Data. *Bioinformatics*, **17**(4), 309-318.

Appendix A

A.1 The Gene Ontology (GO)

The Gene Ontology (GO) provides a consistent vocabulary to describe aspects of a gene product's biology. A gene product's biology is represented by three ontologies: molecular function, biological process and cellular component. The following table gives three ontologies (at level 1). The number in the bracket beside the GO term is the number of genes with the GO term at current level. (The table was from http://www.godatabase.org/cgi-bin/amigo/go.cgi)

```
 - all : all   ( 6456 )   Pie Chart for.
    - GO:0008150 : biological_process ( 6456 )
          + GO:0007610 : behavior ( 0 )
          + GO:0000004 : biological process unknown ( 1664 )
          + GO:0009987 : cellular process ( 4650 )
          + GO:0007275 : development ( 451 )
          + GO:0040007 : growth ( 110 )
          + GO:0007582 : physiological process ( 4741 )
          + GO:0050789 : regulation of biological process ( 572 )
          + GO:0016032 : viral life cycle ( 2 )
    - GO:0005575 : cellular_component ( 6439 )
          + GO:0005623 : cell ( 5408 )
```

```
      + GO:0008372 : cellular component unknown ( 1006 )
      + GO:0031012 : extracellular matrix ( 0 )
      + GO:0005576 : extracellular region ( 22 )
      + GO:0043226 : organelle ( 3921 )
      + GO:0043234 : protein complex ( 1299 )
      + GO:0019012 : virion ( 0 )
- GO:0003674 : molecular_function ( 6439 )
      + GO:0016209 : antioxidant activity ( 20 )
      + GO:0005488 : binding ( 1124 )
      + GO:0003824 : catalytic activity ( 1897 )
      + GO:0030188 : chaperone regulator activity ( 8 )
      + GO:0030234 : enzyme regulator activity ( 159 )
      + GO:0005554 : molecular function unknown ( 2304 )
      + GO:0003774 : motor activity ( 18 )
      + GO:0045735 : nutrient reservoir activity ( 0 )
      + GO:0031386 : protein tag ( 8 )
      + GO:0004871 : signal transducer activity ( 65 )
      + GO:0005198 : structural molecule activity ( 356 )
      + GO:0030528 : transcription regulator activity ( 324 )
      + GO:0045182 : translation regulator activity ( 58 )
      + GO:0005215 : transporter activity ( 425 )
      + GO:0030533 : triplet codon-amino acid adaptor activity ( 300 )
```

A.2 Function of the gap statistic

```
gapstat=function(X, K, B=20)
{
 #calculating the gap statistic
 #X = data matrix (normalized intensities ratios)
 #K = maximum of cluster (equal to or less than the
 #    number of rows of matrix X)
 #B = number of Monte Carlo replicates (greater than 1)

 Cluster=function(X, k, hc)
 {
  #run clustering algorithm
  #X = data matrix
  #k = the number of cluster in the current run
  #hc= clustering object returned by any hierarchical
  #    clustering algorithm
    if(k==1)
    {
       clusters=rep(1, length(X[,1]))
    }
    else
    {
       hc.clust=cutree(hc, k=k)
       centers=matrix(0, nrow=k, ncol=ncol(X))
       for(i in 1:k)
       {
          centers[i, ]=apply(as.matrix(X[hc.clust==i, ]), 2, mean)
          #centers[i, ]=apply(as.matrix(X[hc.clust==i, ]), 2, median)
       }
       clusters=kmeans(X, centers=centers, iter.max=100)$cluster
    }
    clusters
 }

 Refdist=function(x)
 {
  #uniform distribution as reference distribution
    runif(length(x), min=min(x), max=max(x))
 }
```

```
#set initial values
Wk=rep(0, K)        #pooled within-cluster sum of squares around
                    #the cluster mean
Wkb=matrix(0, K, B) #B copies of Wkb*
Gapk=rep(0, K)      #gap statistic
Sdk =rep(0, K)      #standard deviation of B of Wkb*
Sk  =rep(0, K)      #Sk=Sdk*sqrt(1+1/B)

hc=diana(X)         #executing DIANA clustering algorithm to
                    #create a hierarchical clustering object
                    #for computing the initial centers of K-means
#hc=hclust(dist(X), method="average")

for(k in 1:K)
{
   clusterX=Cluster(X, k, hc=hc)
   for(i in 1:k)
   {
      Nr=length(X[clusterX==i, 1])
      #Dr=sum((dist(X[clusterX==i, ]))^2)
      #Wk[k]=Wk[k]+Dr/Nr
      Wk[k]=Wk[k]+(Nr-1)*sum(diag(var(X[clusterX==i, ])))
   }
}

for(b in 1:B)
{
 #draw B Monte Carlo replicates using Method (a)
   Xstar=apply(X, 2, Refdist)

    # draw B Monte Carlo replicates using Method (b)
    # s=svd(X)      # decomposition
    # D=diag(s$d)
    # V=s$v         # X = U D V'
    # Xp=X%*%V      # X'= XV
    # Zp=apply(X', 2, Refdist)
    # Z=Zp%*%t(V)   # Z=Z't(V)
    # Xstar=Z

   hcstar=diana(Xstar)    #create a hierarchical clustering
```

```
                          #object for the initial centers of K-means
   #hcstar=hclust(dist(Xstar), method="average")

   for(k in 1:K)
   {
      clusterXstar=Cluster(Xstar, k, hc=hcstar)
      for(i in 1:k)
      {
         Nr=length(Xstar[clusterXstar==i, 1])
         #Dr=sum((dist(Xstar[clusterXstar==i, ]))^2)
         #Wkb[k, b]=Wkb[k, b]+Dr/Nr
         Wkb[k, b]=Wkb[k, b]+(Nr-1)*sum(diag(var(Xstar[clusterXstar==i, ])))
      }
   }
}

Khat=K
for(k in 1:K)
{
   Gapk[k]=mean(log(Wkb[k,]))-log(Wk[k])
   Lbar=mean(log(Wkb[k,]))
   Sdk[k]=sqrt((1/(B-1))*sum((log(Wkb[k,])-Lbar)^2))
   Sk[k]=sqrt(1+1/B)*Sdk[k]
}

for(k in 1:(K-1))
{
   if(Gapk[k]-(Gapk[k+1]-Sk[k+1])>=0)
   {
      Khat=k
      break
   }
}
par(mfrow=c(1, 2))
plot(1:K, Wk, xlab="Number Of Clusters k",
     ylab="Within sum squares Wk")
lines(1:K, Wk, lty=2)
points(Khat, Wk[Khat], pch=19)
plot(1:K, Gapk, xlab="Number Of Clusters k", ylab="Gap")
lines(1:K, Gapk, lty=2)
points(Khat, Gapk[Khat], pch=19)
```

```
 title("Gap statistic", outer=T, line=-2.5)
 list(Gapk=Gapk, Sk=Sk, Sdk=Sdk, Diff=Gapk[1:(K-1)]-(Gapk[2:K]-Sk[2:K]),
      Khat=Khat)
}
```

A.3 Fitting the polynomial curves

```
Main<-function(dat, K)
{
 #fit the polynomial curve
 #dat = data matrix (normalized intensities ratios)
 #K = the number of cluster
   if(K==1)
   {
      cluster.id=rep(1, length(dat[,1]))
   }
   else
   {
      hc.clust=cutree(diana(dat), k=K)
      centers=matrix(0, nrow=K, ncol=ncol(dat))
      for(i in 1:K)
      {
         centers[i, ]=apply(as.matrix(dat[hc.clust==i, ]), 2, mean)
      }
      cluster.id=kmeans(dat, centers=centers, iter.max=100)$cluster
    }
    clustObj=cbind(dat, cluster.id)

    par(mfrow=c(ceiling(K/5), 5))
    for(j in 1:K)
    {
       Cluster.Plot(j, clustObj)
       Fit.Curve(j, line=T, power=4, clustObj)
    }
}

Cluster.Plot<-function(i, dat)
{
  #draw the scater plot, time vs log ratio
  #i = i_th cluster
  #dat = intensity matrix with clustering vector
   r=dat
   maxy=max(r[, 1:7]) # compute y axis range
   miny=min(r[, 1:7])
   r=as.matrix(r)
   x=c(0, 0.5, 2, 5, 7, 9, 11.5)      #the time points when the
                                      #samples were taken
   num=GenesInCluster(r)              #get the number of the
                                      #genes in each cluster
   plot(0, xlim=c(0, 11.5), ylim=c(miny, maxy), main=paste("Cluster",
        i, "(", num[i], "genes)"), type="n", xlab="time",
        ylab="log(Cy5/Cy3)", axes=F)
   axis(1, at=x, labels=c("0", "0.5", "2", "5", "7", "9", "11.5"))
   axis(2)

   for(j in 1:dim(r)[1])
   {
      if(r[j,8]==i)
      {
         lines(x, r[j,1:7], col="light grey")
      }
   }
}

GenesInCluster<-function(dat)
{
 #Count the total number of genes within a cluster.
 #dat = intensity matrix with clustering vector
   r=dat
   n=max(r[, 8])   #total number of clusters
   NumGenes=matrix(0, nrow=1, ncol=n)
   for(i in 1:n)
   {
      NumGenes[i]=sum(r[, 8]==i)
   }
   NumGenes     #the total number of gene in each cluster
}

Fit.Curve<-function(i, line=T, power, dat)
{
 #fit a curve to the data
 #i = i_th clusters
 #line = T draw the fitted line;
 #power = the highest power of polynomial curve(<=4)
 #dat = intensity matrix with clustering vector
```

```
   r=dat
   n=max(r[, 8])   # total number of clusters
   num=GenesInCluster(r)
   X=matrix(0, nrow=7, ncol=power+1)    # design matrix
   X[, 1]=c(1, 1, 1, 1, 1, 1, 1)
   X[, 2]=x1=c(0, 0.5, 2, 5, 7, 9, 11.5)
   for(j in 3:(power+1))                # construct design matrix
     X[, j]=X[, 2]^(j-1)
   #row binding for mutlple Y's
   tmp=X
   if(num[i]>1)
   {
      j=1
      for(j in 1:(num[i]-1))
         X=rbind(X, tmp)
   }

   Y=matrix(t(r[r[,8]==i,1:7]), nrow=7*num[i], ncol=1)  #construct Y matrix
   bHat=solve(t(X)%*%X)%*%t(X)%*%Y          #estimate the coefficients of
                                            #the polynomial curve function
   yHat=X%*%bHat                            #fitted Y
   SSE=round(t(Y-yHat)%*%(Y-yHat), digits=3) #SSE=Sum Square of (Error)
                                            #Residual
   df=length(Y)-qr(X)$rank                  #df=Dgree of Freedom
   bHat=round(bHat, digits=5)               #round digits for displaying
   SSM=sum(yHat^2)                          #SSM=Sum of Squares Model
   MSS=(sum(Y))^2/length(Y)                 #MSS=Sum of Squares Mean
   SST=sum(Y^2)                             #SST=Sum of Squares Total
   R=round(SSM/SST*100, digits=2)
   Radj=round((SSM-MSS)/(SST-MSS)*100, digits=2)

   yHat=yHat[1:7]
   if(line==T)
   {
      lines(x1, yHat, col="red", lwd=1)
   }
   c(SSE, df, bHat)
}
```

A.4 Normalization of the array at hour 0

```
t0r=t0red-t0redbkg       #background correcting for Cy5 (red)
t0g=t0green-t0greenbkg   #background correcting for Cy3 (green)
A0=1/2*log(t0r*t0g, base=2)      #compute A for hour 0
M0=log(t0r/t0g, base=2)          #compute M for hour 0

loess0=loess(M0~A0, span=0.88)   #LOWESS fitting
newM0 =M0-predict(loess0, A0)
newt0r=2^(A0+newM0/2)
newt0g=2^(A0-newM0/2)
newt0 =newt0r/newt0g
lognewt0=log(newt0, base=2)      #normalized log ratio of hour 0

#MA plot (before normalization)
predict0=predict(loess0)
plot(A0, M0, xlab="A (Average log intensity)",
     ylab="M (log ratio)", col="light grey")
lines(A0[order(A0)], predict0[order(A0)], col="red", lwd=2)

#MA plot (after normalization)
plot(A0, newM0, xlab="A (Average log intensity)",
     ylab="M (log ratio)", col="light grey")
lines(A0[order(A0)], predict(loess(newM0~A0, span=span,
      degree=2))[order(A0)], col="red", lwd=2)
```

A.5 Improved K-means clustering method

```
Cluster=function(X, k, hc)
{
 #computer the initial centers for K-means clustering algorithm
 #X = data matrix
 #k = the number of cluster in the current run
 #hc = clustering result from another clustering algorithm (diana)
    if(k==1)
    {
       clusters=rep(1, length(X[,1]))
    }
    else
    {
       hc.clust=cutree(hc, k=k)
       centers=matrix(0, nrow=k, ncol=ncol(X))
       for(i in 1:k)
       {
          centers[i, ]=apply(as.matrix(X[hc.clust==i, ]), 2, mean)
       }
       clusters=kmeans(X, centers=centers, iter.max=100)$cluster
    }
    clusters
}
```

A.6 Attaching functional annotation to the expression data matrix

```
#annotation.csv: functional annotation database
annotation=read.csv("annotation.csv", header=T, sep=",")
gene_anno=combine(as.vector(dat[, 1]),as.vector(annotation[, 3]),
                  as.vector(annotation[, 4]))
km=Cluster(dat[, 2:8], 9, iter.max=200)   #run K-means clustering
cl.anno1=gene_funcs[km$cluster==1, ]      #annotation for cluster 1

spaces=function(n)
{ #generate n spaces
   spaces_str=""
   if(n==0)
   {
      return("")
   }
   else
   {
      for(i in 1:n)
      spaces_str=paste(spaces_str, " ", sep="")
      return(spaces_str)
   }
}

combine=function(c1, c2, c3)
{ #combine the description columns for annotation
   n=length(c1)
   max_len_c1=max(nchar(c1))
   max_len_c2=max(nchar(c2))
   results=rep("", n)
   for(i in 1:n)
   {
      results[i]=paste(c1[i], spaces(1+max_len_c1-nchar(c1[i])), c2[i],
                       spaces(1+max_len_c2-nchar(c2[i])), c3[i], sep="")
   }
   as.matrix(results)
}
```

A.7 Gene functions and expressions in Cluster 1

ORFs	DESCRIPTION	TEMPORAL CLASS	t0	t05	t2	t5	t7	t9	t115
YAL062W	NADP-glutamate dehydrogenase	Metabolic	0.38200	3.326421	3.7190581	2.53136567	-0.01819543	0.16155006	1.007761845
YAR007C	replication factor A, 69 kD subunit	Metabolic	0.13700	2.102698	2.9156549	1.96572862	1.72112550	0.74138717	0.675133318
YAL054C	acetyl-CoA synthetase	Metabolic	0.54000	3.304114	2.6214284	3.03557597	2.17190924	0.16044949	0.362131916
YBL043W	unknown	Metabolic	0.34400	3.116136	2.0841143	1.13345508	1.54122048	0.95783122	0.395195596
YBR088C	DNA polymerase processivity factor	Metabolic	0.06980	1.434064	2.8083567	2.02531107	1.46927755	0.66215004	0.476037472
YCR005c	peroxisomal citrate synthase	Early I	0.07500	1.938328	2.5223558	2.15087139	1.81361794	0.46151472	0.712103734
YDR180W	unknown; binds chromosomes	Early I	-0.17900	1.483697	1.9426578	1.92862410	1.72974487	0.19281574	0.091994342
YDR256C	catalase A	Metabolic	0.37200	4.183543	2.2564419	1.78267455	0.88164964	1.37165634	2.616370225
YER024w	similar to Yat1p	Metabolic	0.24900	3.586683	3.0521148	2.21359124	1.95802901	0.82246925	2.088450390
YER055c	ATP phosphoribosyltransferase	Metabolic	-0.19200	3.198279	1.3477982	0.47815272	1.22822047	0.96880340	0.800948333
YER069w	acetylglutamate kinase and	Metabolic	0.02940	2.701990	1.5209777	0.24102406	0.52626751	1.17740074	0.541036487
YER091c	homocysteine methyltransferase	Metabolic	0.32600	3.060462	1.6587843	0.20363083	-1.54264751	-0.39530805	0.297327475
YFR030W	sulfite reductase subunit	Metabolic	0.09840	3.583718	1.2629609	1.02381834	1.12321685	0.49516748	1.188728695
YGL062W	pyruvate carboxylase 1	Metabolic	0.05480	2.692410	2.3323696	1.59561554	0.52192057	-0.03650271	-0.268792926
YGR067C	unknown	Metabolic	0.17600	3.089401	2.2833270	0.09437134	-0.52196861	-0.53089387	-0.532048341
YGR239C	unknown	Metabolic	-0.00596	2.538358	1.9969214	0.86273101	0.73384813	0.38484594	0.221866599
YHL024W	similar to RNA-binding proteins in the N-terminal	Early I	-0.14500	2.336907	2.8054093	3.88017028	2.53535073	1.24848976	0.482582230
YHL030W	unknown	Metabolic	0.01280	2.588238	1.5700469	1.49137128	1.71886875	1.20189114	0.208800151
YHR053C	metallothionein	Metabolic	0.16500	3.243503	0.1176645	-0.50892354	0.57635691	0.93394418	0.165432820
YIR029W	allantoicase	Metabolic	-0.10500	2.857225	3.2249737	0.30970174	0.44812188	0.17354567	-0.089710790
YIR042C	unknown	Metabolic	0.45500	2.678291	2.0635724	0.10641650	1.19881582	1.29477715	0.830075937
YJL045W	similar to succinate dehydrogenase flavoprotein	Early I	0.46600	2.394626	1.7968528	1.98767594	0.71093478	0.30408197	1.394578570
YJL089W	transcription factor	Metabolic	0.08700	2.218829	1.8568243	1.62580248	0.95357346	0.69010428	0.327737182
YJL153C	peripheral membrane protein	Metabolic	-0.13200	3.052649	1.4919460	2.63355520	1.96334053	0.05300374	-0.754258826
YJR016C	dihydroxyacid dehydratase	Metabolic	0.16900	2.515357	2.2090341	1.26996153	0.39429380	0.76641336	0.676762572
YJR109C	carbamyl phosphate synthetase	Metabolic	0.07920	3.355053	1.9639593	-0.44923519	0.01268921	0.50984512	0.382585405
YJR152W	allantoate permease	Metabolic	0.17700	4.497459	3.8287357	0.14270701	1.02525857	1.01734777	0.960833692
YKL120W	similar to members of the mitochondrial carrier	Early I	-0.18900	3.140462	3.3673026	2.00583936	1.89924944	1.82446726	1.855881599
YKR009C	peroxisomal beta-oxidation protein	Metabolic	0.30400	3.107466	2.1058788	2.48214293	1.22746815	0.46803182	1.943894766
YKR033C	similar to Gat1p	Metabolic	-0.00775	3.743488	3.3391670	0.81222084	2.14721152	0.80175115	1.205533092
YKR034W	transcription factor	Metabolic	0.07280	3.469603	3.1731000	1.41040927	1.98635474	1.31322723	1.757846515
YKR071C	unknown		0.02550	1.589268	2.1879982	1.49371295	1.60262392	0.63501558	0.527659494
YLL027W	unknown	Early I	0.00726	2.561277	2.3730312	1.50725442	2.05876842	1.30740362	1.045830299
YLR303W	O-acetylhomoserine sulfhydrylase	Metabolic	-0.23400	3.985322	2.0400287	-0.70459650	-0.76486148	-0.73849681	0.355987391
YLR304C	aconitase	Metabolic	-0.03860	3.125141	2.4900813	1.97124101	2.26258619	0.05445716	-0.032563011
YLR438W	ornithine aminotransferase	Metabolic	0.11600	3.617980	3.1980299	2.79402836	2.11287886	1.77788175	1.343301386
YML042W	carnitine O-acetyltransferase	Metabolic	-0.05100	3.956638	3.0879112	2.90888846	2.34343943	0.73355699	1.426292294
YMR018W	similar to Pex5p/Pas10p (GB:Z49211)	Metabolic	-0.15700	3.265122	0.8828237	0.62903096	0.44138208	0.26832851	-0.171832079
YMR095C	unknown; induced in stationary phase	Metabolic	0.08120	2.532611	1.1959349	0.80948193	0.83728011	0.88661528	1.216433371
YMR147W	unknown	Early I	0.24400	2.435344	2.1360096	2.06002633	2.31926829	0.91901498	1.221490206
YNL117W	malate synthase	Metabolic	-0.04510	3.191729	2.0716376	0.62285178	0.84208662	-1.73277463	-1.724330175
YNL142W	ammonia permease	Metabolic	-0.28800	4.246421	4.6267008	1.52893311	-0.32181054	-0.45385929	-0.003502410
YNL202W	peroxisomal 2,4-dienoyl-CoA reductase	Early I	0.00951	2.225263	1.5619210	0.78045778	0.92480388	0.85385112	0.942031801
YOR100C	similar to members of the mitochondrial carrier	Metabolic	-0.04490	3.277857	3.0613603	2.18075087	1.87482695	0.76445919	0.616558580
YOL125W	unknown	Metabolic	0.16300	2.407919	2.4954927	0.98843164	1.02493439	0.74265505	-0.003323616
YOR225W	unknown	Metabolic	-0.03040	2.141477	2.1323865	1.10541858	0.91440029	0.72194163	0.464522303
YOR375C	glutamate dehydrogenase	Metabolic	-0.07000	3.390150	4.3347712	2.04731640	0.68683844	-0.77690990	-0.396953079
YPL111W	arginase	Metabolic	-0.07750	2.914627	2.9171662	3.09481529	1.82411487	1.07643066	0.692528757
YPR002W	similar to Bacillus subtilis MMGE protein	Metabolic	-0.19600	2.754872	2.4513502	2.11973421	0.98762691	-0.15117276	-1.057753376
YPR006C	isocitrate lyase, nonfunctional	Metabolic	-0.29000	3.318945	2.6714266	2.44014164	2.18508592	0.36416000	-1.061381115

A.8 Agglomerative hierarchical clustering algorithm

The agglomerative hierarchical clustering algorithm builds the hierarchy by merging. The objects initially belong to a list of singleton sets $S_1, \ldots, S_n$. Then a distance function is used to find the pair of sets $\{S_i, S_j\}$ from the list that is the shortest to merge. Once merged, S_i and S_j are removed from the list of sets and replaced with $S_i \cup S_j$. This process iterates until all objects are in a single group. Three variants of agglomerative hierarchical clustering algorithms may use three distance functions which measure the distance between two clusters. Complete linkage, average linkage, and single linkage methods use maximum, average, and minimum distances between the members of two clusters, respectively [Chen *et al.* 2002].

A.9 DIANA (Divisive Analysis) algorithm

DIANA (Divisive Analysis) [Kaufman 1990] is a divisive hierarchical clustering algorithm. DIANA construct the hierarchy by initially spliting the largest cluster containing all objects into clusters with only single object. The details of the DIANA algorithm is:

- Select a cluster C with the highest diameter

$$\text{diameter}(C) = \max_{i,j \in C} d(i, j),$$

  where $d(i, j)$ is the distance between two objects i and j.

- Find the object $i \in C$ with the highest average dissimilarity

$$\frac{1}{\|C\| - 1} \sum_{j \in C, j \neq i} d(i, j)$$

  to all other objects in the cluster C, then construct a new cluster C_1 with the object i such that $C_0 = Ci$ and $C_1 = \{i\}$.

- For each object $i \in C_0 (i \notin C_1)$, compute the difference

$$D_i = \frac{1}{\|C_0\| - 1} \sum_{j \in C_0, j \neq i} d(i, j) - \frac{1}{\|C_1\|} \sum_{j \in C_1} d(i, j)$$

- Find the object k with the largest difference

$$D_K = \max_{i \in C_0} \{D_i\}$$

- If $D_k > 0$, move the object k into C_1 from C_0 and repeat steps 3 and 4. Otherwise stop processing and the cluster C is split into two smaller clusters C_0 and C_1.

- If there exists a cluster C with the number of objects $\|C\| > 1$, then goto step 1 until all clusters contains only 1 object