Coherent Gene Expression Pattern Finding Using Clustering Approaches

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Rosy Das
Registration No. 001 of 2009

School of Engineering
Department of Computer Science and Engineering
Tezpur University
April 2010

Abstract

Analysis of gene expression data is an important research field in DNA microarray research. Data mining techniques have proven to be useful in understanding gene function, gene regulation, cellular processes and subtypes of cells. Most data mining algorithms developed for gene expression data deal with the problem of clustering. The purpose of this thesis is to study different clustering approaches for gene expression data. Our first contribution is a dissimilarity measure (DBK) which retains the regulation information and is robust to outliers. We have developed a graph-based clustering algorithm (GCA) for gene expression data. Its main idea is that, inter-cluster genes have more repulsion among them than intra-cluster genes. In particular, at any given moment, genes are clustered based on a repulsion factor which is based on the genes that are yet to be assigned a cluster. This consideration leads to an objective function that is used to find the cluster parameter that optimizes this objective function. Comparison of GCA with competitive algorithms over different real world data sets shows the superiority of our approach. We have also developed a nearest neighbor based clustering algorithm which incorporates frequent itemset mining (FINN). The output of the frequent itemset mining phase is fed as input to the nearest neighbor clustering for detection of clusters. The process is iterated over multiple passes. After each pass, the dataset is pruned by not considering the genes that have already been assigned clusters. Experimental evaluation shows the method is capable in finding finer clustering of the dataset. This thesis also includes a density based clustering algorithm (DGC) which uses the regulation information and the order preserving property of gene expression profiles to cluster genes into high density regions separated by sparse density regions. The proposed algorithm has been validated on several real-life data sets and found to perform well in comparison to similar algorithms. This thesis also incorporates an incremental version of the DGC algorithm (incDGC). Experimental results on six real world gene expression datasets demonstrate that incDGC can cluster the data in an efficient manner while at the same time obtain the same result as when DGC is applied to the whole updated database. All clustering algorithms have been validated using various statistical measures to show their effectiveness

over biological data.

Keywords — Clustering, Gene expression, coherent pattern, co-expressed gene, proximity measure, graph based clustering, frequent itemset mining, shared nearest neighbor, density based clustering, incremental clustering

Tezpur University

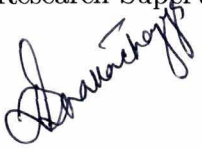
Certificate

This is to certify that the thesis entitled "Coherent Gene Expression Pattern Finding Using Clustering Approaches" submitted to the Tezpur University in the Department of Computer Science and Engineering under the School of Engineering in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Computer Science is a record of research work carried out by Ms. Rosy Das under my personal supervision and guidance.

All helps received by her from various sources have been duly acknowledged.

No part of this thesis has been reproduced elsewhere for award of any other degree.

Signature of Research Supervisor



(Dhruba Kumar Bhattacharyya)
Designation: Professor
School: Engineering
Department: Computer Science and Engineering

University of Colorado at Colorado Springs

Department of Computer Science
1420 Austin Bluffs Parkway
P.O. Box 7150
Colorado Springs, Colorado 80933-7150
(719) 262-3325
(719) 262-3369 Fax

January 11, 2010

Certificate of the Joint Research Supervisor

This is to certify that the thesis entitled "Coherent Gene Expression Pattern Finding Using Clustering Approaches" submitted to *Tezpur University* in the *Department of Computer Science & Engineering* under the *School of Engineering* in partial fulfillment for the award of the degree of the **Doctor of Philosophy in Computer Science** is a record of research work carried out by **Ms. Rosy Das** under my supervision and guidance.

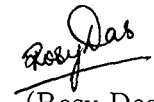
All helps received by her from various sources have been duly acknowledged.

No part of this thesis has been reproduced elsewhere for award of any other degree.

J. K. Kalita
(Research Supervisor)

Declaration

I, Rosy Das, hereby declare that the thesis entitled "*Coherent Gene Expression Pattern Finding Using Clustering Approaches*" submitted to the Department of Computer Science and Engineering under the School of Engineering, Tezpur University, in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy is based on bona fide work carried out by me. The results embodied in this thesis have not been submitted in part or in full, to any other university or institute for award of any degree or diploma.



(Rosy Das)

Acknowledgements

First, my greatest gratitude goes to Professor Dhruba Kumar Bhattacharyya, my supervisor, for his devoted guidance, advice, support and endless patience throughout the course of this research. With his excellent research experience, he steered me to research on clustering gene expression data. This thesis would not have been finished without his supervision and meticulous attention to details.

I want to express my deep gratitude to my thesis co-supervisor, Professor Jugal Kumar Kalita for his constant support. He is such a wonderful advisor, mentor, and motivator. Throughout my research, he gave me countless constructive comments and insightful suggestions.

I owe a debt of gratitude to Mr. Sauravjyoti Sarmah for his suggestions and fruitful discussions.

I am also very thankful to the rest of my thesis guidance committee, including Prof. Malay Ananda Dutta and Dr. Shyamanta M. Hazarika. Their advice and suggestions have been very helpful.

I would like to express my sincerest appreciation to all my colleagues and especially Dr. Utpal Sharma for his help. I am grateful to all the technical and non-technical members of the Department for their support.

My deep gratitude goes to my family members specially my father and my friends. I can never express my thanks enough for their endless love, support and understanding.

Tezpur University

Certificate

This is to certify that the thesis entitled "Coherent Gene Expression Pattern Finding Using Clustering Approaches" submitted by Ms. Rosy Das to Tezpur University in the Department of Computer Science and Engineering under the School of Engineering in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Computer Science has been examined by us on 12/8/11 and found to be satisfactory.

The Committee recommends for award of the degree of Doctor of Philosophy.

Signature of

Principal Supervisor

External Examiner

Date:

Contents

List of Tables

List of Figures

Notations Used in this Thesis

R	:	Set of real numbers.
$d(x, y)$	:	distance between x and y, x and y are elements of set Y.
$S_{x,y}$	:	similarity between objects x and y.
$min_{S_{x,y}}$	:	Minimum similarity.
$max_{S_{x,y}}$	:	Maximum similarity.
SMC	:	Simple Matching Coefficient.
J	:	Jaccard coefficient.
n	:	Number of dimensions.
EJ	:	Extended Jaccard coefficient.
C_i	:	i^{th} cluster.
DBK	:	Proposed dissimilarity measure.
NoC	:	number of clusters.
T_G	:	Gene Transaction database.
s	:	Support count.
MFIS	:	Maximal frequent itemset.
C_r	:	Set of core genes consisting of set of MFISs.
$\varepsilon_{i,j}$	:	Expression value of gene g_i at condition t_j.
$\xi_{i,j}$	:	Discretized value of gene g_i at condition t_j.
$\wp_{g_i}$	:	Regulation pattern of g_i.
MMRP	:	Maximal Matching Regulation Pattern.
$\wp'_{g_i}$	:	MMRP of g_i.
D_G	:	Gene database.
G^*	:	Set of all genes in D_G.
T^*	:	Set of all Conditions in D_G.
G	:	Total number of Genes.
T	:	Total number of Conditions.
D_I	:	Incremental database.
y	:	Total number of genes in the incremental dataset, D_I.
D_{upd}	:	Gene database updated incrementally.
D_{upd}	:	$D_G \cup D_I$.

Chapter 1

Introduction

Microarrays provide an extremely powerful way to analyze gene expression. Using a microarray, it is possible to examine the expression levels of thousands of genes across different developmental stages, clinical conditions or time points. It helps in understanding gene functions, biological processes, gene networks, effects of medical treatments, etc. Microarrays can be classified into two general types:

(i) cDNA arrays which consist of cDNA copies of mRNA spotted onto a glass slide and

(ii) oligo arrays which consist of strands of oligonucleotides either spotted onto a glass slide or lithographed onto a solid surface.

A typical microarray experiment consists of extracting RNA from the cells or the tissues being examined, converting the RNA to cDNA, labeling the cDNA with fluorescent dyes and allowing the labeled cDNA to hybridize with the material (cDNA or oligonucleotide) on the microarray slide. The control and subject RNAs are synthesized with different fluorescent dyes and mixed on the same slide. Then an image of the surface of the hybridized array (chip or microarray slide) is produced by scanning the chip to read the signal intensity that is emitted from the fluorescent dye of the heteroduplexes on the array where the target has bound to the probe. Raw data is obtained from this step. Next normalization and standardization steps are performed to clean and filter the data. Finally the real-valued gene expression data is obtained in the form of a matrix where the rows refer to the genes and the columns represent the conditions. The next step

is to use data mining techniques (such as clustering and association rule mining) to extract the hidden information in this data Finally validation is performed to check if the result obtained is good from a biological point of view

Data mining [HK04] is the technique of analyzing datasets (often large) in order to extract implicit, previously unknown and potentially useful information that might otherwise remain unknown

Data mining techniques are useful in microarray analysis because

- Data volumes are too large for traditional analysis methods
- High dimensionality
- Only a small portion of data is analyzed
- Decision support process becomes more complex

1.1 Cluster Analysis

Cluster analysis is an important technique in data mining, where the knowledge about the distribution of the observed data mfile ///home/rosy/thesisay not be available apriori Clustering is a data mining technique used to place data elements into related groups without advance knowledge of group definitions Clustering methods divide the data according to inherent classes present in it and are used in different scientific disciplines and engineering applications In recent years, clustering methods have been used extensively in analyzing biological data, especially from DNA microarrays measurements Among the different data mining techniques used in the analysis of gene expression data, clustering is an important technique that reveals natural structures and identifies interesting patterns in the underlying data A key step is the identification of a group of genes that manifest similar expression patterns over several conditions into clusters, thus revealing relations among genes and their functions A cluster of genes can be defined as a set of biologically relevant genes which are similar based on a proximity measure Intra-cluster genes are similar while inter-cluster genes are dissimilar The basic steps to develop a clustering algorithm can be summarized as follows

1 Feature selection This process identifies the most effective subset of the original features to use in clustering Irrelevant and redundant genes or conditions are not considered for future analysis

2 Clustering process This step refers to the application of a clustering algorithm to generate a good clustering scheme that fits the data set A clustering algorithm uses a proximity measure and a search method to find the optimal or sub-optimal groupings in the dataset according to some clustering criterion

   A proximity measure quantifies the similarity (or dissimilarity) of two data points

   The clustering criterion is based on the working definition of a cluster and/or an expected distribution of underlying data in specific application domain

3 Cluster validation Cluster validation is the assessment of a clustering scheme Typically, validation indices are defined to assess the quality of clusters

1.2 Criteria for Evaluating Clustering Algorithms

It is desirable that optimal algorithms for analysis of gene expression data satisfy the following properties [HK04]

1 Scalability and efficiency Algorithms should be efficient and scalable considering the large amount of data to be handled

2 Irregular shape Algorithms should be able to identify a dense set of objects which may be organized in irregular non-spherical shapes, including those with lacunae or concave sections and nested shapes, as a cluster

3 Robustness Clustering algorithms should be robust to noise and outliers

4 Order insensitivity Algorithms should be independent of data order

5 Cluster number The number of clusters in the data set should be determined by the algorithm itself and should not be an user input

6 Parameter estimation The algorithms should be able to estimate any parameters required by the algorithm from the dataset itself

7 Dimensionality Algorithms need the ability to handle data with high dimensionality or the ability to find clusters in subspaces of the original space

8 Stability The clustering result should remain the same for different runs of the algorithm

9 Incrementability Algorithms should be able to incrementally handle the addition of new data or the deletion of old data instead of re-running the algorithms on the entire new data set

10 Interpretability The clustering results of the algorithms need to be interpretable That is, clustering may need to be tied up with specific biological interpretations and applications

1.3 Co-expressed Genes and Coherent Patterns

Genes that have similar expression profiles are known as co-expressed genes A *coherent expression pattern* represents the common trend in expression levels for a group of co-expressed genes Furthermore, co-expressed genes in the same cluster are likely to be involved in the same cellular processes, and a strong correlation of expression patterns between those genes indicates co-regulation In practice, co-expressed genes may belong to the same or similar functional categories indicating co-regulated families [THC+99] Coherent gene expression patterns may characterize important cellular processes and may provide a foundation for understanding the regulation mechanism in the cells [SSI+98] According to [ABN+99], [ESBB98], [IER+99] and [JPZ03c] *there is usually a hierarchy of co-expressed genes and coherent expression patterns in a typical gene expression data* The co-expressed genes at the higher level have a "rough" coherent expression pattern while those at the lower levels have "finer" coherent expression patterns The

interpretation of co-expressed genes and coherent expression patterns depends mainly on the domain knowledge. Some challenges during gene expression data analysis are given below:

- Gene expression data consists of thousands of genes. However, only a subset of those genes may actually participate in the formation of coherent patterns.
- What level of "coherence" (roughness and fineness of the gene patterns) is required is dependent on the biologists, i.e., whether a group of genes should be further sub-divided into finer patterns depends on domain knowledge. By finer patterns we mean highly coherent patterns.
- It is ideal if biologists browse the rough patterns and decompose the patterns of interest to them into finer patterns.

Gene clustering is usually the first step in uncovering regulatory elements in transcriptional regulatory networks [ABN+99], [THC+99] as well.

1.4 Gene Expression Clustering

Clustering identifies genes with similar expression profiles (*co-expressed genes*). To identify co-expressed genes and coherent expression patterns, different clustering algorithms have been used. These include k-means [THC+99], SOM (Self Organizing Map) [TSM+99], QT Clustering [HKY99], Hierarchical clustering approaches [ESBB98], DHC [JPZ03a], CAST [BDSY99] and CLICK [SS00] Gene expression clustering algorithms are broadly divided into Partitional, Hierarchical, Density-based, Graph-based and Model-based approaches. The purpose of this thesis is to study different clustering approaches for gene expression data. Clustering algorithms use a proximity measure to group similar genes into the same cluster. Different proximity measures give different results. The choice of proximity measure depends on the application as well as the clustering approach being used. A study of proximity measures is included in this thesis. we also propose an effective proximity measure that is been found capable of detecting

clusters over gene expression data. Many different clustering algorithms have been used over gene expression data. A survey of some of clustering algorithms is given in [AMS94]. Generally clustering algorithms partition the set of genes into clusters, where each cluster represents a group of co-expressed genes and the coherent pattern of that cluster is the mean (centroid) of the expression profiles of that cluster. Challenges in clustering gene expression data include the following.

- Most clustering algorithms generate disjoint clusters at a single level without hierarchical representation among the groups of co-expressed genes.
- Most clustering algorithms cannot adapt to local structures within the clusters
- The results of most clustering algorithms are dependent on appropriate parameter settings. Often results are different depending on parameter values and domain knowledge is required to assess the quality of the result.
- It is difficult to integrate domain knowledge into clustering algorithms.
- Since gene expression datasets consist of highly connected genes clustering becomes a difficult task and it is often hard to find clear borders [JPZ04].

In this thesis, we have developed three clustering algorithms. They include a graph-based algorithm, a frequent itemset-nearest neighbor based algorithm, and a density based clustering algorithms. Each clustering algorithm uses our own dissimilarity measure, presented in Chapter 3 of this thesis. Finally, we also develop an incremental clustering algorithm based on the density based clustering algorithm mentioned earlier. All clustering algorithms have been tested over synthetic and real life data and have been found to detect biologically relevant clusters w.r.t. various cluster validity measures.

1.5 Motivation

Based on a comprehensive literature survey we come to the following conclusions.

- Clustering algorithms are dependent on the proximity measure being used. Choosing an appropriate proximity measure is of utmost importance. That there exists no particular measure which can handle all the issues of gene clustering further complicates the job. It is highly desirable that the proximity measure being used is robust to outliers and can retain the regulation information inherent in a gene expression data.

- It has been observed that various clustering algorithms require different types of input parameters and clustering results are highly dependent on the value of the parameters. Graph based algorithms have a great advantage in that, they do not require the number of clusters as an input parameter and are robust to noise. However, it has been seen that graph based algorithms require an input parameter (threshold). It would be of great help if a graph based clustering algorithm could calculate the threshold dynamically during clustering.

- Gene expression data contains highly connected clusters. Therefore, it would be very helpful if finer clusters in the dataset could be identified. The finer clusters consists of genes having highly coherent patterns.

- A density-based clustering algorithm discovers clusters as highly dense regions separated by sparse regions. It is based on the concept of density connectedness between objects and can detect clusters of arbitrary shapes even in presence of noise. Therefore, detecting clusters over gene expression data using a density based approach would give rise to quality clusters.

- Due to the large number of microarray experiments being conducted the quantity of gene expression data is always increasing and new genes are continuously being discovered. As a result, it is desirable to cluster the newly available data incrementally instead of having to perform a re-clustering of the whole database.

1.6 Work Done

Following are our contributions reported in this thesis.

- Clustering is dependent on the selection of a proximity measure. In this thesis we present our own dissimilarity measure (DBK) which retains the regulation information present in the gene expression data and is robust to outliers. We establish it to perform equally well or better compared to existent proximity measures in clustering both synthetic and real-world data.

- We present an effective graph-based clustering algorithm (GCA) for gene expression data. Its main idea is that inter-cluster genes have more repulsion among them than intra-cluster genes. In particular, at any given moment, the genes are clustered based on a repulsion factor which is based on the genes that are yet to be assigned a cluster. This consideration leads to a objective function that is used to find cluster parameters that optimize this objective function. Comparison of this proposed algorithm with similar algorithms over different real world data sets shows the superiority of our algorithm.

- Frequent itemset and nearest neighbor concepts are considered to be useful for gene clustering. We develop a nearest neighbor based clustering algorithm based on a popular frequent itemset generation technique. It expands clusters using the nearest neighbor concept based on frequent itemsets. The process is iterated over multiple passes. After each pass, it prunes those genes that have already been assigned clusters. Experimental evaluation establishes that the method can find finer clustering of the dataset. The finer clustering produces clusters consisting of highly coherent gene patterns.

- We present a density based clustering algorithm (DGC) that uses the regulation information and the order preserving nature that exist in gene expression profiles to cluster genes into high density regions separated by sparse density regions. We validate DGC on real-life data sets and establish it to be effective.

- We also present a modified version of the DGC algorithm to handle the scenario when the input data, instead of being all available simultaneously, arrive incrementally. It is based on the concept that the density connections

of a newly arrived gene affects only the neighborhood of the gene. Experimental results on several real world gene expression data demonstrate that the incremental algorithm can cluster the data significantly faster while at the same time obtain the same result as when DGC is applied to the whole updated database.

1.7 Organization of the Thesis

The thesis is organized as follows:

- *chapter* 2 describe how gene expression data is collected. This chapter also gives a survey of literature regarding coherent gene expression pattern identification using data mining techniques.
- *Chapter* 3 presents our own dissimilarity measure (DBK). The measure is established to be appropriate while clustering both synthetic and real-world data.
- In *Chapter* 4, an effective graph-based clustering algorithm (GCA) for gene expression data is presented.
- A frequent itemset nearest neighbor based clustering algorithm (FINN) is reported in *Chapter* 5.
- *Chapter* 6 of this thesis describes the density based clustering algorithm (DGC).
- An incremental version of DGC algorithm that can handle incremental datasets is presented in *Chapter* 7.
- Finally, concluding remarks are given in *Chapter* 8.

All our clustering algorithms are validated using various statistical validity measures to show their effectiveness over biological data while comparing with well-chosen similar algorithms.

Chapter 2

Gene Expression Pattern Identification

2.1 Microarray Technology: A Brief Overview

In 1970, Francis Crick introduced the central dogma of molecular biology [KW02] which has ever since, been one of the pillars of modern molecular biology It pins down DNA (Deoxyribonucleic acid) as the carrier of genetic information and describes the unidirectional flow of information from DNA via RNA (Ribonucleic acid) to protein in three steps *Replication*, *Transcription* and *Translation* This dogma is at the heart of bioinformatics which provides the framework to interrelate and interpret different types of data encountered in this field The central dogma of molecular biology refers to the process of protein synthesis, which occurs in three major stages The first stage, *Replication* is the process which results in the duplication of the genetic information coded in DNA strands The second stage, *Transcription*, is the transfer of information from double stranded DNA into single-stranded mRNA The third stage, *Translation*, refers to the conversion inside the cell where mRNA is translated to produce a protein Together *Transcription* and *Translation* constitute *Gene Expression* Gene expression experiments provide a method to quantitatively measure the transcription phase of protein synthesis The objective of gene expression experiments is the quantitative measurement of mRNA expression particularly under the influence of drug

or disease perturbations.

A DNA microarray or gene chip consists of an array of oligonucleotides or complementary DNA (cDNA) molecules of known composition chemically bonded to a solid surface (made of chemically coated glass, nylon membrane or silicon) [Ste06]. Gene chips are usually categorized into one of two classes, based on the DNA actually arrayed onto the support. An oligo array is comprised of synthesized oligonucleotides, whereas a cDNA array contains cloned or PCR-amplified cDNA (complementary DNA) molecules [KW02] Both classes involve three common basic procedures [Ste06] which are depicted in Figure 2.1.

i *Chip manufacture*: A microarray is a small chip, onto which tens of thousands of DNA molecules (probes) are attached in fixed grids. Each grid cell relates to a DNA sequence. The DNA on the array are referred to as *probes* and the labeled DNA in solution as *target*.

ii *Sample preparation, labeling, hybridization and washing.* The first step is the extraction of RNA from the tissue of interest. Next, two mRNA samples are reverse-transcribed into cDNA and labeled using either fluorescent dyes (Cy3 and Cy5) or radioactive isotopes. It is then hybridized with the probes on the surface of the chip. Hybridization is the step in which the DNA probes on the glass and the labeled DNA (or RNA) target form heteroduplexes via Watson-Crick base pairing. After hybridization, the slides are washed (using a low-salt wash or with a high-temperature wash) to remove excess hybridization solution from the array. This ensures that only the labeled target on the array is the target that has specifically bound to the features on the array. This step also reduces cross-hybridization. This process is illustrated in Figure 2 2 reproduced from *http://titan.biotec.uiuc.edu/cs491jh/slides/cs491-lei ppt.*

iii *Image Acquisition*: In this step, an image of the surface of the hybridized array (chip) is produced by scanning the chip to read the signal intensity that is emitted from the fluorescent dye of the heteroduplexes on the array where the target has bound to the probe.

The image acquisition process is shown in detail in Figure 2.3 and has been repro-

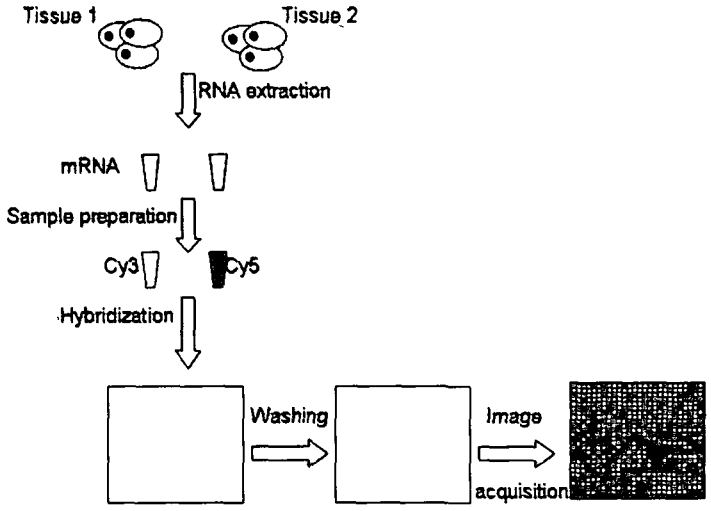


Figure 2.1: Steps in a microarray experiment (Courtesy of [Ste06]). The Cy3 and Cy5 in the diagram refers to the mRNAs dyed using the two fluorescent dyes of Cy3 and Cy5.

duced from *http://archive.student.bmj.com/issues/08/07/education/images/fig_2.jpg*.

After hybridization, the slides are scanned using a laser device to determine the amount of fluorescent label that is attached to each cDNA on the slide. The amount of fluorescence is displayed as a cell on a matrix corresponding to the spot on the original slide. The images output from the scanner are colored according to a standard where a higher level of fluorescent label (enhanced gene expression) is colored red, a lower level (repressed level of gene expression) is colored green and equal levels are yellow.

The digital image obtained from the image acquisition step is converted into numerical measures of hybridization intensity for each channel on each feature [Ste06]. The image is analyzed by (i) *Gridding*: Identify spots (this step can be automatic, semiautomatic or manual); (ii) *Segmentation*: Separate spots from background using fixed circle, adaptive circle, adaptive shape or histogram methods; (iii) *Intensity extraction*: Obtain mean or median of pixels in spots and (iv) *Background correction*: can be either local or global. The microarray data thus generated is then cleaned, transformed and normalized to resolve any errors, noise

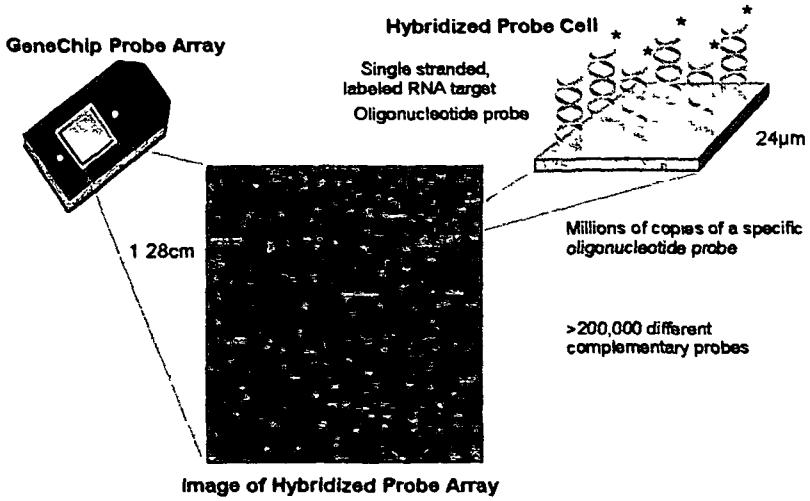


Figure 2.2: Hybridization of probe array

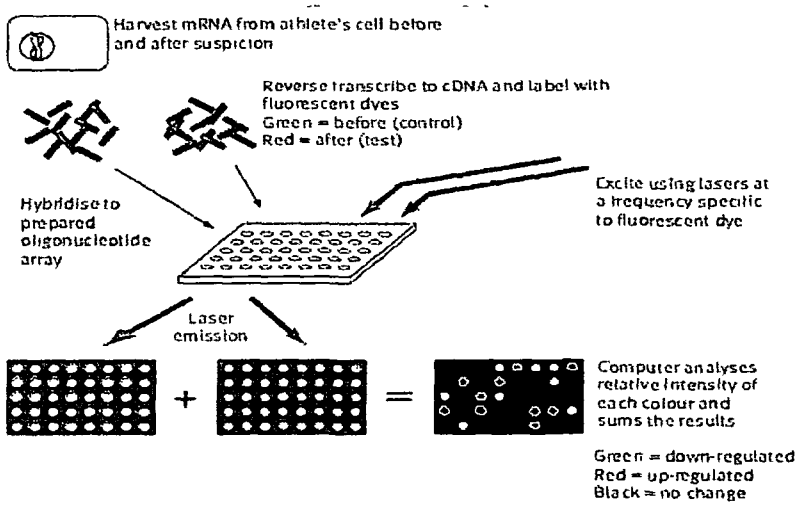


Figure 2.3: The image acquisition process.

and bias introduced by the microarray experiments [Ste06]. The logarithm of the raw intensities are taken to convert them into log intensities. Once expression data is obtained from the microarray images using various standardization and normalization procedures [Ste06], the information embedded in the data has to be analyzed.

A gene is expressed in a cell when the protein it codes for is actually synthesized. About 10,000 genes are expressed in an average human cell. The set of (say 10,000) numbers that indicate the expression level of each of these genes is called the expression profile of the cell.

The power of a microarray experiment derives from the fact that many thousands of different DNA molecules are bonded to a single array. So it is possible to measure the expression of many thousands of genes simultaneously, conveniently and efficiently.

2.2 Gene Expression Data

Gene expression of a gene refers to effective production of the protein that a gene encodes. A microarray experiment assesses a large number of DNA sequences (genes) under multiple conditions such as time-series, tissue samples (e.g., normal versus cancerous tissues), and experimental conditions. A gene expression data set from a microarray experiment may be considered as a $G \times T$ matrix D_G as shown in Equation 2.1, the rows of which represent expression patterns of a set of G genes $\{g_1, \cdots, g_G\}$, and the columns represent expression profiles of a set of T samples, $S = \{s_1, \cdots, s_T\}$ and each cell $g_{i,j}$ is the expression level of gene g_i $(where\ 1 \leq i \leq G)$ on sample s_j $(where\ 1 \leq j \leq T)$.

$$D_G = \begin{bmatrix} g_{11}\ g_{12} \cdots \cdots g_{1T} \\ g_{21}\ g_{22} \cdots \cdots g_{2T} \\ \vdots \\ g_{G1}\ g_{G2} \cdots \cdots g_{GT} \end{bmatrix} \tag{2.1}$$

Cluster analysis starts with this gene expression matrix and calculates proximity among genes. Clustering algorithms group genes which are similar based on a

proximity measure into the same cluster Therefore, similar genes are grouped into the same cluster and dissimilar genes are grouped into different clusters

During the last few years, several significant coherent pattern identification techniques have been developed under the categories of gene based, sample based and subspace clustering approaches The next section is dedicated to reviewing the popular algorithms

2.3 Gene Expression Pattern Identification using Data Mining

Cluster Analysis is an unsupervised grouping technique used to group similar objects (in this case genes) into disjoint sets based on their attribute (condition) similarities Clustering identifies genes with similar expression profiles (co-expressed genes) Co-expressed genes have similar expression profiles, while a coherent expression pattern represents the common trend among expression levels for a group of co-expressed genes Furthermore, co-expressed genes in the same cluster are likely to be involved in the same cellular processes, and a strong correlation among expression patterns of the genes indicates co-regulation In practice, co-expressed genes may belong to the same or similar functional categories and indicate co-regulated families [AMS94] Various gene clustering methods have been used to identify co-expressed genes and discover coherent expression patterns A cluster of genes contains a group of co-expressed genes and the coherent expression pattern is obtained as the mean (the centroid) of the expression profiles of the genes in the cluster

Gene clustering techniques are divided into three different types *gene-based clustering* where the genes are treated as objects while the samples are features, *sample-based clustering* in which the samples can be partitioned into homogeneous groups where the genes are regarded as features and the samples as objects, and *subspace clustering* in which either genes or samples can be regarded as objects or features Both gene-based and sample-based clustering approaches search exclusive and exhaustive partitions of objects that share the same feature space Subspace clustering algorithms capture clusters formed by a subset of

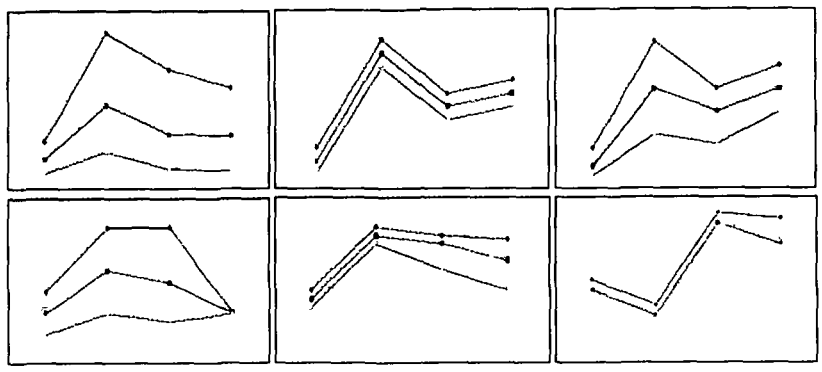


Figure 2.4: Clusters of genes showing similar temporal patterns

genes across a subset of samples. Throughout the work reported in this thesis, we use the gene-based clustering approach. In gene-based clustering, similar rows (genes) are grouped together into unique clusters. The premise is that each cluster shows a similar temporal expression pattern as shown in Figure 2.4 and may represent a group of functionally related genes i.e., a *biological module.*

2.4 Proximity Measures

A microarray experiment compares genes from an organism under different development time points, conditions or treatments. For a T condition experiment, a single gene has a T-dimensional observation vector known as its gene expression profile. A similarity (or dissimilarity) measure is a real-valued function that assigns a real number as a similarity or dissimilarity value between any two expression vectors. Therefore, to identify genes or samples that have similar expression profiles, appropriate similarity (or dissimilarity) measures are required. Some of the commonly used distance metrics are: Euclidean distance, Pearson's Correlation coefficient and Spearman's Rank-order Correlation Coefficient [Ste06]. The Euclidean distance measure imposes a fixed geometrical structure and finds clusters of that shape even if they are not present. It is scale variant and cannot detect negative correlation. Euclidean distance gives the distance between two genes but does not focus on the correlation between them. Pearson's Correlation, on the other hand, retains the correlation information between two genes as well

as the regulation information However, since it uses the mean values while computing the correlation between genes, a single outlier can aberrantly affect the result Spearman's Rank Correlation is not affected by outliers, however there is information loss w r t regulation since it works on ranked data

2.5 Gene Expression Data Clustering Approaches

Data mining techniques have proven to be useful in understanding gene function, gene regulation, cellular processes and subtypes of cells According to [Ste06], most data mining algorithms developed for gene expression time series deal with the problem of clustering Clustering identifies subsets of genes that behave similarly along a course of time Categorization of gene expression data clustering techniques is discussed next

2.5.1 Partitioning Approaches

k-means [McQ67] is a typical partition-based clustering algorithm which divides the data into pre-defined number of clusters in order to optimize a predefined criterion The major advantages of it are its simplicity and speed, which allows it to run on large datasets However, it may not yield the same result with each run of the algorithm Often, it can be found incapable of handling outliers and is not suitable to detect clusters of arbitrary shapes A Self Organizing Map (SOM) [Koh95] is more robust than k-means for clustering noisy data It requires the number of clusters and the grid layout of the neuron map as user input Specifying the number of clusters in advance is difficult in case of gene expression data Moreover, partitioning approaches are restricted to data of lower dimensionality, with inherent well-separated clusters of high density But, gene expression data sets may be high dimensional and often contain intersecting and embedded clusters QT (quality threshold) clustering [HKY99] is an alternative method of partitioning data, invented for gene clustering It requires more computing power than k-means, but does not require specifying the number of clusters apriori, and always returns the same result when run several times The distance between a point and a group of points is computed using complete linkage, i e , as the

maximum distance from the point to any member of the group [ESBB98]. A hierarchical structure can also be built based on SOM such as Self-Organizing Tree Algorithm (SOTA) [DC97]. Recently, several new algorithms such as [HVD01] and [THHK02] have been proposed based on the SOM algorithm. These algorithms can automatically determine the number of clusters and dynamically adapt the map structure to the distribution of data. Herrero et al. [HVD01] extend the SOM by a binary tree structure. At first, the tree only contains a root node connecting two neurons. After a training process similar to that of the SOM algorithm, the data set is segregated into two subsets. Then the neuron with less coherence is split in two new neurons. This process is repeated level by level, until all the neurons in the tree satisfy some coherence threshold. Other examples of SOM extensions are Fuzzy Adaptive Resonance Theory (Fuzzy ART) [THHK02] which provide some approaches to measure the coherence of a neuron (e.g., vigilance criterion). The output map is adjusted by splitting the existing neurons or adding new neurons into the map, until the coherence of each neuron in the map satisfies a user specified threshold.

2.5.2 Hierarchical Approaches

Hierarchical clustering generates a hierarchy of nested clusters. These algorithms are divided into agglomerative and divisive approaches. Unweighted Pair Group Method with Arithmetic Mean (UPGMA), presented in [ESBB98], adopts an agglomerative method to graphically represent the clustered dataset. However, it is not robust in the presence of noise. In [ABN+99], the genes are split through a divisive approach, called the Deterministic-Annealing Algorithm (DAA). The Divisive Correlation Clustering Algorithm (DCCA) [BD08] uses Pearson's Correlation as the similarity measure. All genes in a cluster have highest average correlation with genes in that cluster. Hierarchical clustering not only groups together genes with similar expression patterns but also provides a natural way to graphically represent the data set allowing a thorough inspection. However, a small change in the data set may greatly change the hierarchical dendrogram structure. Another drawback is its high computational complexity.

2.5.3 Density Based Approaches

Density based clustering identifies dense areas in the object space. Clusters are highly dense areas separated by sparsely dense areas. DBSCAN [EKSX96] was one of the pioneering density based algorithms used over spatial datasets. In [JPZ03a], Jiang *et. al.* propose the Density-Based Hierarchical clustering method (DHC) to identify co-expressed gene groups. It can identify embedded clusters in the dataset and can also handle outliers. It can effectively visualize the internal structure of the data set. A kernel density clustering method for gene expression profile analysis is reported in [SZCS03]. An alternative to this is to define the similarity of points in terms of their shared nearest neighbors. This idea was first introduced by Jarvis and Patrick [JP73]. A density-based approach discovers clusters of arbitrary shapes even in the presence of noise. However, density-based clustering techniques suffer from high computational complexity with increase in dimensionality (even if spatial index structure is used) and input parameter dependency.

2.5.4 Model Based Approaches

Model based approaches provide a statistical framework to model the cluster structure in gene expression data. The Expectation Maximization (EM) algorithm [DLR77] discovers good values for its parameters iteratively. It can handle various shapes of data, but can be very expensive since a large number of iterations may be required. In [TH09], a signal shape similarity method used to cluster genes using a Variational Bayes Expectation Maximization algorithm [BG03]. A model-based approach provides an estimated probability that a data object will belong to a particular cluster. Thus, a gene can have high correlation with two totally different clusters. However, the model-based approach assumes that the data set fits a specific distribution which is not always true.

2.5.5 Graph Theoretical Approaches

In graph-based clustering algorithms, graphs are built as combinations of objects, features or both, as nodes and edges, and partitioned by using graph theoretic

algorithms Graph theoretic algorithms are also used for the problem of clustering cDNAs based on their oligo-nucleotide fingerprints [HSL+99] CLuster Identification via Connectivity Kernels (CLICK) [SS00] is suitable for subspace and high dimensional data clustering The Cluster Affinity Search Technique (CAST) by [BDSY99] takes as input pairwise similarities between genes and an affinity threshold It does not require a user-defined number of clusters and handles outliers efficiently But, it faces difficulty in determining a good threshold value In CAST, the size and number of clusters produced is directly affected by the fixed user-defined parameter t and hence, apriori knowledge of the data set is required To overcome this problem, E-CAST [BPC02] calculates the threshold value dynamically based on the similarity values of the objects that are yet to be clustered

2.5.6 Soft Computing Approaches

Fuzzy c-means [Bez81a] and Genetic Algorithms (GA) (such as [BMM07] and [MMB09]) have been used effectively in clustering gene expression data The Fuzzy c-means algorithm requires the number of clusters as an input parameter The GA based algorithms have been found to detect biologically relevant clusters but are dependent on proper tuning of the input parameters

The current information explosion, fuelled by the availability of the World Wide Web and the huge amount of microarray experiments being conducted, have led to ever-increasing volume of data Therefore, there is a need to introduce incremental clustering so that updates can be clustered in an incremental manner Though a lot of research has been performed on incremental clustering in other application domains, incremental clustering of gene expression data has not been exploited much yet

2.5.7 Incremental Algorithms

In [EKS+98], the authors present an incremental clustering approach based on the DBSCAN [EKSX96] algorithm A one pass clustering algorithm for relational datasets is proposed in [TS08] Rough set theory is employed in the incremental

approach for clustering interval datasets in [ANS06]. In [LLF+04b], an incremental genetic k-means algorithm is presented. In [RRAR06], an incremental gene selection algorithm using a wrapper-based method that reduces the search space complexity since it works on the ranking directly, is presented.

2.6 Discussion

From the discussion above, we conclude that various clustering algorithms require different types of input parameters and clustering results are highly dependent on the values of parameters. Gene expression data has coherent patterns embedded in the full gene space, identification of which is an important research field. Coherent genes may indicate co-regulation and hence fall under the same functional classification. Clustering algorithms that do not require the number of clusters as an input parameter and are robust to noise are of utmost importance. Clustering algorithms are sensitive to the proximity measure chosen. In this thesis, we present several clustering methods and all of them use a proximity measure developed by us which is introduced in the next chapter.

Chapter 3

A Novel Dissimilarity Measure for Clustering Gene Expression Data

Distance or similarity measures, also known as proximity measures, are essential to solve many pattern recognition problems such as classification, clustering, and retrieval Various distance (similarity) measures have been reported in the literature However, choosing an appropriate measure for a specific problem depends on the problem domain as well as data domain Clustering is based on proximity measures that help in detecting clusters based on proximity among different objects Clustering of gene expression data is highly sensitive to the proximity measure used Choosing an appropriate dissimilarity measure is of utmost importance In this chapter, a review of some similarity (dissimilarity) measures is given and a dissimilarity measure (DBK) is proposed for effective clustering of gene expression data In this thesis we have used the gene based clustering The DBK measure retains the good characteristics of several commonly used proximity measures while avoiding the bad characteristics It retains the up-down- regulation information inherent in gene expression data and is robust in the presence of outliers

3.1 Distance Metric

A metric or distance function defines a distance between elements of a set A metric on a set Y is a function (called the distance function or simply distance)

$$d \quad Y \times Y \rightarrow R$$

where R is the set of real numbers For all $x, y, z \in Y$, this function should satisfy the following properties

i) $d(x, y) \geq 0$ (non-negativity) Distance is always positive or zero

ii) $d(x, y) = 0$ iff $x = y$ (identity of indiscernibles) Distance is zero if and only if it measured to itself

iii) $d(x, y) = d(y, x)$ (symmetry) Distance is symmetric

iv) $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality) Distance satisfies triangle inequality

A distance function is also called metric if it satisfies all four conditions given above Thus, because of the triangle inequality (condition iv), not all distance measures are metric, but all metrics are distances

3.2 Similarity and Dissimilarity Measures

From scientific and mathematical points of view, distance is defined as a quantitative degree of how far apart two objects are Similarity is a numerical quantity that reflects the strength of relationship between two objects or two features Similarities are higher for pairs of objects that are more alike This quantity is usually in the range of either -1 to +1 or is normalized into 0 to 1 If the similarity between object x and object y is denoted by $S_{x\,y}$, we can measure this quantity in several ways depending on the scale of measurement (or data type) that we have

Distance measure is also known as dissimilarity measure. Similarity and dissimilarity measures are often called proximity measures. Dissimilarity measures the discrepancy between the two objects, i.e., it measures the degree to which two objects are different. There are many types of distance and similarity measures. Each similarity or dissimilarity measure has its own characteristics. Next, we consider several important issues concerning proximity measures.

3.2.1 Relationship between Similarity and Dissimilarity

Let normalized dissimilarity between object x and object y be denoted by $d(x, y)$. Then the relationship between dissimilarity and similarity is given by

$$S_{x,y} = 1 - d(x, y). \tag{3.1}$$

Here, $S_{x,y}$ is normalized similarity between objects x and y. Similarity is bounded by 0 and 1. When similarity is one (i.e., two objects are exactly similar), the dissimilarity is zero and when the similarity is zero (i.e., two objects are very different), dissimilarity is one. If the value of similarity has range of -1 to +1, and the dissimilarity is measured with range of 0 and 1,

$$S_{x,y} = 1 - 2d(x, y). \tag{3.2}$$

When dissimilarity is one (i.e., two objects are very different), similarity is minus one and when the dissimilarity is zero (i.e., two objects are very similar), similarity is one. In many cases, measuring dissimilarity (i.e., distance) is easier than measuring similarity. Once we can measure dissimilarity, we can easily normalize it and convert it to similarity measure. It is also common for dissimilarities to range from 0 to ∞.

Frequently, proximity measures are transformed to the interval [0, 1]. The transformation of similarities to the interval [0, 1] is given by

$$S'_{x,y} = \frac{S_{x,y} - min_{S_{x,y}}}{max_{S_{x,y}} - min_{S_{x,y}}} \tag{3.3}$$

where, $min_{S_{x,y}}$ and $max_{S_{x,y}}$ are minimum and maximum similarities respectively. Similarly, dissimilarity measures with a finite range can be mapped to the interval

[0, 1] by using the formula

$$d'(x,y) = \frac{d(x,y) - min_{d(x,y)}}{max_{d(x,y)} - min_{d(x,y)}} \tag{3.4}$$

where, $min_{d(x,y)}$ and $max_{d(x,y)}$ are minimum and maximum dissimilarities respectively.

If the proximity measure has values in the range $[0, \infty]$, a non-linear transformation is needed and the values in the transformed scale will not have the same relationship to one another as the original. But, whether such a transformation is desirable or not depends on the application it is used.

3.2.2 Different Similarity and Dissimilarity Measures

In this section various kinds of dissimilarities and similarities are discussed.

A. Some Dissimilarity Measures

This section reports on some of the popular dissimilarity measures.

i. Euclidean distance

Euclidean distance (Equation 3.5) expressed in terms of the Pythagorean theorem is one of the most popular distance measures in use today. The Euclidean distance between two sets of objects x and y in n-dimensional space is defined as

$$Euclidean(x,y) = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2} \tag{3.5}$$

where, $x = x_1, x_2, ..., x_n$ and $y = y_1, y_2, ..., y_n$.

ii. Manhattan distance

Minkowski presented the city block distance [Kra75]. The city block or the Manhattan distance or L_1 norm is given below

$$d(x,y) = \left| \sum_{i=1}^{n} (x_i - y_i) \right|. \tag{3.6}$$

iii. Minkowski distance

Minkowski's distance is the generalized form of the two distance metrics discussed above. It is given as

$$d(x,y) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p} \tag{3.7}$$

where p is a parameter. For $p = 1$, we get the Manhattan distance, for $p = 2$ we get the Euclidean distance.

iv. Chebyshev distance

For $p = \infty$ we get the Supremum or Chebyshev (L_{max} or L_∞ norm) distance named after Chebychev [HDRT04]. This is the maximum distance between any attribute of the objects. Formally, L_∞ is defined as

$$d(x,y) = \lim_{p \to \infty} \left(\sum_{k=1}^{n} |x_k - y_k|^p \right)^{\frac{1}{p}}. \tag{3.8}$$

B. Some Similarity Measures

The triangle equality does not hold for similarity measures but the following properties hold true:

i) $S_{x,y} = 1$ only if $x = y$ $(0 \leq S \leq 1)$. and

ii) $S_{x,y} = S_{y,x}$ for all x and y (Symmetry).

i. Similarity measures for binary data

Similarity measures between objects that have only binary attributes are called *similarity coefficients* and have values between 0 and 1. A value of 0 means that the objects are completely dissimilar and a value of 1 means that the objects are completely similar.

Suppose objects x and y have n binary attributes. Then, on comparing x and y the following quantities are obtained:

i) q_{00}: the number of attributes where $x = 0$ and $y = 0$,

ii) q_{01}: the number of attributes where $x = 0$ and $y = 1$,

iii) q_{10}: the number of attributes where $x = 1$ and $y = 0$, and

iv) q_{11}: the number of attributes where $x = 1$ and $y = 1$.

file:///home/rosy/thesis Using the above quantities different similarity coefficients can be obtained.

Simple Matching Coefficient

Simple Matching Coefficient or SMC [HK04] is one of the most commonly used similarity coefficients and is defined as,

$$SMC = \frac{Total\ number\ of\ matched\ attributes}{Total\ attributes} = \frac{q_{00} + q_{11}}{q_{00} + q_{01} + q_{10} + q_{11}}. \quad (3.9)$$

SMC gives equal weight to both presences and absences.

Jaccard Coefficient

Jaccard Coefficient [HK04] is used for handling objects consisting of asymmetric binary attributes. Jaccard Coefficient (J) is defined as follows,

$$J = \frac{Number\ of\ matched\ attributes}{Number\ of\ attributes\ not\ involved\ in\ 00\ matches} = \frac{q_{11}}{q_{01} + q_{10} + q_{11}}. \quad (3.10)$$

ii. Cosine Similarity

Cosine similarity [TSK09] is useful for finding document similarity. If x and y are two document vectors, $cos(x, y)$ is given by the following equation,

$$cos(x, y) = \frac{x \cdot y}{\|x\|\|y\|} \quad (3.11)$$

where . indicates the vector dot product, $x.y = \sum_{k=1}^{n} x_k y_k$, and $\|x\|$ is the length of vector x, and $\|x\| = \sqrt{\sum_{k=1}^{n} x_k^2} = \sqrt{x \circ x}$.

iii. Extended Jaccard Coefficient

The Extended Jaccard Coefficient (EJ) [TSK09] can be used for document data and it reduces to the Jaccard Coefficient in case of binary data.

$$EJ(x,y) = \frac{x \cdot y}{\|x\|^2 + \|y\|^2 - x \cdot y}. \tag{3.12}$$

iv. Correlation

Pearson's correlation coefficient [TSK09] is a widely used similarity measure. It is defined as

$$corr(x,y) = \frac{covariance(x,y)}{standard\_deviation(x) * standard\_deviation(y)} = \frac{cov_{xy}}{\sigma_x \sigma_y} \tag{3.13}$$

where

$$\begin{aligned} cov_{xy} &= \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y), \\ \sigma_x &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_x)^2}, \\ \sigma_y &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \mu_y)^2}, \\ \mu_x &= \frac{1}{n}\sum_{i=1}^{n} x_i \text{ is the mean of } x, \text{ and} \\ \mu_y &= \frac{1}{n}\sum_{i=1}^{n} y_i \text{ is the mean of } y. \end{aligned} \tag{3.14}$$

Pearson's correlation is always in the range [-1, 1]. Correlation and Euclidean distance are useful for dense data such as time series or two-dimensional points while Jaccard and cosine similarity measures are useful for sparse data like documents.

Pearson's correlation is a powerful similarity measure. However, empirical study has shown that it is not robust in presence of outliers [HKY99], thus potentially yielding false positives which assign a high similarity score to a pair of

dissimilar patterns If two patterns have a common peak or valley at a single feature, the correlation will be dominated by this feature, although the patterns at the remaining features may be completely dissimilar Another drawback of Pearson's correlation coefficient is that it assumes an approximate Gaussian distribution of the points and may not be robust for non-Gaussian distributions [Bic01]

v. Jackknife correlation

Jackknife correlation [JTZ03], helps in overcoming the single outlier problem of Pearson's correlation It is defined as

$$Jackknife(x,y) = min\{corr(x,y)^1, \quad , corr(x,y)^l, \quad , corr(x,y)^n\} \quad (3\ 15)$$

where $corr(x,y)^l$ is the Pearson's correlation coefficient of data objects x and y with the l^{th} feature deleted Use of Jackknife correlation avoids the dominance effect of single outliers More general versions of Jackknife correlation that are robust to more than one outlier can similarly be derived However, generalized Jackknife correlation, which involves the enumeration of different combinations of features to be deleted, is computationally costly and is rarely used

vi. Spearman/s rank-order correlation coefficient

To address the problem of non-Gaussian distributions w r t Pearson's correlation, Spearman's rank-order correlation coefficient [JTZ03] has been suggested as a similarity measure The ranking correlation is derived by replacing the data x_{in} with its rank r_{in} among all conditions For example, $r_{in} = 3$ if x_{in} is the third highest value among x_{ik} , where $1 \leq k \leq n$ Spearman's correlation coefficient does not require the assumption of Gaussian distribution and is more robust against outliers than Pearson's correlation coefficient However, as a consequence of ranking, a significant amount of information present in the data is lost

vii. $CorHsim$

In [LWN+09], a new similarity measure foi gene expression data, $CorHsim$, is presented It reflects the magnitude and shape information of gene expression

data at the same time and is defined as follows:

$$CorHsim(x,y) = \frac{1}{2n}\sum_{i=1}^{n}\left(\frac{|\,(x_i-\mu_x)(y_i-\mu_y)\,|}{\sigma_x\sigma_y} + \frac{1}{1+|\,x_i-y_i\,|}\right). \quad (3.16)$$

The disadvantage of $CorHsim$ is that it uses the mean value and therefore may be affected by a very large or a small value of x or y.

The similarity (dissimilarity) measures discussed above have been applied in various domains. However, not all the measures are applicable in all domains. There is a qualitative domain specific dependency among similarity (dissimilarity) measures. For gene expression data domain, not all the measures discussed above are applicable. Gene expression data with its inherent high dimensionality and direction information becomes challenging for similarity (dissimilarity) measures. The following section discusses in detail the various aspects of proximity measures used for gene expression data and presents our dissimilarity measure, DBK.

3.3 Proximity Measures for Gene Expression Data

There are many methods for quantifying similarity or dissimilarity between a pair of gene expression profiles. Different methods give different results and therefore one should carefully choose which method to use.

3.3.1 Features of a Distance Measure

Similarity or dissimilarity between two profiles is described in terms of the distance between them in the high dimensional space of gene expression measurements. A dissimilarity measure, $d(g_i, g_j)$, obeys the following four properties (1-4) for any two genes g_i and g_j, while a true distance measure also satisfies a fifth ([JW98]). The properties are given below.

1. The distance between any two profiles cannot be negative.

2. The distance between a profile and itself must be zero.

3. The distance between profile g_i and profile g_j is the same as distance between profile g_j and profile g_i, i.e., $d(g_i, g_j) = d(g_j, g_i)$.

4. The distance measure should obey the triangle inequality property, i.e., for profiles g_i, g_j, g_q, we have $d(g_i, g_q) \leq d(g_i, g_j) + d(g_j, g_q)$.

3.3.2 Comparing Similarity Measures

A microarray experiment compares genes from an organism under different development time points, conditions or treatments. For an n condition experiment, a single gene has an n-dimensional observation vector known as its gene expression profile. A similarity (or dissimilarity) measure is a real-valued function that assigns a positive real number as a dissimilarity value between any two expression vectors. Therefore, to identify genes or samples that have similar expression profiles, appropriate similarity (or dissimilarity) measures are required. Some of the commonly used distance metrics are: Euclidean Distance, Pearson's Correlation coefficient and Spearman's rank-order correlation coefficient [JTZ03]. A comparison of three most widely used distance measures used for gene expression data is given in Table 3.1[1]. Euclidean distance imposes a fixed geometrical structure [Ste06] and finds clusters of that shape even if they are not present. It is scale variant and cannot detect negative correlation. Euclidean distance gives the distance between two genes but does not focus on the correlation between them. Pearson's Correlation, on the other hand, retains the correlation information between two genes as well as the regulation information. However, since it uses the mean values while computing the correlation between genes, a single outlier can aberrantly affect the result. Spearman's rank correlation is not affected by outliers; however, there is information loss w.r.t. regulation since it works on ranked data. Thus, it can also be observed from Table 3.1 that choosing an appropriate distance measure for gene expression data is a difficult task.

[1]Note that this table's first column corresponds to the distance measure we introduce later in this chapter

Table 3 1: Comparison of Different Distance Measures

DBK	*Euclidean Distance*	*Pearson's Correlation*	*Spearman's Rank Correlation*
Statistical	Geometric Interpretation	Statistical	Robust for non-Gaussian distributions
Retains up- and down-regulation information	Retains up- or down- regulation information with appropriate scaling	Spots positive and negative correlation	Spots positive and negative correlation
Scale invariant on centered data	Not scale invariant; Results dependent on scaling	Scale invariant on centered data	Completely scale invariant; No scaling or centering required
Robust to outliers	Cannot detect negative Correlation	Susceptible to outliers	Robust to outliers
Can detect magnitude of change	Can detect magnitude of change if used without scaling	Assumes linearity	Ignores up- or down regulation in time series

3.4 Motivation

From the discussion above, we conclude that choosing an appropriate similarity measure is of utmost importance. That there exists no particular measure which can handle all the issues further complicates the job.

In the rest of the chapter, we introduce a dissimilarity measure, DBK that addresses the challenge of identifying the coherent patterns from a gene expression dataset It can find dissimilarity between gene profiles effectively. While clustering genes, DBK measure can identify coherent patterns even in the presence of

outliers That it retains information on the shape (trend) of the patterns is the major attraction and thus DBK gives the coherency measure between genes

3.5 DBK Dissimilarity Measure

To compute the dissimilarity between pairs of genes, our method uses a grid-based approach, where the number of grid cells is computed according to the spread of the dataset

3.5.1 Grid Approximation

Initially, the dataset is normalized to mean 0 and standard deviation 1 The dataset (gene profiles) is then plotted as a two-dimensional curve with time points in the horizontal direction and expression levels in the vertical direction The maximum and minimum expression levels as well as the maximum and minimum time points are found The spatial region is approximated by a quadrilateral of size $A \times B$, where the quadrilateral can be represented by the set of points (*minimum time point, minimum expression level*), (*maximum time point, maximum expression level*) This quadrilateral is divided into rectangular cells of width W, where $W = 2^{q-l}$, $A = 2^q a$, $B = 2^q b$ for some positive integers a, b, q and level, l as shown in Figure 3 1 Initially, $l = 0$ i e , the block width is 2^q, $a = b = 1$ and we have one rectangular cell with all the gene profiles lying within it For each cell, a feature condition (discussed in the next section) is tested based on the expression values of the gene profiles within that cell If the condition is not satisfied, the region is further divided into 2^k equal sub-cells, where $k = 2$, 4, 8, The process continues till the condition is satisfied Figure 3 2 (a) shows the initial condition of some example gene profiles at level, $l = 0$, Figure 3 2 (b) shows the intermediate condition at level, $l = 1$ and Figure 3 2 (c) shows the final condition of the example gene profiles at level, $l = 2$

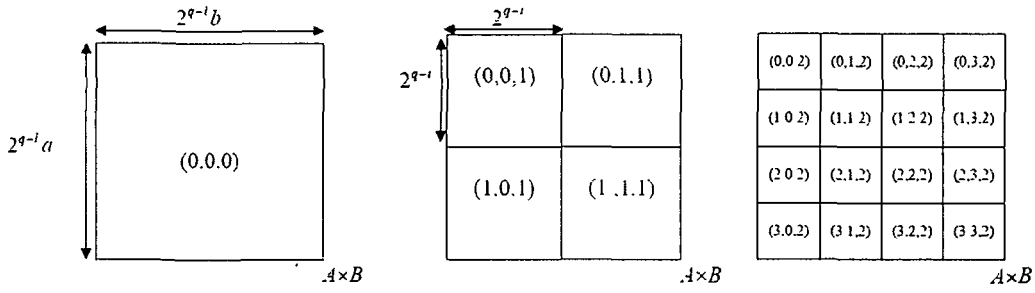


Figure 3.1: Cell ID for (a) level 0, (b) level 1 and (c) level 2.

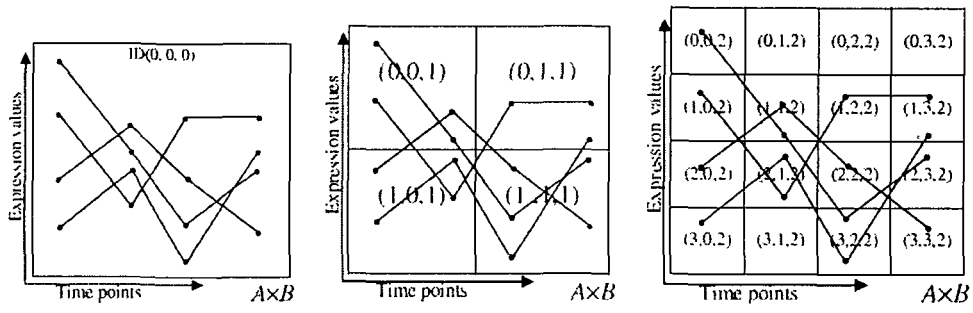


Figure 3.2: Example gene profiles in (a) level 0, (b) level 1 and (c) level 2.

3.5.2 Feature Condition for a Grid Cell

Initially at level, $l = 0$, all the profiles reside in one cell as shown in Figure 3.2 (a). So, at $l = 0$, the row, $r = 0$ and column $c = 0$. Therefore, the ID of this cell is given by $ID(r, c, l)$ i.e., $ID(0, 0, 0)$. Note that, points of a profile refer to the points across time points (i.e., points of a profile in the horizontal direction) as is shown by the solid circles for each example profile in the figure. Now, the condition for cell division is that, two consecutive points of a profile cannot reside within the same column, and cannot reside within the same row, if their expression levels are unequal. We call this the *feature condition.*

Figure 3.2 (a) shows the initial condition of the example gene profiles at level $l = 0$. We see that consecutive points of all the profiles reside within the same row and column. Therefore, we divide the cell into 2^k equal sub-cells, where k

= 2, and level, l now becomes 1. Thus we obtain 4 grid cells as can be seen in Figure 3.2 (b) at $l = 1$. We observe in this figure that the second gene profile has consecutive points in the cell ID(0,1,1) but since both the points have equal expression values they can reside in the same row. But, since they reside in the same column for the two consecutive time points, it does not satisfy the feature condition and subdivision is required. Similarly, for the other genes, consecutive time points reside within the same column and therefore subdivision is required. Also for the other genes, we see that genes with dissimilar expression values reside in the same row, for example, gene 4, whose consecutive expression values even though being dissimilar reside in the same row, $r = 1$, of cell ID(1,0,1) and ID(1,1,1) and so on. Thus, we again subdivide the cells into 2^k equal sub-cells, where $k = 4$, and $l = 2$. Now, we obtain the 16 grid cells as depicted in Figure 3.2 (c). We conclude from this figure that (i) no two consecutive time points fall in the same column and (ii) no two consecutive points with different expression values fall in the same row. Thus the feature condition is satisfied and cell division terminates. We thus obtain the row and column values for every gene profile across time points. For example the r values of gene 1 is 0,1,3,2 and its c values are 0,1,2,3 at $l = 2$. Similarly, the r and c values for the other genes may be obtained.

Let $ID(r, c, l)$ be a cell in row r and column c at level l in the region $A \times B$, with block width 2^{q-l}. For the top level, $l = 0$, the block width is $W = 2^q$. $ID(r, c, l)$ is defined below. For a particular $ID(r, c, l)$, the corresponding points of a profile that have coordinates (y, x), where y refers to the time point and x the corresponding expression level value, fall in column c and row r at level l and is given by the following equation.

$$ID(r,c,l) = \begin{cases} (x,y) : \frac{W}{2^l}(r) \leq x < \frac{W}{2^l}(r+1), \ \ \frac{W}{2^l}(c) \leq y < \ \frac{W}{2^l}(c+1) \\ (x,y) : 2^{q-l}(r) \leq x < 2^{q-l}(r+1), \\ \qquad\qquad 2^{q-l}(c) \ \ \leq y < 2^{q-l}(c+1) \end{cases} \tag{3.17}$$

where y and x are positive real numbers including zero, for $0 \leq r < 2^l, 0 \leq c < 2^l$. Figure 3.1 shows an example of constructing quadrilateral mesh from level $l = 0$ to level $l = 2$.

For a better understanding of how the grid approximation is done for gene

profiles, we put forward the following example. For simplicity we assume that the quadrilateral is a square, i.e., $A = B$. At $l = 0$, block width is $2^{q-l} = 2^q = 16$, for $q = 4$ and $a = b = 2^l = 1$ as seen in Figure 3.1 (a). We subdivide this cell into 2^k (where $k = 2, 4, 8, \cdots$) cells if the feature condition is not satisfied as explained before. Now, we obtain 4 cells on subdivision with $k = 2$ and level is incremented by 1 to obtain $l = 1$. Now, the width of a cell is $2^{q-l} = 2^{4-1} = 8$ and $A = 2^{q-l}a = 8 \times 2 = 16$ and $B = 2^{q-l}b = 8 \times 2 = 16$, where, $a = b = 2^l = 2$. This is shown in Figure 3.1 (b). Further cell division results in $l = 2, k = 4$ and number of cells as $2^4 = 16$. Then, the width of each cell becomes 4 (i.e., $2^{q-l} = 2^{4-2} = 4$). Therefore, $A = 2^{q-l}a = 4 \times 4 = 16$ and $B = 2^{q-l}b = 4 \times 4 = 16$, where, $a = b = 2^l = 4$. This process of cell division goes on iteratively till the feature condition is satisfied. Now, for finding the corresponding points (y, x) that fall in row r and column c at level l, we use either of the equations given in Equation 3.17. We assume that the top left corner of the grid is $(0, 0)$. We show the working of the equation next.

For, $l = 0$,

$$\frac{W}{2^l}(r) \leq x < \frac{W}{2^l}(r+1) \ \ and \ \ \frac{W}{2^l}(c) \leq y < \frac{W}{2^l}(c+1)$$

Initially, $W = 2^q = 2^4 = 16$, Then for $r = 0$, we have,

$$\frac{W}{2^l}(0) \leq x < \frac{W}{2^l}(0+1) = 0 \leq x < 16$$

For $c = 0$, we have,

$$\frac{W}{2^l}(0) \leq y < \frac{W}{2^l}(0+1) = 0 \leq y < 16$$

Now, for $l = 1$ and $r = 0$, we have,

$$\frac{W}{2^l}(0) \leq x < \frac{W}{2^l}(0+1) = 0 \leq x < 8$$

For $c = 0$, we have,

$$\frac{W}{2^l}(0) \leq y < \frac{W}{2^l}(0+1) = 0 \leq y < 8$$

for $r = 1$, we have,

$$\frac{W}{2^l}(0) \leq x < \frac{W}{2^l}(0+1) = 8 \leq x < 16$$

For $c = 1$, we have,

$$\frac{W}{2^l}(0) \leq y < \frac{W}{2^l}(0+1) = 8 \leq y < 16$$

Similarly, the (y, x) values may be computed in level, $l = 2, 3,$ and so on Same results are obtained if we use the second equation of Equation 3 17 Thus, for each (y, x) point of a gene profile, we obtain a string of r and c values For example, from the Figure 3 2 (c), the r values of gene 1 is 0,1,3,2 and its c values are 0,1,2,3 at $l = 2$ For gene 2, r values are 1,2,1,1 and c values are 0,1,2,3 For gene 3, r values are 2,1,2,3 and c values are 0,1,2,3 For gene 4, r values are 3,2,3,1 and c values are 0,1,2,3

3.5.3 Computation of Dissimilarity

Once cell division terminates, each cell has a unique ID, i e , $ID(r, c, l)$ Since each point of the curve of every profile now resides in a separate cell, the ID of the cell is the ID of each point During comparison, every pair of points (one from each profile) always lies in the same column So, the contribution of column value and level information is null and only the row information is considered which reflects the maximum variability For every point in a profile, its corresponding r value is obtained For every gene, the median of the row values (r) is calculated across time points (conditions) For every gene, the difference of the row value of each point from the median is computed If the difference obtained is negative, the point is down-regulated, a positive value indicates up-regulation and 0 indicates equilibrium from the median This regulation information helps in finding the coherency between two genes As an example we take the gene profile plotted in Figure 3 3 has the r values as 2, 1, 2, 3 The median of these r values for this gene profile is 2 Next, the difference of the r values of each time point from the median of this gene will result in the difference value as 0, 1, 0, -1 From now on, in this chapter time point and point will refer to the same thing The median (m_i) of the r values for each gene i is computed For each gene, i, the difference of the r values of each point, x, from its respective median m_i is computed as shown below Here, x represents a particular time point (or condition) of a gene

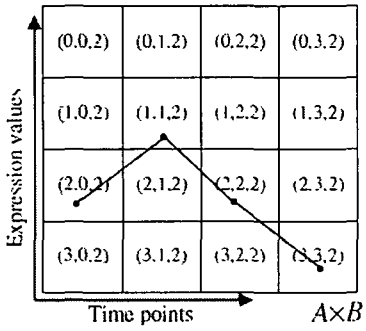


Figure 3.3: Example gene profile in level 1.

and the value of x varies across time points for every gene under consideration.

$$Diff_{xi} = m_i - r_{xi}, \quad x = 1, 2, 3, \cdots, T \tag{3.18}$$

where, T is the number of time points (conditions) and r_{xi} is the row value for the x^{th} time point of the i^{th} gene.
Note that $Diff_{xi}$ is actually a pattern of length T for the i^{th} gene.

For a pair of genes, the difference between their individual $Diff_{xi}$ is calculated and summed. This gives us the dissimilarity measure for pairs of genes. The dissimilarity measure $DBK(i, j)$ for gene i and gene j is computed as follows:

$$DBK(i,j) = \sqrt{\sum_{x=1}^{T}(Diff_{xi} - Diff_{xj})^2}. \tag{3.19}$$

Coherent Gene Identification by DBK Measure

Coherent gene identification is very important for gene expression clustering. Coherent genes are those genes which follow a similar trend. Two genes having the same trend will be highly coherent. For similarity measure the value of the similarity between two genes with same trend should be 1 and for dissimilarity measure, the dissimilarity between them should be 0.

Our proposed DBK can identify coherent genes based on the shape or trend information. This can be illustrated by an example and the values of the example

gene profiles are given in Table 3 2 and their normalized values (mean 0 and standard deviation 1) are given in Table 3.3. The normalized gene profiles are then

Table 3.2. Uncentered expression values of two example genes

Time	*gene1*	*gene2*	*gene3*
1	600	800	450
2	200	400	50
3	300	500	100
4	100	300	250
5	500	700	600

Table 3.3: Centered expression values of two example genes

Time	*gene1*	*gene2*	*gene3*
1	1.36	2.31	0.64
2	-0.55	0.41	-1.26
3	-0.07	0.88	-1.03
4	-1.03	-0.07	-0.31
5	0.88	1.84	1.36

plotted as a two-dimensional curve with time points in the horizontal direction and expression values in the vertical direction as given in Figure 3.4. The grid approximation is then done as explained before to obtain the grid approximation as shown in Figure 3.4 at level, $l = 3$. The row values (r) of the three genes are:

$$r_{gene1} = 0 \quad 8 \quad 6 \quad 10 \quad 2$$

$$r_{gene2} = 4 \quad 12 \quad 10 \quad 14 \quad 6$$

$$r_{gene3} = 7 \quad 15 \quad 14 \quad 11 \quad 4$$

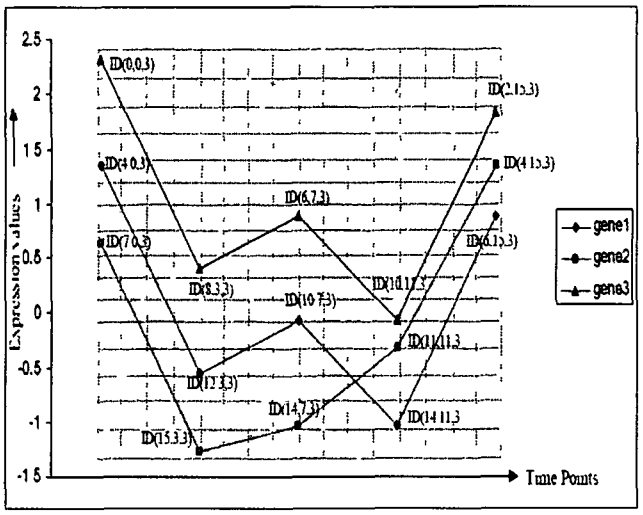


Figure 3.4: The grid approximation and the $ID(r, c, l)$ values for each point of the 3 gene profiles

The median (m) of the row values are:

$$m_{gene1} = 6 \quad m_{gene2} = 10 \quad m_{gene3} = 11$$

Therefore the $Diff$ values of the three genes are:

$$Diff_{gene1} = 6 \quad -2 \quad 0 \quad -4 \quad 4$$

$$Diff_{gene2} = 6 \quad -2 \quad 0 \quad -4 \quad 4$$

$$Diff_{gene3} = 4 \quad -4 \quad -3 \quad 0 \quad 7$$

Now,

$$DBK(gene1, gene2) =$$
$$\sqrt{(6-6)^2 + (-2-(-2))^2 + 0 + (-4-(-4))^2 + (4-4)^2} = 0$$

$$DBK(gene1, gene3) =$$
$$\sqrt{(6-4)^2 + (-2-(-4))^2 + (0-(-3))^2 + (-4-0)^2 + (4-7)^2} = 6.48$$

$$DBK(gene2, gene3) =$$
$$\sqrt{(6-4)^2 + (-2-(-4))^2 + (0-(-3))^2 + (-4-0)^2 + (4-7)^2} = 6.48$$

We notice from Table 3 2, that *gene*1 and *gene*2 have the same trend values only that the magnitude of the values of *gene*2 have been increased by 200 However, both of them have the same trend or shape This is reflected in the value of DBK where we see that both *gene*1 and *gene*2 have the DBK value of 0 Also, the value of DBK for *gene*1 and *gene*3 is same as that of *gene*2 and *gene*3 Since, *gene*1 and *gene*2 have the same trend so their coherency is high and also their distance from the third gene is same respectively The Pearson's correlation between *gene*1 and *gene*2 is 1 and similarity between *gene*1 and *gene*3 and *gene*2 and *gene*3 is same and is equal to 0 74 Therefore, we conclude that both DBK and Pearson's correlation can identify the coherent patterns based on the shape information

In another example, we consider three yeast sporulation expression profiles (YBR148W, YDR260C and YKL166C) from the study of [ESBB98] as given in Figure 3 5 We have replaced the missing values with the median We denote Pearson's correlation by *cor*, Euclidean distance by *euc* and our dissimilarity measure by *prop* We have $cor_{(YBR,YDR)} = 0\ 984$, $cor_{(YDR,YKL)} = 0\ 815$ and $cor_{(YBR,YKL)} = 0\ 851$ Now, $euc_{(YBR,YDR)} = 3\ 422$, $euc_{(YDR,YKL)} = 5\ 244$ and $euc_{(YBR,YKL)} = 8\ 552$ For our measure, $DBK_{(YBR,YDR)} = 3\ 606$, $DBK_{(YDR,YKL)} = 3\ 742$ and $DBK_{(YBR,YKL)} = 7$ We see that according to both Euclidean and our measure YKL166C is more similar to YDR260C than to YBR148W, a result that does not agree with the correlation coefficient result

3.5.4 Effectiveness of DBK

The proof that DBK satisfies the properties listed in Section 3 3 is given below

Property 1 For genes i and j, $D(i,j) \geq 0$, the square of $Diff_{xi}$ - $Diff_{xj}$, $x = 1, \ldots, T$, gives a positive number and moreover, we take the positive square root of the sum of $Diff_{xi}$ - $Diff_{xj}$, $x = 1, \ldots, T$ This results in always positive numbers Therefore, $DBK(i,j) \geq 0$

Property 2 Dissimilarity between a profile and itself must be zero $DBK(i,i) = \sqrt{\sum_{x=1}^{T}(Diff_{xi} - Diff_{xi})^2} = 0$

Property 3 Dissimilarity between gene i and gene j is equal to the distance between gene j and gene i

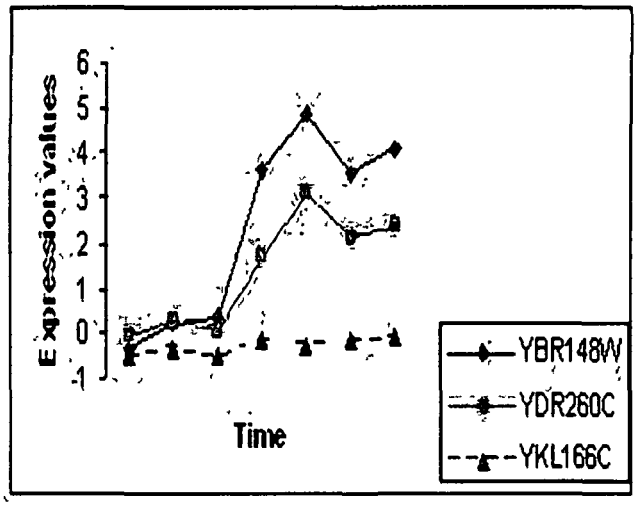


Figure 3.5: Three expression profiles from yeast sporulation [ESBB98] data set

For identical genes. Assume, $DBK(i,j) = 0$. Then according to Property 2, i and j are identical genes and $DBK(j,i) = 0$ Therefore, $DBK(i,j) = DBK(j,i) = 0$ for identical genes.

For non-identical genes. Again assume, $DBK(i,j) = \beta$. According to Equation 3 19, $DBK(i,j) = \sqrt{\sum_{x=1}^{T}(Diff_{xi} - Diff_{xj})^2}$ and $DBK(j,i) = \sqrt{\sum_{x=1}^{T}(Diff_{xj} - Diff_{xi})^2}$ Let $(Diff_{1i} - Diff_{1j}) = \gamma$ for the 1^{st} time point, then $(Diff_{1j} - Diff_{1i}) = -\gamma$. But, since while calculating DBK, we take the square of the terms both the terms become equal Similarly, for rest of the terms, we get $(Diff_{xi} - Diff_{xj})^2 = (Diff_{xj} - Diff_{xi})^2$ for $x = 2, 3, \cdots, T-1$ time points Therefore, we can conclude that $DBK(i,j) = DBK(j,i) = \beta$

Property 4: DBK satisfies the triangle inequality property, i.e., $DBK(E_x, F_y) \leq DBK(E_x, H_z) + d(H_z, F_y)$, where E_x, F_y, H_z are gene profiles

Assume, E_x, F_y, H_z are three row sequences and $E_x = e_1\ e_2\ \cdots\ e_T$, $F_y = f_1\ f_2\ \cdots\ f_T$ and $H_z = h_1\ h_2\ \cdots\ h_T$

We next transform them to the $Diff$ values with medians, $m_{E_x}, m_{F_y}, m_{H_z}$, where m_{E_x}, m_{F_y} and m_{H_z} are the medians of of E_x, F_y and H_z respectively

$Diff_{E_x} = \Delta e_1\ \Delta e_2\ \cdots\ \Delta e_T$, where $\Delta e_i = m_{E_x} - e_i$

$Diff_{F_y} = \Delta f_1\ \Delta f_2\ \cdots\ \Delta f_T$, where $\Delta f_i = m_{F_y} - f_i$

$Diff_{H_z} = \Delta h_1\ \Delta h_2 \qquad \Delta h_T$, where $\Delta h_i = m_{H_z} - h_i$

Now, the calculation of $DBK(E_x, F_y)$ reduces to the Euclidean distance calculation, i e ,

$$DBK(E_x, F_y) = \sqrt{\sum_{i=1}^{T} (\Delta e_i - \Delta f_i)^2}$$

Similarly, $DBK(E_x, H_z)$ and $DBK(H_z, F_y)$ are calculated Now, it is obvious that the triangle inequality property of DBK is satisfied based on the Euclidean space [2]

Since, DBK satisfies all the metric measure properties, we can say that is a metric measure

To establish that our dissimilarity measure satisfies the properties listed above, we take data from the work of [CDE+98] and [CCW+98] To validate the properties, we take the genes YJL157C, YKL185W, YAL008W from the dataset of [CCW+98] and two other synthetic profiles, viz , Series3 and Series4 Series 3 and Series 4 have similar expression levels as that of YAL008W except at the 4^{th} time point, which is replaced by two random values We will refer to the profiles of YJL157C, YKL185W Series3, Series4 and YAL008W as profiles 1, 2, 3, 4, and 5, respectively The expression plot is as shown in Figure 3 6 When we compute dissimilarities between pairs of profiles, the values are positive For example, dissimilarity between profile 1 and 2 is 5 292, profile 1 and 3 is 5 568, profile 1 and 4 is 5 477, profile 1 and 5 is 12 288, profile 2 and 3 is 7, profile 2 and 4 is 6 633, profile 2 and 5 is 14 457, profile 3 and 4 is 1, profile 3 and 5 is 10 and profile 4 and 5 is 11 These are non-negative numbers and hence the first property follows A detailed workout shows that the other properties are also satisfied The dissimilarity between profiles 3 and 4 is only 1 This is due to the fact that their expression values are identical except for the value at time point 30

We next compare DBK with three most commonly used proximity measures over gene expression data

[2] www ubicc org/files/pdf/UBICC_TS_final_Submission_192_192 pdf

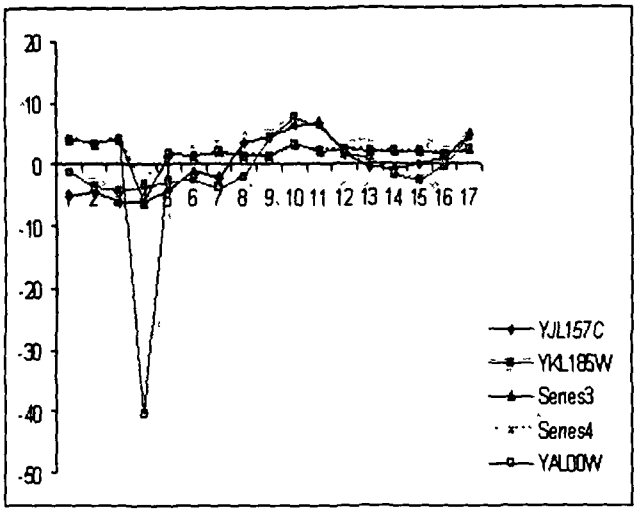


Figure 3.6: Time series plot of expression profiles for indicated genes: YJL157C (diamond), YKL185W (solid square), Series3 (triangle), Series4 (dotted line with cross sign) and YAL008W (hollow square)

3.5.5 Comparison between DBK and Euclidean measure

Euclidean measure is highly influenced by the magnitude of changes in the dataset. Although both Euclidean measure and DBK reflect the magnitude of the objects, Euclidean measure is concerned about the global information, and DBK is more concerned about the local information (since it uses the median of r values), which makes DBK more robust than Euclidean measure. An important advatage of DBK over Euclidean measure is that DBK retains the regulation (trend) information due to the use of the median and thus is capable of finding coherent genes from the dataset. This can be justified by again taking the example gene profiles of Table 3.3. We saw that the DBK value of the two coherent genes, *gene*1 and *gene*2 were 0. However, their Euclidean distance is 2.1377 even on the centered data. This shows that Euclidean distance doesnot take the shape (trend) information into account.

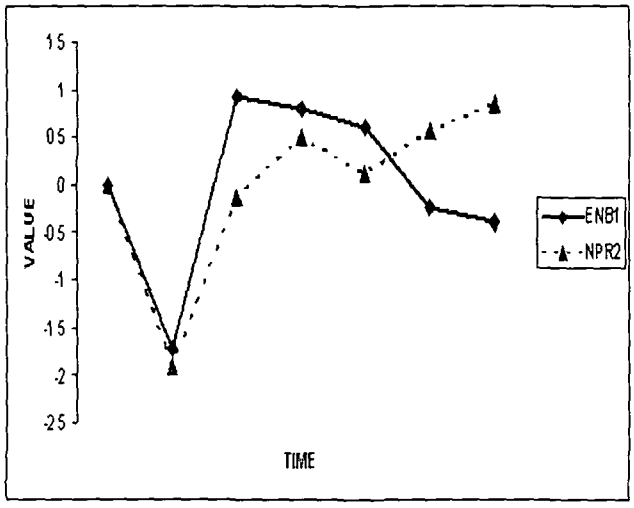


Figure 3.7: Time series plot of ENB1 (triangle) and NPR2 (square)

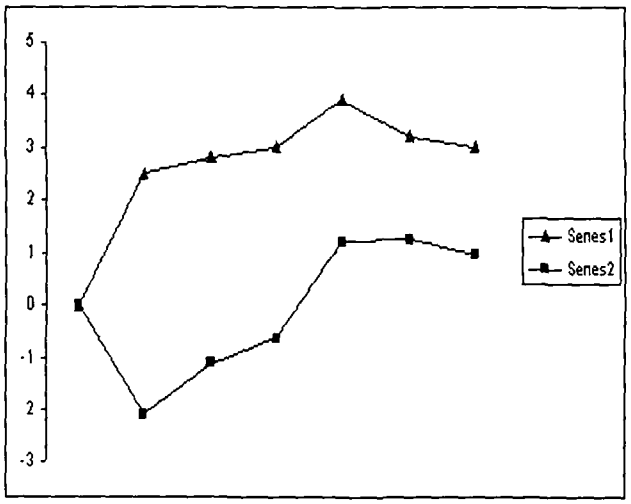


Figure 3.8. Time series plot of CAR2(Series1) and FYV4(Series2)

3.5.6 Falsely Correlated Time Series by Pearson's Correlation

During computation of Pearson's correlation, the mean of each profile is used whereas the DBK measure uses the median The disadvantage of using Pearson's correlation is that an aberrant value will shift the mean and may falsely reflect the similarity between two genes The advantage of DBK over Pearson's correlation is that due to the use of median value, an aberrant value does not affect the similarity between two genes significantly This observation can be illustrated with the help of an example The uncentered log ratio values of two genes, viz ,

Table 3 4 Uncentered log ratio values of two genes, viz , ENB1 and NPR2 from the time series dataset of [CDE+98]

Time	*ENBI*	*NPR2*
0 5	-0 76359	-4 05957
2	2 276659	-1 7788
5	2 137332	-0 97433
7	1 900334	-1 44114
9	0 932457	-0 87574
11	0 761866	-0 52328

ENB1 and NPR2 from the time series dataset of [CDE+98] is given in Table 3 4 Figure 3 7 is the time series plot for these two genes after centering Pearson's correlation for the two genes is 0 633 indicating strong correlation However, the graph of Figure 3 7 show that the genes are not related with respect to the trend (behaviour of time series points) The 'outlier' that results in the strong correlation is at 30 minutes, where both genes are down-regulated [Ste06] However, after the 30 minute time point, the behaviors of the two genes are completely different Our dissimilarity measure gives a value of 3 3166, which is sufficiently high showing that the two series are not similar Therefore, we see that Pearson's correlation is susceptible to outliers whereas our measure is robust

to outliers

3.5.7 False Correlation result by Spearman's Correlation

Spearman's correlation does not work directly on the expression values, instead it works on the ranked data However, the rank-based methods could mistakenly interpret a pattern since the use of rank causes information loss Two gene profiles may be dissimilar w r t their expression values but may be similar w r t their ranks Therefore, the result will falsely reflect the similarity between them This problem is not there in DBK, as it does not work on ranked data In fact the discretized value (r) on which DBK works has been discretized with minimum information loss thereby retaining the shape of patterns

The genes CAR2 and FYV4 of dataset [CDE+98] are shown in Figure 3 8 These two genes have similar shape profiles but one is up-regulated and the other is down-regulated The genes have a strong positive correlation (Spearman's correlation = 0 91) even though the expressions are very different The Euclidean distance (= 8 5) is large reflecting the difference Our measure also gives a sufficiently large value (=4 455) indicating the substantial difference between the profiles In this case, both Euclidean distance as well as our measure are probably good measures Table 3 1 gives a comparison of our measure with Euclidean distance, Pearson's and Spearman's correlation We note that our measure works best when used over mean zero and standard deviation 1 centered data

3.6 Performance Evaluation

In this thesis, we have used the following real-life data sets

- *Dataset 1* In [CCW+98], Cho *et al* used the temperature sensitive mutant strain CDC28-13 to produce a synchronized cell culture of the *Saccharomyces cerevisiae* from which 17 samples were taken at 10 minute intervals and hybridized to Affymetrix chips The final data is publicly available at *http //yscdp stanford edu/yeast_cell_cycle/full _data html* Cho's dataset is widely available and has functional classification that allows validation of

clustering results. This dataset contains 6218 genes at 17 time points.

- *Dataset 2*: Out of the full Dataset 1, a subset of 384 genes have been obtained from *http* : *//faculty.washington.edu/kayee/cluster*.

- *Dataset 3*: In [DI97], the authors use DNA microarrays to study the temporal gene expression of 6089 genes in *Saccharomyces cerevisiae* during the metabolic shift from fermentation to respiration. Expression levels were measured at seven time points during the diauxic shift (the two growth phases of a microorganism in batch culture as it metabolizes a mixture of two sugars). The full data set can be downloaded from the Gene Expression Omnibus website, *http://www.ncbi.nlm.nih.gov/geo/query*.

- *Dataset 4*: The dataset used is from the study of [WFM+98] where the authors study the relationship among gene expression patterns of genes involved in the rat Central Nervous System (CNS), measured during the development of the rat's CNS. Gene expression patterns for 112 genes were measured at nine different developmental time points. This yields a 112×9 matrix of gene expression data. This data set can be downloaded from *http* : *//faculty.washington.edu/kayee/cluster*.

- *Dataset 5*: The dataset used is from the study of [RWDF00] where the authors study the relationship among gene expression patterns of genes of Arabidopsis Thaliana. Gene expression patterns for 138 genes were measured at eight different time points. This yields a 138×8 matrix of gene expression data. This data set can be downloaded from *http* : *//homes.esat.kuleuven.be/ thijs/Work/Clustering.html*.

- *Dataset 6*: The dataset describes the response of human fibroblasts to serum on cDNA microarrays in order to study growth control and cell cycle progression. These data were obtained from the study of [IER+99]. Primary cultured fibroblasts from human neonatal foreskin are induced to enter a quiescent state by serum deprivation for 48 hours and then stimulate by addition of medium containing 10% FBS. DNA microarray hybridization is used to measure the temporal changes in mRNA levels of 8613 human genes.

The data set has 13 dimensions corresponding to 12 time points (0, 0.25, 0.5, 1, 2, 4, 6, 8, 12, 16, 20 and 24 hours) and one unsynchronized sample. In this thesis, we choose a subset of 517 genes whose expression changed substantially in response to serum. The detailed information about the data set can be found at the Web site: http://genome-www.stanford.edu/serum/.

- *Dataset 7*: The dataset used is from the study of [SSI+98] where the authors study the yeast cell cycle. There are a total of 698 genes at 72 conditions in the subset of this data obtained from the sample input files in Expander [SMKS03].

A brief overview of the datasets is given in Table 3.5. All the datasets are normalized to have mean 0 and standard deviation 1.

3.6.1 Results

We exhaustively tested our dissimilarity measure on all the datasets. We included the Euclidean distance, Pearson's correlation and our own DBK dissimilarity measure in both k-means [Ste06] and hierarchical clustering (UPGMA) [ESBB98] algorithms. Figure 3.9, Figure 3.10 and Figure 3.11 show the result when the full Dataset 1 is clustered using k-means for Euclidean distance, Pearson's correlation and our measure. The result when UPGMA clustering was applied on Dataset 1 using Euclidean distance, Pearson's correlation and our measure are shown in Figure 3.12, Figure 3.13 and Figure 3.14, respectively.

It is seen from Figure 3.9 and Figure 3.10 that at k=30, both Pearson's and Euclidean merge the single outlier into a cluster whereas our measure does not (cluster at 3^{rd} row 2^{nd} column of Figure 3.11). At higher k values, the outlier is separated for k-means clustering using Euclidean distance. However even for higher k values the outlier is not separated for k-means clustering using Pearson's correlation. As can be seen from Figure 3.12, Figure 3.13 and Figure 3.14, all major cluster patterns can be detected by our measure when hierarchical clustering is applied on 100% of the data. However, for Euclidean distance, the cluster at 3^{rd} row, 5^{th} column of Figure 3.12 is produced as a singleton cluster whereas using our measure it is clustered with patterns similar to it in shape (6^{th} row , 1^{st}

Table 3.5: Datasets used for evaluating the clustering algorithms introduced in this thesis

Serial No.	Dataset	No. of genes	No of conditions	Source
1	Yeast CDC28-13 [CCW+98]	6218	17	http://yscdp. stanford.edu/yeast_cell_cycle/full_data.html
2	Subset of Yeast Cell Cycle [CCW+98]	384	17	http://faculty. washington.edu/kayee/cluster
3	Yeast Diauxic Shift [DI97]	6089	7	http://www.ncbi.nlm.nih.gov/geo/query
4	Rat CNS [WFM+98]	112	9	http://faculty. washington.edu/kayee/cluster
5	Arabidopsis Thaliana [RWDF00]	138	8	http://homes.esat.kuleuven.be/thijs/Work/Clustering.html
6	Subset of Human Fibroblasts Serum [IER+99]	517	13	http://www.sciencemag.org/feature/data/984559.hsl
7	Yeast Cell Cycle [SSI+98]	698	72	Sample input files in Expander [SMKS03]

column) of Figure 3.14. The Pearson's correlation measure mixes up the patterns with low variation across time points in different clusters; moreover it also suffers from the single outlier problem as can be seen in more than one of the clusters in Figure 3.13. Our measure on the other hand separates the low variation clusters from the rest and can detect the single outliers as can be seen from the singleton clusters.

The results obtained when both k-means and hierarchical clustering algo-

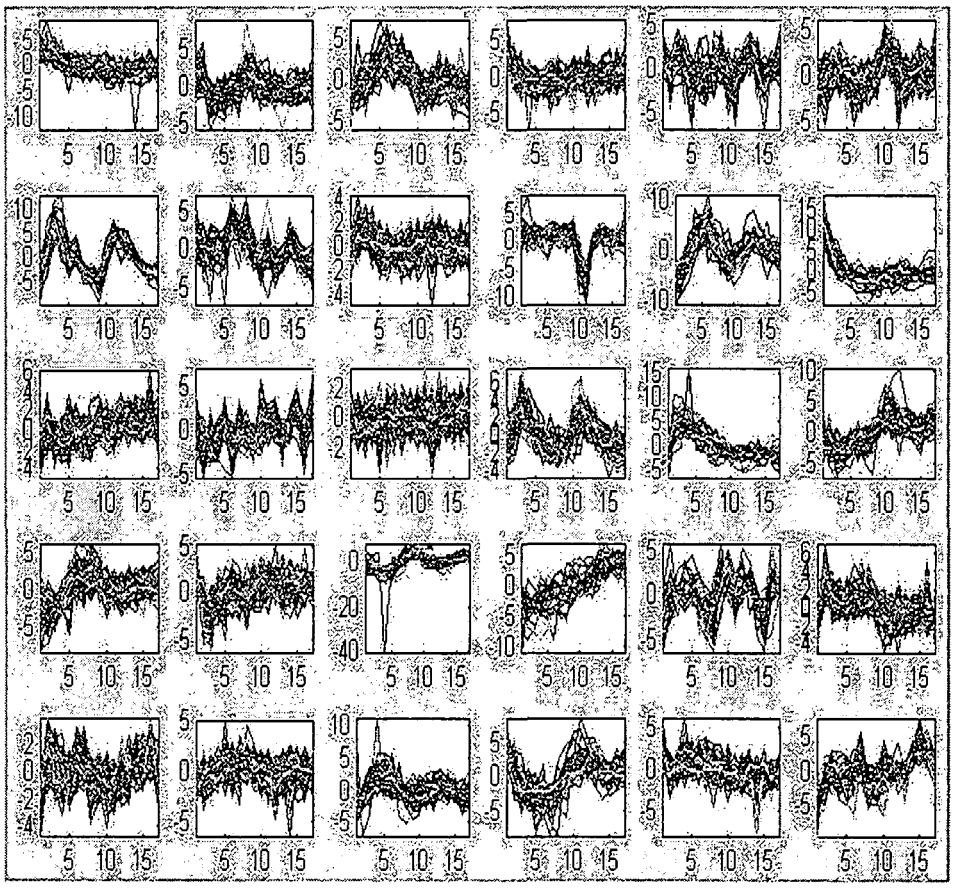


Figure 3.9: k-means clustering of profiles for Euclidean distance at k=30. The single outlier present at the 4^{th} row and 3^{rd} column is merged into a cluster

rithms are applied on Dataset 2 are shown in Figure 3.15, Figure 3.16, Figure 3.17 and Figure 3.18.

To assess the quality of DBK, we employed the Rand index as given next.

Rand Index

Rand index is a measure of the similarity between two clusters. The Rand index is defined as the numbers of pairs of objects that are either in the same group or in different groups in both partitions divided by the total number of pairs of objects. It can be calculated as follows.

The performance of a clustering process can be tested by comparing the

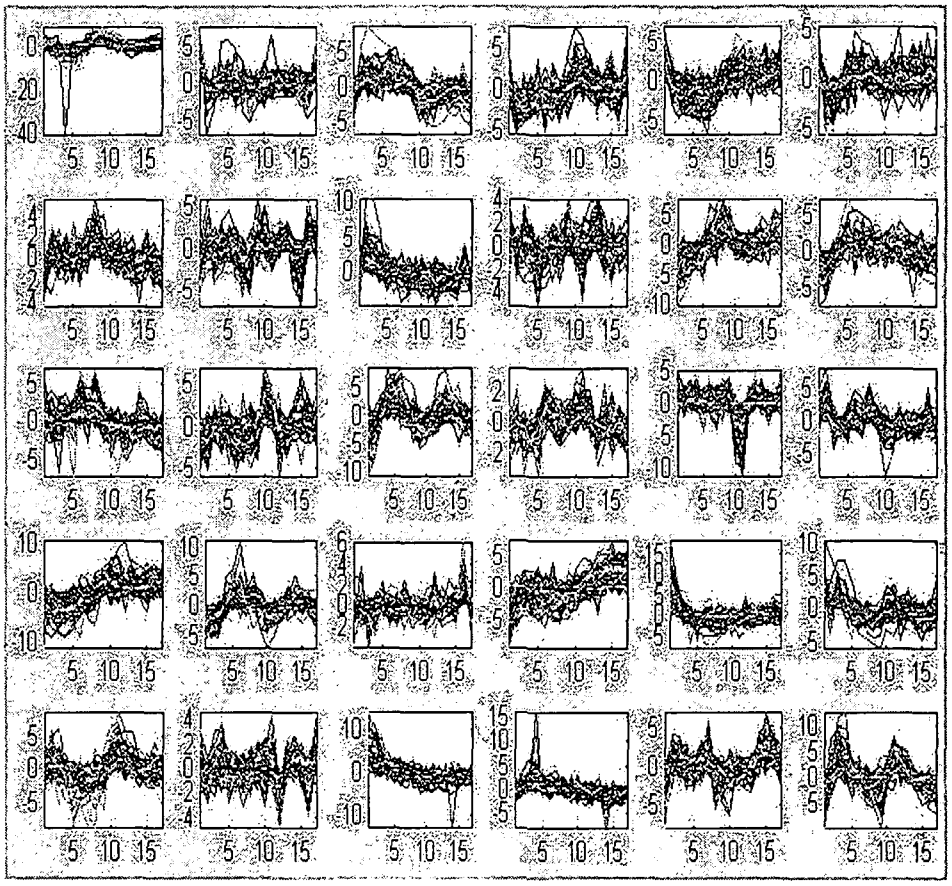


Figure 3.10: k-means clustering of profiles for.Pearson's correlation coefficient at k=30. The single outlier present at the 1^{st} row and 1^{st} column is merged into a cluster

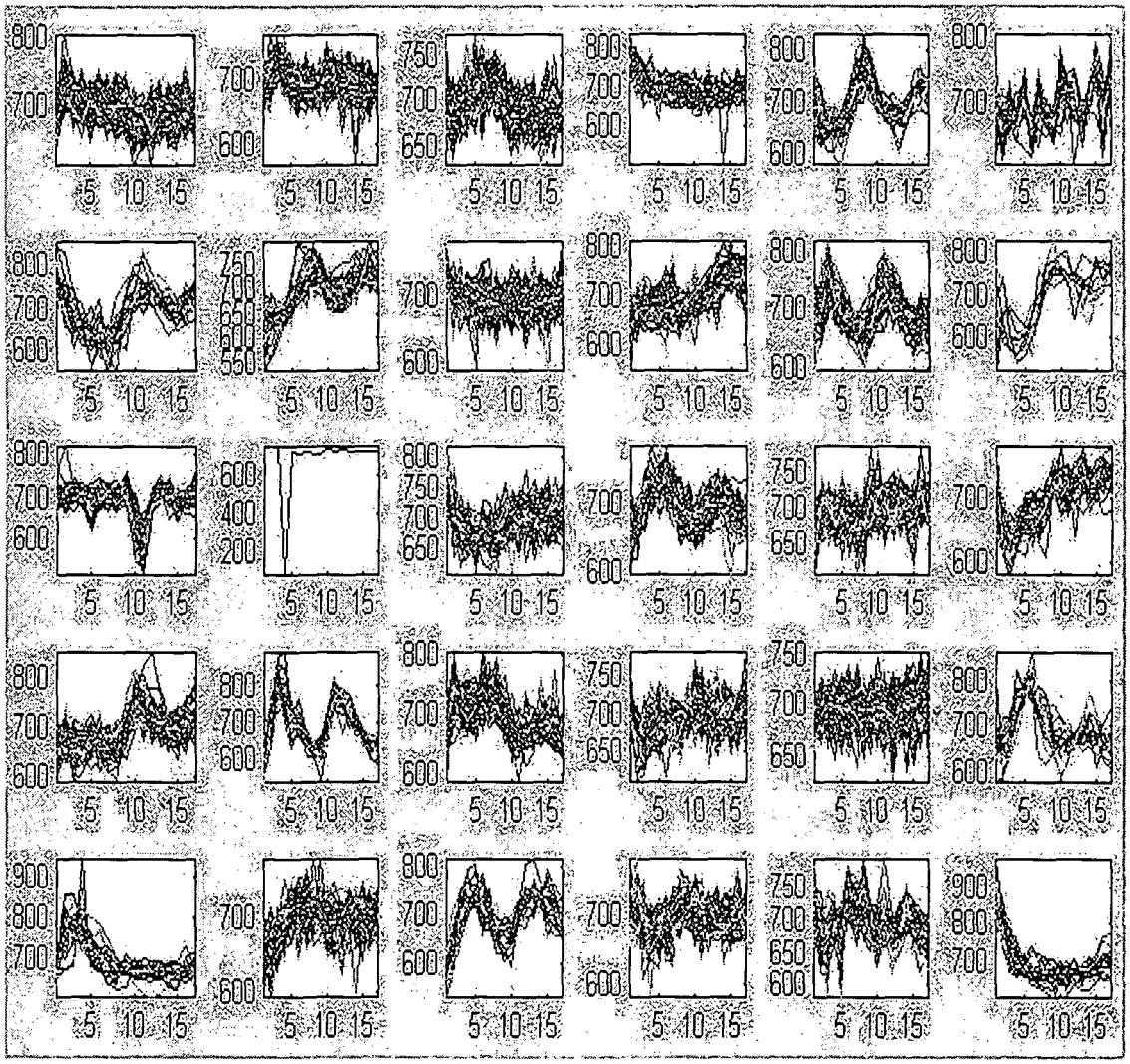


Figure 3.11: k-means clustering of profiles using DBK dissimilarity measure at k=30. Our measure does not merge the single outlier into a cluster as can be seen in the 3^{rd} row 2^{nd} column

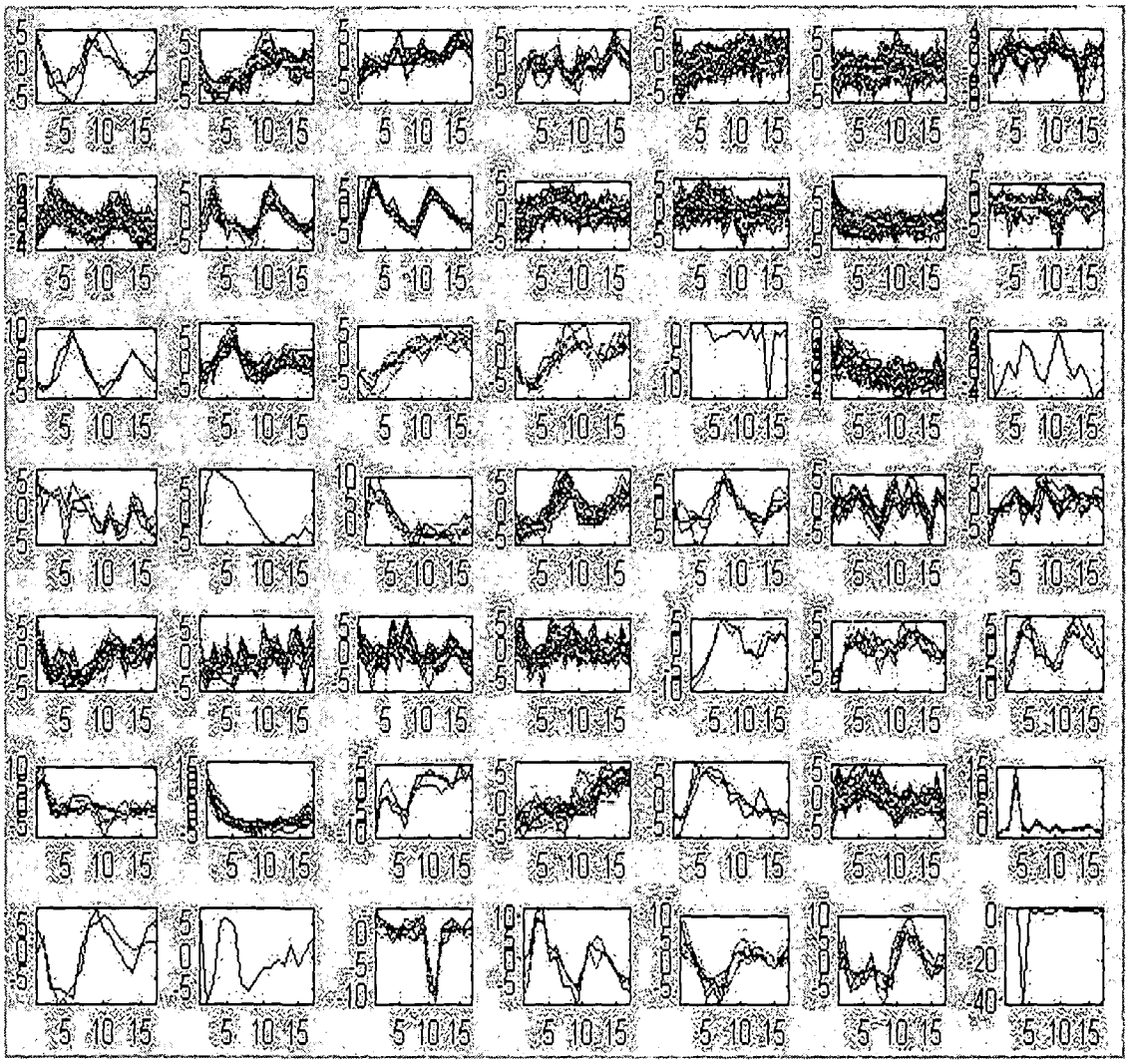


Figure 3.12: Hierarchical clustering of profiles for Euclidean distance at cutoff=49 and using complete linkage

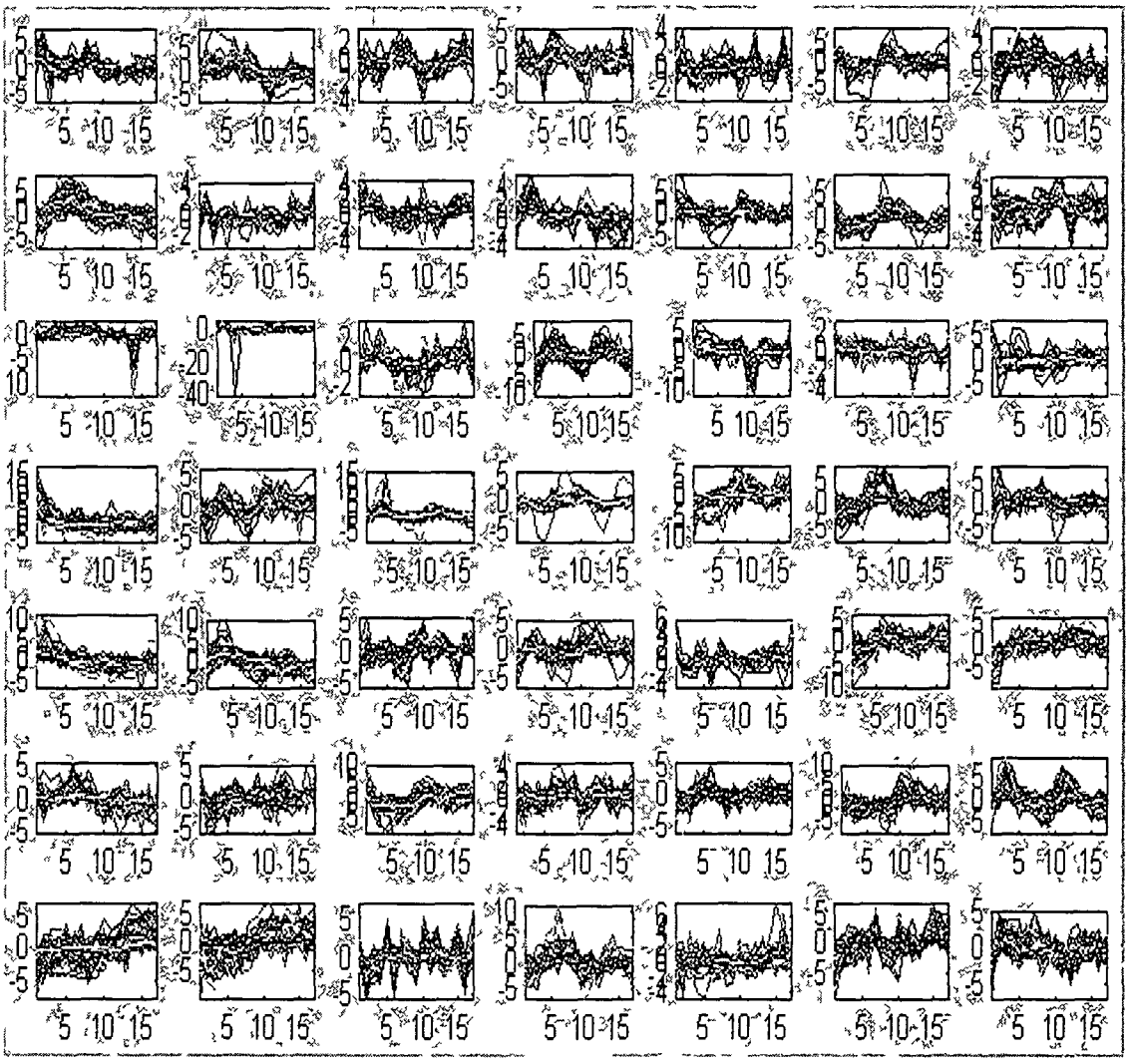


Figure 3 13 Hierarchical clustering of profiles for Pearson's correlation coefficient at cutoff=49 and using complete linkage

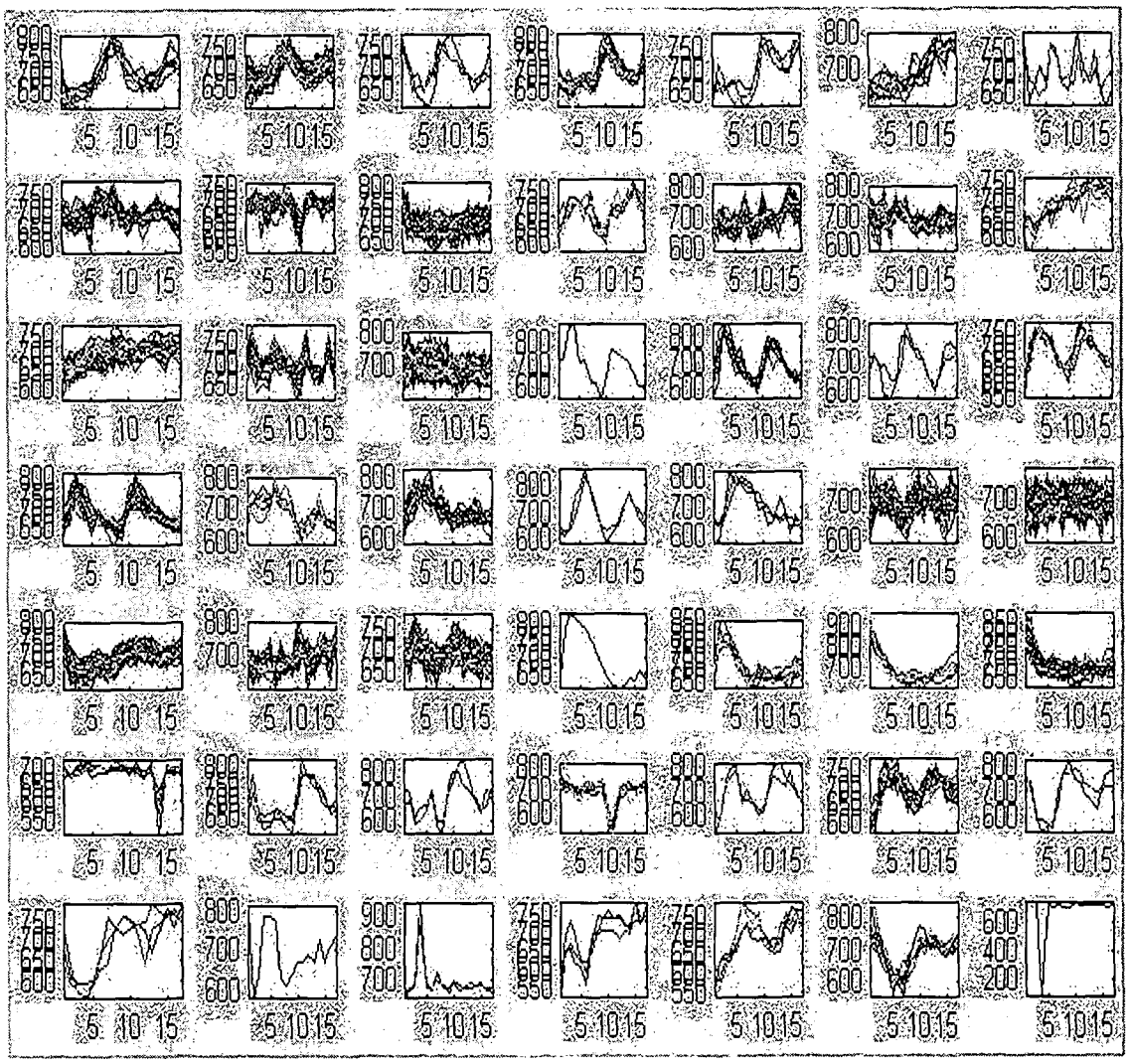


Figure 3.14: Hierarchical clustering of profiles for DBK measure at cutoff=49 and using complete linkage

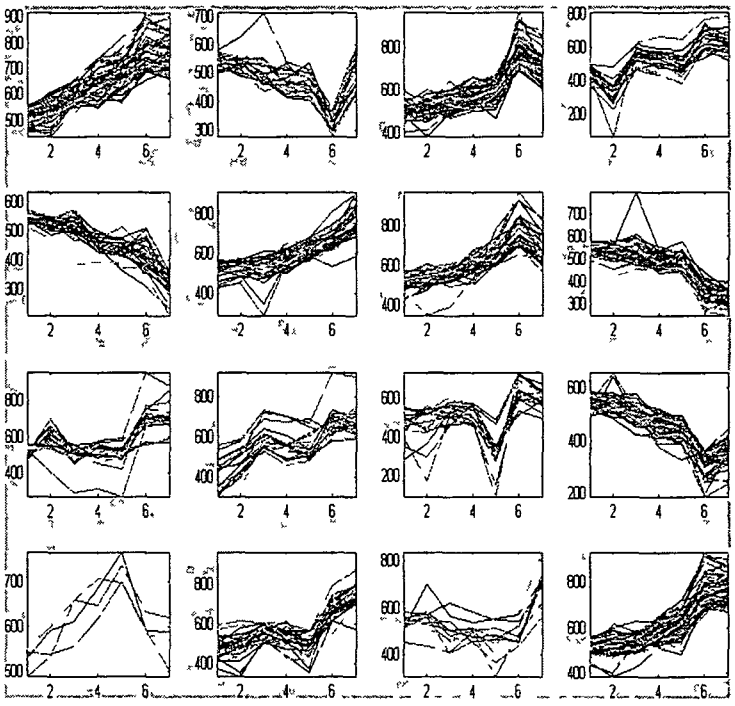


Figure 3.15 k-means clustering of profiles for Pearson's correlation coefficient at k=16

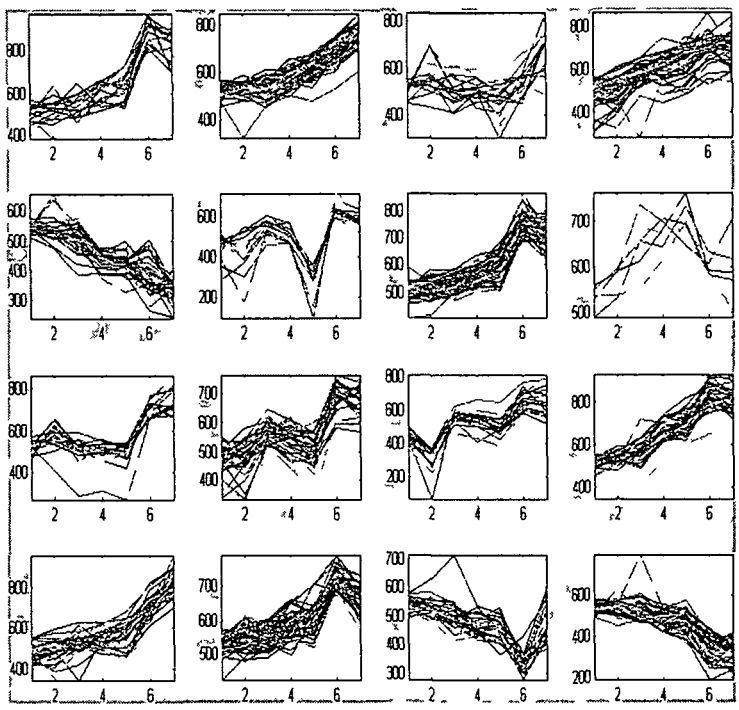


Figure 3 16 k-means clustering of profiles for DBK measure at k=16

clustering results with the ground truth of the cluster structure of the data set Given the clustering results $C = C_1 \quad C_p$, we can construct a $G \times G$ binary matrix C, where G is the number of genes, $C_{ij} = 1$ if g_i and g_j belong to the same cluster, and $C_{ij} = 0$ otherwise Similarly, we can build the binary matrix P for the ground truth $P = P_1, \quad , P_s$ The agreement between C and P can be disclosed via the following values

- a is the number of object pairs (g_i, g_j), where $C_{ij} = 1$ and $P_{ij} = 1$
- b is the number of object pairs (g_i, g_j), where $C_{ij} = 1$ and $P_{ij} = 0$
- c is the number of object pairs (g_i, g_j), where $C_{ij} = 0$ and $P_{ij} = 1$
- d is the number of object pairs (g_i, g_j), where $C_{ij} = 0$ and $P_{ij} = 0$

$$Rand\ index = \frac{a+d}{a+b+c+d}$$

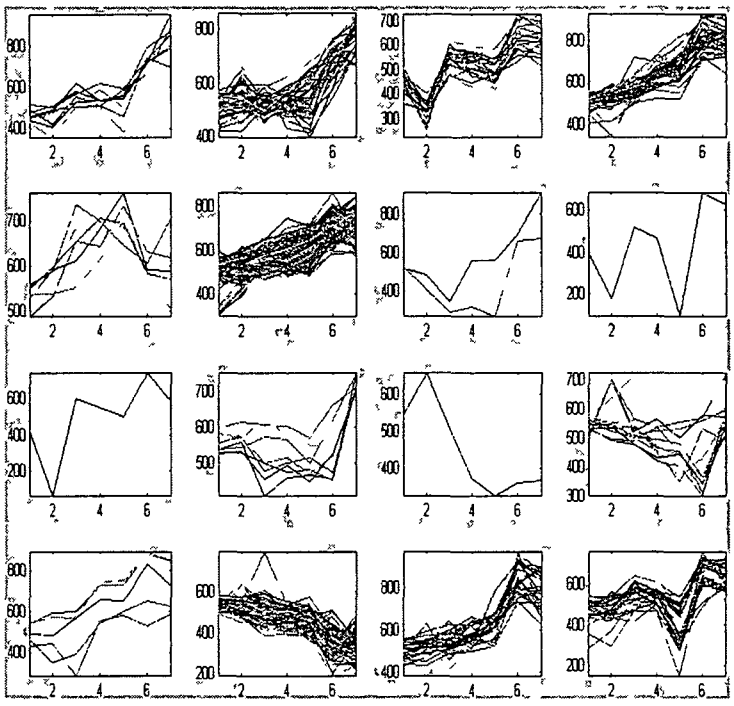


Figure 3 17 Hierarchical clustering of profiles for DBK measure at cutoff=16 and using complete linkage

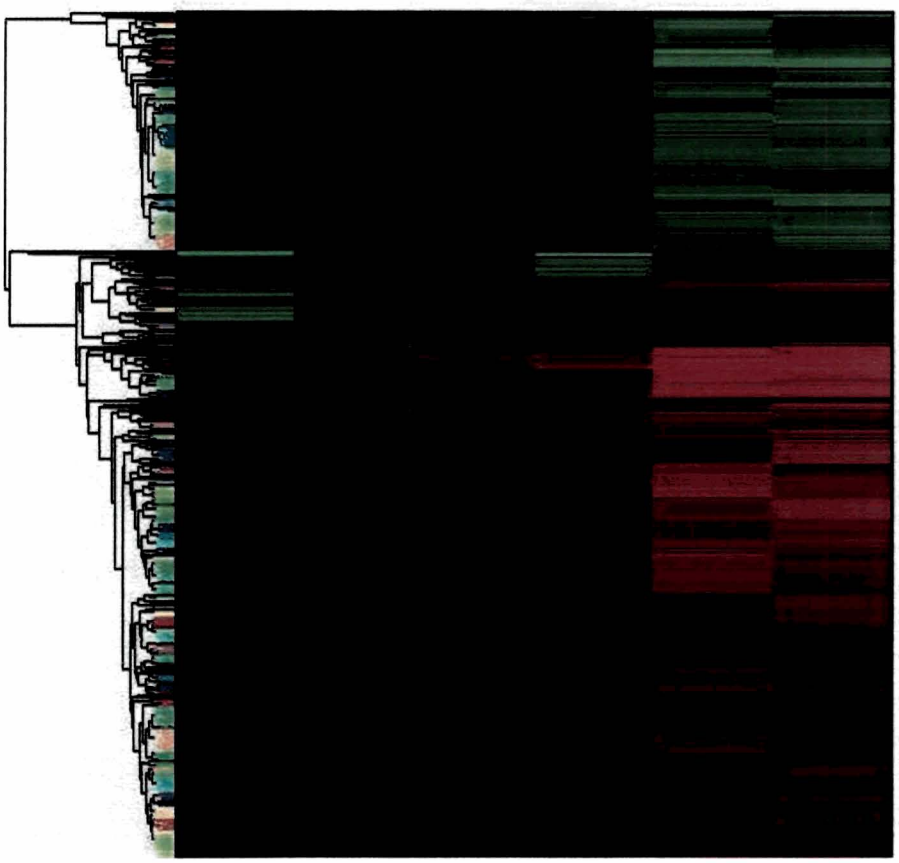


Figure 3.18: The dendrogram at cutoff=16

The Rand index lies between 0 and 1. The maximum value *i.e.*, 1 is achieved when both partitions, C and P, agree perfectly. To test the performance of the clustering algorithm, we compare clusters identified by our method with the 'ground truth' and with the results from k-means and UPGMA. With reference to the partitions in the Rand index, one partition is derived from the clustering results and the other partition is derived from the ground truth. The result of applying the Rand index on full Dataset 1 is shown in Table 3.6. It is observed from Table 3.6 that the DBK dissimilarity measure performs better for different values of the number of clusters, NoC, for Hierarchical clustering. However, for k-means it performs better than both Pearson's correlation and Euclidean distance for $NoC = 30$.

Table 3.6: Rand index on Yeast CDC28 data for various number of clusters (NoC)

Method	*Setting*	*NoC=30*	*NoC=42*	*NoC=45*	*NoC=49*
kmeans	Euclidean dist	0.520	0.622	0.671	0.739
kmeans	Pearson's corr	0.604	0.667	0.739	0.806
kmeans	DBK	0.614	0.622	0.698	0.802
Hierarchical	Euclidean dist	0.561	0.588	0.642	0.738
Hierarchical	Pearson's corr	0.601	0.623	0.640	0.697
Hierarchical	DBK	0.568	0.638	0.684	0.787

3.7 Discussion

An effective dissimilarity measure, DBK for clustering gene expression time series data is introduced in this chapter. The dissimilarity measure gives the shapes of the patterns of the gene expression data unlike Euclidean distance. In comparison to Pearson's correlation coefficient, our method is less susceptible to outliers as we use row values as well as the median while computing dissimilarity. Moreover, unlike Spearman's rank correlation, it also retains information about the regulation of the patterns. In the succeeding chapters, DBK is used in various

clustering techniques and found effective. The next chapter presents a graph based clustering technique which uses DBK as the proximity measure.

Chapter 4

A Graph-based Method for Clustering Gene Expression Data

This chapter presents a Graph based Clustering Algorithm, GCA, for clustering gene expression data. One of the main problems with clustering algorithms is the need to provide appropriate values for input parameters; this requires domain knowledge on the part of the user. This problem has been handled in GCA by using a dynamically calculated parameter for the clustering. GCA clusters genes based on a repulsion factor a gene has with other genes. Genes in a cluster have low repulsion whereas genes in different clusters have high repulsion. GCA uses the DBK dissimilarity measure introduced in the previous chapter. We also perform GCA using other commonly used proximity measures and compare results with those obtained using the DBK measure.

4.1 Introduction

A number of classical algorithms are commonly used for performing the task of clustering genes These include hierarchical algorithms (UPGMA) [ESBB98] and partitioning algorithms (k-means) [McQ67] as well as many novel approaches proposed recently Graph based clustering algorithms are suitable for data that do not follow a Gaussian or spherical distribution [FPSV07a] They can be used to detect clusters of varying shapes and sizes without the need to specify the number of clusters apriori Some popular partitioning based clustering algorithms such as k-means [McQ67] and SOM [Koh95] fail if data are distributed in the feature space along a non-smooth manifold [Jus06] Such algorithms assume a Gaussian or a spherical distribution for the data Moreover, they also require the number of clusters or some other input parameters to the algorithm Clustering algorithms based on graph theoretic approaches can alleviate the problems just mentioned Graph based algorithms represent the data by an undirected graph where each node represents an object in the feature space and each edge represents the proximity measure among the nodes it connects A cluster in this notion is defined to be a connected sub-graph, obtained according to criteria specific to each specific algorithm [FPSV07a] Algorithms based on this definition are capable of detecting clusters of various shapes and sizes, especially when the clusters are well separated [Jus06] Objects that are not connected form singleton clusters and are later on discarded as noise

In graph-based clustering algorithms, graphs are built as combinations of objects, features or both, as nodes and edges The graph is then partitioned by using graph theoretic algorithms Graph theoretic algorithms are also used for the problem of clustering cDNAs based on their oligo-nucleotide fingerprints ([HSL+99], [LL91])

4.2 Related Work

We now present a review of some selected graph based clustering algorithms

4.2.1 Fuzzy C-Means MST Clustering Algorithm (FMC)

The FMC algorithm [FPSV07b] starts by constructing a complete graph where each node is associated with an object and the edge weight gives the distance between two connected nodes. Then the minimum spanning tree (MST) of the graph is computed by using Prim's algorithm [HS78]. By removing all edges $> \lambda$ (a user defined threshold), a forest of trees is obtained. Each tree corresponds to a cluster. In this way, the method automatically groups nodes into clusters. For finding the optimal λ, the method uses a fuzzy c-means approach [Bez81b] and partitions the whole set of edges into two clusters according to their weights, one containing the edges of the MST with small weights while the other cluster contains edges removed from the MST.

4.2.2 Markov Clustering Algorithm (MCL)

The MCL algorithm [vD00] is based on the observation that if a group of nodes is strongly connected internally and has few connections (weakly connected) to the outside (both are properties of a cluster), a random walk starting at a node inside the group, is more likely to remain inside the group after a few steps than go outside. The MCL algorithm alternates between two phases: expansion and inflation, until a fixed point is reached. In expansion, the probability of a random walk of length k is computed by raising the matrix of the edge probabilities to the k^{th} power. In the inflation phase, re-normalization of the matrix is performed after raising each element to r where r is an input parameter. The matrix resulting from these two phases is used as input for the subsequent expansion process. The inflation phase reduces the smaller probabilities towards 0 and enhances the larger ones towards one. At the end, the clustering is determined by resulting probabilities which are significantly different from 0. Though there is no proof of convergence yet after a few tens of iterations, a fixed point is usually achieved.

4.2.3 Iterative Conductance Cutting Algorithm

The Iterative Conductance Cutting Algorithm (ICC) was proposed in [KVV00] and works in a divisive hierarchical manner. At first, the whole graph is con-

sidered a cluster and at each step, a cluster is split into two depending on the performance measure (known as *cluster conductance*) being $< \alpha$. The splitting process stops when there are no more clusters that can be divided based on α.

The cluster conductance compares the sum of the inter-cluster edges with the sum of the intra-cluster edges. Lower the values of conductance, better is the clustering result. The maximum value of conductance is one which is attained for singleton (one node) clusters or whole-graph clusters.

4.2.4 The Geometric MST Clustering Algorithm

The Geometric MST Clustering (GMC) Algorithm introduced in [Gae02] gives a solution to the problem of finding a suitable threshold for cutting the edges of the minimum spanning tree. For various possible thresholds, a performance measure is computed and the optimal one is chosen. For non-attributed graphs [Gae02] a geometric graph embedding is used to define the distance between nodes.

4.2.5 CLuster Identification via Connectivity Kernels

The CLuster Identification via Connectivity Kernels (CLICK) method ([SS00]) is suitable for subspace and high dimensional data clustering. (A subspace is a subset of a vector space that is itself a vector space. Here, the vector space refers to the high dimensional space of gene expression data. Subspace clustering is the task of detecting all clusters in all subspaces). CLICK is robust to outliers and does not make assumptions about the number or structure of clusters. Although CLICK does not need the number of clusters apriori, the algorithm may generate a large number of clusters because of the use of a homogeneity parameter.

Initially, the algorithm generates a fully connected weighted graph where the nodes represent objects and edges connect pairs of objects with weight assigned equal to proximity values among the objects. CLICK searches for highly connected components in the graph as clusters. CLICK makes the assumption that after standardization, pair-wise proximity values between objects (genes) are normally distributed. It, then recursively divides the graph in two using the minimum weight cut computations, until a certain kernel condition is met. The

division of the graph in two is done in such a manner that the sum of the weights of the discarded vertices is minimized. The clustering process in CLICK iterates by searching for the minimum cut in the graph and recursively splitting the dataset into a set of connected components. A partition with a single object is set apart as a singleton set. The kernel condition tests if the cluster formed by a given graph is highly coupled or not. If it is highly coupled the cluster is not subdivided further. CLICK builds a statistical estimator to evaluate the probability that the edges contained in a given graph belong to a single cluster. CLICK also uses two post-processing steps to refine cluster results. The *adoption* step handles singleton clusters and updates current clusters while the *merging* step iteratively merges two clusters having similarity greater than some predefined threshold.

The authors in [SS00] compared the clusters obtained using CLICK with those of SOM [Koh95] and Eisen's Hierarchical approach [ESBB98] and have found them to be better in terms of homogeneity and separation of clusters. However, there is little guarantee that CLICK does not generate unbalanced partitions (e.g., by mixing of noise in partitions with data objects).

4.2.6 Cluster Affinity Search Techniques (CAST)

Ben-Dor introduced the idea of *corrupted clique graphs* [BDSY99] and used the concept of a clique graph and divisive clustering in his algorithm, Cluster Affinity Search Techniques (CAST) [BDSY99]. A clique graph is an undirected graph formed by the union of disjoint complete sub-graphs where each clique represents a cluster. The model assumes that there is a *true biological partition of the genes into disjoint clusters based on the functionality of genes* [BDSY99]. The genes (objects) form sub-graphs or cliques where intra-clique genes are completely similar and inter-cluster genes are completely dissimilar.

CAST takes as input the pairwise similarities between genes and an affinity threshold, t. The algorithm searches through the clusters one at a time The currently searched cluster is denoted as C_{open}. Each gene g_i has an affinity value $a(g_i)$ w.r.t. C_{open} computed as $a(g_i) = \sum_{g_j \in C_{open}} S(g_i, g_j)$ where $S(g_i, g_j)$ denotes the similarity value between g_i and g_j A gene is said to have high affinity if $a(g_i) \geq t \mid C_{open} \mid$, else it has low affinity. CAST alternatively adds high

affinity genes and removes low affinity genes from the current cluster. When the process stabilizes, C_{open} is considered a complete cluster, and the process starts with another cluster. The process continues iteratively until all genes have been assigned to a cluster.

The affinity threshold, t, in CAST is actually the average of pairwise similarities within a cluster. It does not require a user-defined number of clusters and handles outliers efficiently. But, it faces difficulty in determining a good threshold value. In CAST, the size and number of clusters produced is directly affected by the fixed user-defined parameter, affinity threshold, t. Hence, prior domain knowledge of the data set is required. To overcome this problem, E-CAST [BPC02] calculates the threshold value dynamically based on similarity values of the objects that are yet to be clustered. The threshold is computed at the creation of each cluster.

4.3 Motivation

From the discussion above, we conclude that various clustering algorithms require different types of input parameters and resulting clusters are highly dependent on the values of the parameters. Graph based algorithms have a great advantage in that, they do not require the number of desired clusters as an input parameter and are robust to noise. However, the graph based algorithms are not totally free from input parameters.

In this chapter, we develop a graph-based clustering algorithm, GCA, that addresses the challenges presented by a gene expression dataset. It can find clusters from gene expression data without using any input parameters and is robust to outliers. It uses the DBK dissimilarity measure discussed Chapter 3. GCA requires an input parameter during cluster expansion, however, it is calculated dynamically.

4.4 An Effective Graph Based Clustering Algorithm (GCA)

Among the large number of genes encoded in microarray gene expression data, only a small fraction is pertinent to a certain task Identification of useful features (genes) is a challenging problem that needs to be addressed According to [LAA05], the selection of genes is important because it is impossible for biologists to examine the whole feature space at one time Moreover, taking into account irrelevant features results in unnecessary noise and computational cost Once the relevant features have been selected, the next step is to find an appropriate proximity measure for the gene expression data This chapter presents a graph based clustering algorithm (GCA) which uses the dissimilarity measure, DBK introduced in the previous chapter Our graph based clustering method works in three phases In the first phase, the gene expression data is normalized to mean 0 and standard deviation 1 Also the low variance and low entropy genes are filtered out The second phase computes the dissimilarities among genes using a grid based method and the third phase is dedicated to the task of clustering using a graph based approach

4.4.1 Clustering

Our proposed clustering method, GCA, is graph theoretic which exploits the concept of clique graph introduced by [BDSY99] However, GCA is different from CAST [BDSY99] in the following aspects

1 GCA uses 'repulsion' factor instead of 'affinity' used in CAST [BDSY99] to form cluster

2 GCA adds genes with low repulsion to a cluster, whereas CAST [BDSY99] adds high affinity genes to a cluster and deletes low affinity genes from a cluster

3 GCA uses a threshold, α, which is computed dynamically in each iteration to find the connectivity of an unclassified gene to a cluster, while CAST [BDSY99] uses a constant threshold value, t as the similarity cutoff

The dynamic computation of α makes GCA a parameter-less method The graph can be thought of as a disconnected graph with the nodes being the genes and the repulsion value of the genes is initially set to zero In this chapter, we will use the terms node and gene interchangeably to refer to the same thing A gene (node) with minimum pairwise dissimilarity value is selected and becomes the initiator of a cluster The repulsion value of the other genes (nodes) are updated w r t the cluster recently formed As the algorithm proceeds genes are being added to the cluster based on a connectedness condition defined later The following definitions and concepts provide the basis for the proposed clustering method

Repulsion

A gene cluster consists of similar genes while dissimilar genes belong to different clusters Thus, we can say that genes belonging to different clusters will repel each other i e , inter-cluster genes have more repulsion between them while the repulsion of intra-cluster genes is less A gene will belong to a cluster if its repulsion w r t the genes belonging to that particular cluster is least The repulsion of an unclustered gene to a cluster can be computed by the summation of the dissimilarity (distance) values of that gene w r t all the genes belonging to that particular cluster Thus, repulsion is the distance of an unclustered gene from the cluster under consideration and is defined next

Definition 4 1 The repulsion r of a node x from a cluster C is defined as

$$r(x) = \sum_{j=1, y_j \in C}^{|C|} DBK(x, y_j) \tag{4 1}$$

Connectivity of a Node to a Cluster

Connectivity of an unclustered node to a cluster is very important for clustering To check the connectivity of an unclustered node to a cluster, we need to consider the connectivity of the node in consideration to the nodes present in the particular cluster

The basic idea is that a node among a large number of unclustered nodes will be included into the highly connected region identified as a cluster if it's repulsion

w r t the cluster satisfies a connectivity condition

We know that a dissimilarity (distance) measure $d(i, j)$ defined between all pairs of nodes i and j will correspond to similar expressions for small distances and large distances means dissimilar expressions In clustering, clusters have a higher density of nodes than the surrounding background i e , clusters contain highly connected nodes This may be obtained by connecting a pair of nodes that are within a connectivity threshold, ξ, from each other For $\xi = 0$, every node is a cluster by itself Now, if ξ is gradually increased from zero, then the nodes from the region of highest density would get interconnected first to form tight clusters, next, the more dilute (sparser) clusters will form Later, as connections are made between nodes, they merge to form even larger clusters until eventually at some large ξ, all nodes will be interconnected Thus, ξ should have a value that will stop the over-dilution (sparcification) of a cluster For this, we compute the value of ξ for each cluster based on the unclustered nodes and the cardinality of the current cluster Here, we note that since we have to find the connectivity of an unclustered node to a cluster, we use the repulsion factor as discussed before If the repulsion of a node is within the connectivity threshold, ξ, then the node is included in the current cluster The value of ξ is updated dynamically, so that it reflects the change in the clustering after every insertion of a node to the current cluster Thus, every insertion of a node to the current cluster results in the change of the value of α which is a deciding factor in the process of clustering

Definition 4 2 The connectivity threshold, ξ, of a cluster C is defined as the product of the threshold α and cardinality of cluster C

$$\xi = \alpha \mid C \mid \tag{4 2}$$

During expansion of the cluster, the threshold α is calculated dynamically based on the number of unclassified genes, U_n i e ,

$$U_n = D_G - (C_o \cup C_1 \cup C_2 \cup \quad \cup C_n) \tag{4 3}$$

where, D_G is the total set of all genes, C_i is the i^{th} cluster

The parameter α is based on dissimilarity values of the nodes (genes) yet to

be clustered and the number of pairs of genes yet to be clustered

$$\alpha = \frac{\sum_{i,j \in U_n} DBK(i,j)}{(\mid U_n \mid \times (\mid U_n \mid -1))/2} \tag{4 4}$$

The dynamically calculated value of α reflects the overall dissimilarity of the unclustered genes and is further used in the calculation of ξ as discussed before

Definition 4 3 A node is said to be connected to a cluster C if its repulsion from the cluster is less than the connectivity, i e ,

$$r(x) \leq \xi \tag{4 5}$$

where $r(x)$ is the repulsion of node x from cluster C

In the clustering process first the repulsion of all of the genes is set to zero Clustering starts by selecting the gene l from the set of unclassified genes with the least dissimilarity value with its pair as given in Figure 4 1 The function module $find\_min\_DBK()$ given in Figure 4 2 finds the gene with minimum dissimilarity The function $find\_DBK(x, y)$ calculates the DBK distance between genes x and y As given in Figure 4 1, the gene l is then selected as the seed for cluster expansion and is sent to the *Cluster_expand*() module reported in Figure 4 3 l is assigned to a cluster $C_{cluster\_id}$ and the cardinality $\mid C_{cluster\_id} \mid$ is found as in Figure 4 4 The repulsion of all the unclassified genes is updated with respect to the elements in the current cluster $C_{cluster\_id}$ as in Figure 4 5 The connectivity con_t and threshold *alpha* (as in Figure 4 6) are calculated using Equation 4 2 and Equation 4 4, respectively, where $\xi = \alpha \mid C \mid$ In the algorithm, $\alpha = alpha$ and $\xi = con_t$ From the set of unclassified genes, we select a gene x that has minimum repulsion with the cluster $C_{cluster\_id}$ (as given in Figure 4 7) and whose repulsion value satisfies the connectedness condition given in Equation 4 5 Cluster expansion continues recursively with this selected gene x When no more genes can be added to a cluster, the cluster creation process starts with another unclassified gene, and the process continues till all the genes have been classified In the clustering process, we do not use any global threshold Our threshold value is calculated by the process and adapts dynamically to the number of unclassified genes

Cluster_creation()

// Initially, $U_n = D_G$ do

FOR x from 0 to G do

x.classified = 0; // initially all genes are unclassified

x.repulsion = 0; // initially all genes have a repulsion of 0

x.cluster_id = -1; // initially all genes have cluster_id = -1

End FOR

cluster_id = 0;

DO

$l = find\_min\_DBK()$;

$Cluster\_expand(l, cluster\_id)$;

cluster_id ++;

WHILE $l \neq -1$;

End

Figure 4.1: Algorithm for Cluster formation

```
find_min_DBK()
  min_DBK = 9999.99;
  min_DBK_gene = -1;
  FOR l from 0 to G do
    IF l.classified == 0 do
      FOR m from 0 to G do
        IF m.classified == 0 do
          x = find_DBK(l, m)
          IF x < min_DBK do
            min_DBK_gene = l;
            min_DBK = x;
          End IF
        End IF
      End FOR
    End IF
  End FOR
  return min_DBK_gene;
```

Figure 4.2: Algorithm for finding the gene with minimum dissimilarity

```
Cluster_expand(l, Cluster_id)
  IF l.classified == 1
    RETURN
  End IF
  l.classified == 1
  l.cluster_id = Cluster_id;
  //Update repulsion of all unclassified genes present in current cluster C
  FOR x from 0 to G do
    IF x.classified == 0 OR x. cluster_id == Cluster_id do
      x.repulsion += sum_DBK(x, Cluster_id));
    End IF
  End FOR
  alpha = calculate();
  con_t = alpha × total_gene_cluster(Cluster_id);
  x = find_minimum_repulsion();
  IF x.repulsion ≤ con_t AND x > −1 do
    Cluster_expand(x, Cluster_id);
  End IF
```

Figure 4.3: Algorithm for Cluster expansion

total_gene_cluster(*Cluster_id*)

```
count = 0;
FOR x from 0 to G do
  IF x.cluster_id == Cluster_id do
    count++;
  End IF
End FOR
return count;
```

Figure 4.4: Algorithm for computing the cardinality of a cluster

sum_DBK(x, *Cluster_id*)

```
sum = 0;
FOR y from 0 to G do
  IF y.cluster_id == Cluster_id do
    sum += find_DBK(x, y);
  End IF
End FOR
return sum;
```

Figure 4.5: Algorithm for computing the repulsion of a gene from a cluster

calculate()

```
count = 0;
FOR x from 0 to G do
  IF x.classified == 0 do
    count++;
  End IF
End FOR
total_unclassified_pairs = (count * (count - 1))/2;
total_unclassified_DBK = 0;
FOR x from 0 to G do
  IF x.classified == 0 do
    FOR y from 0 to G do
      IF y.classified == 0 do
        total_unclassified_DBK += find_DBK(x, y);
      End IF
    End FOR
  End IF
End FOR
total = (total_unclassified_DBK) / (total_unclassified_pairs);
return total;
```

Figure 4.6: Algorithm for computing α

find_minimum_replusion()

```
min_repulsion = 99999.99;
min_rep_gene = -1;
FOR x from 0 to G do
  IF x.classified == 0 do
    IF x.repulsion ≤ min_repulsion do
      min_repulsion = x.repulsion;
      min_rep_gene = x;
    End IF
  End IF
End FOR
return min_rep_gene;
```

Figure 4.7: Algorithm for finding the gene with minimum repulsion to a cluster

4.5 Performance Evaluation

We implemented the GCA method in C in Linux environment and evaluated it using the real-life datasets discussed in Chapter 3.

4.5.1 Results

We exhaustively test our graph based clustering algorithm on Dataset 1 taking 10%, 20%, 50%, 75% and 100% of the data. The exhaustive results are shown in Figure 4.8 and Figure 4.9 for 20% and 75% of the data from Dataset 1. From these detailed experiments we come to the following conclusions.

1. Most gene profiles are flat and do not significantly differ from others, and
2. Most genes have low variation over time.

Due to these reasons, the cluster space becomes cluttered with unimportant gene profiles making it difficult to extract real cluster structures. Therefore, we remove

genes with low variation as well as flat gene profiles All other authors do the same

When we apply the filtering process on Dataset 1, we obtain a reduced gene set consisting of 800 genes

We test our method on the previously published dataset of [CCW+98] to determine whether it can quickly and automatically find known patterns without using prior knowledge In [CCW+98], expression levels of 6,218 yeast ORFs were measured at 17 time points

Similar to [TSM+99], our method can also automatically and quickly (computation time 15 07 sec on a Pentium IV machine having 1GHz speed and 128MB RAM in Linux environment) extract the cell-cycle periodicity The trends of the clusters (a total of 30 clusters were detected) identified by our method are shown in Figure 4 10, with expression levels along y-axis and time points along x-axis The clusters (for example 0, 1, 7) contain genes with peak expression in late G1 phase are shown in Figure 4 10 and Figure 4 11 The genes agree well with those identified by visual inspection

Table 4 1 Rand index on Yeast CDC28 data for the clustering method GCA

Method	*Setting*	*Rand index*
GCA	Euclidean dist	0 789
GCA	Pearson's corr	0 778
GCA	DBK	0 806

4.5.2 Cluster Quality

To assess the quality of our method, we need an objective external criterion In order to validate our clustering result, we employed Rand index, Homogeneity, Silhouette index [JTZ03] and z-score as the measures of agreement

In this section, the reported Rand index is averaged over 20 repeated experiments According to [TSM+99], the total number of clusters contained in Dataset 1 is 30 The results found on comparing the GCA using Euclidean distance, Pear-

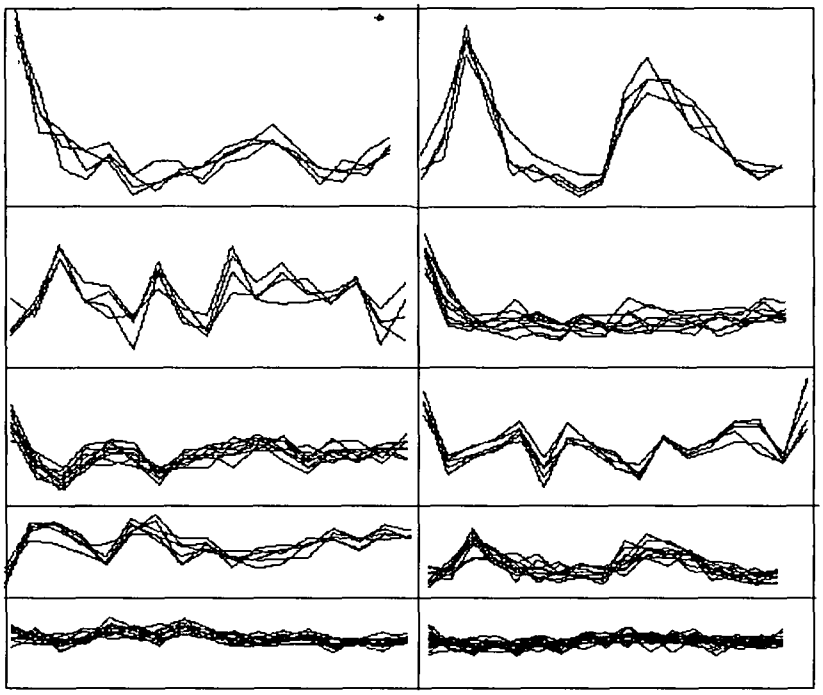


Figure 4.8: Some of the clusters obtained when our algorithm is used on 20% of Dataset 1

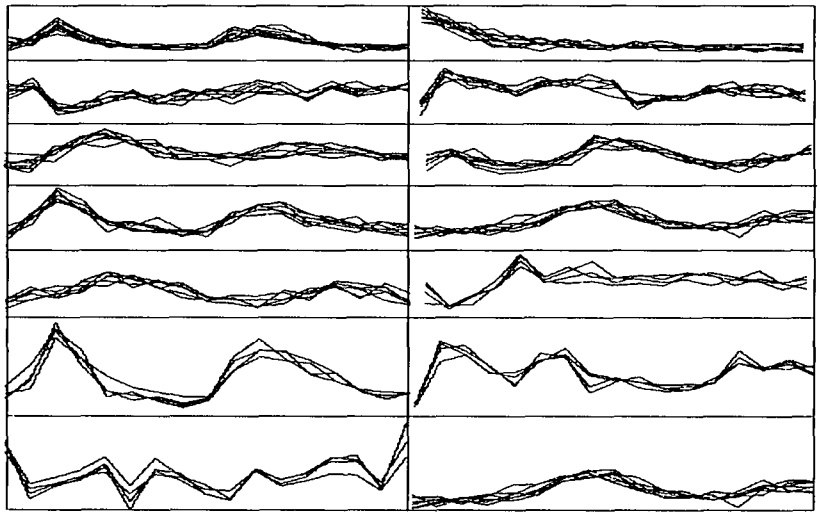


Figure 4.9: Some of the clusters obtained when our algorithm is used on 75% of Dataset 1

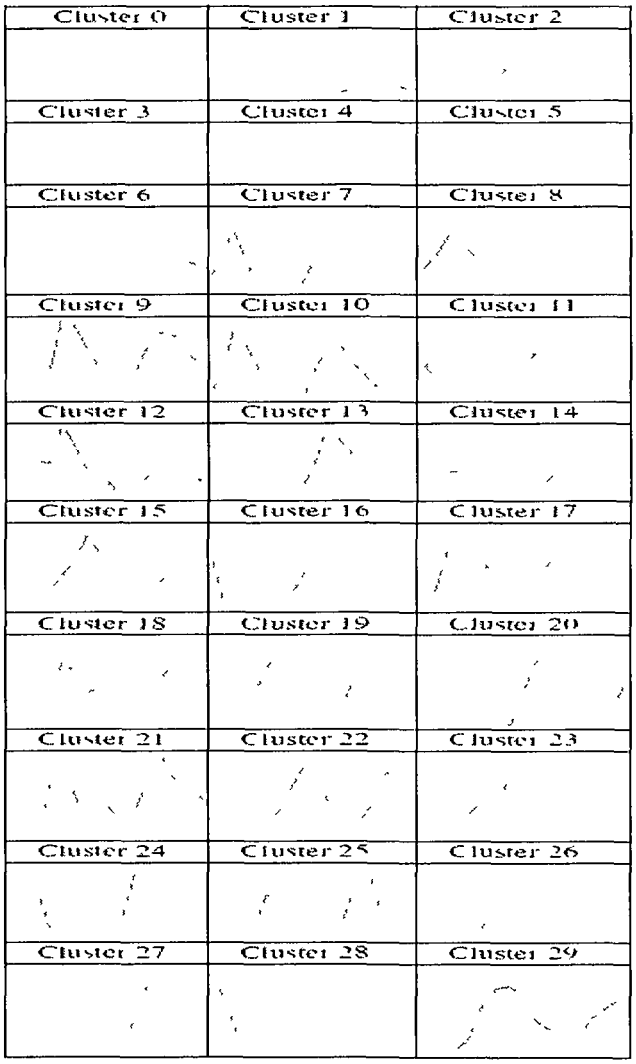


Figure 4.10: The trends of the clusters detected on Dataset 1

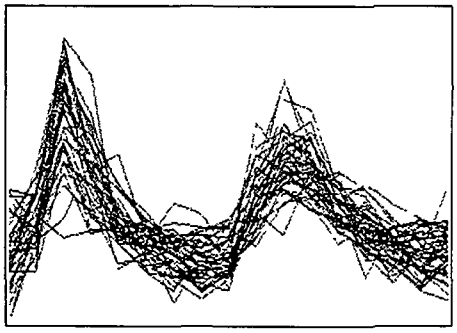


Figure 4.11: Cluster 1 consisting of 46 genes. The genes obtain peak expression in late G1 phase

son's correlation and DBK is given in Table 4.1. Clearly our measure performs better than Euclidean distance and Pearson's correlation when used with GCA. Due to non-availability of the functional classification for the other datasets, we can not compile the Rand index for them.

i. Cluster Homogeneity

Homogeneity measures the quality of clusters on the basis of the definition of a cluster: objects within a cluster are similar while objects in different clusters are dissimilar. Homogeneity measure used in this section is that of the overall average homogeneity used in [SMKS03]. It is calculated as follows.

a) Compute the average value of similarity between each gene g_i and the centroid of the cluster to which it has been assigned.

$$H(C_i) = \frac{1}{|C_i|} \sum_{g_i \in C_i} Similarity(g_i, g_i') \tag{4.6}$$

where g_i' is the centroid of C_i.

b) Calculate the average homogeneity for the clustering C weighted according to the size of the clusters as,

$$H_{avg} = \frac{1}{|G|} \sum_{C_i \in C} |C_i| H(C_i). \tag{4.7}$$

The homogeneity values for GCA and some other algorithms are reported in Table 4.2. It can be observed that the homogeneity value for GCA is the highest from which we can conclude that the coherence of the clusters produced by GCA are better than those produced by competing algorithms.

ii. Silhouette Index

Silhouette index [Rou87] is used to assess the quality of any clustering solution. This index reflects the compactness and separation of clusters. It is calculated as follows.

a) Compute $a(g_i)$, *i.e.*, the average distance of gene i to the the other genes of cluster A to which it belongs, *i.e.*, $g_i \in A$.

b) Compute $d(g_i, C_k)$ where $d(g_i, C_k)$ is the average distance of gene g_i from the genes of cluster C_k where $g_i \notin C_k$.

c) Compute $b(g_i)$, where $b(g_i) = min\{d(g_i, C)\}$ where $C = \{C_1, C_2, \ldots, C_m\}$ and $A \notin C$, *i.e.*, $b(g_i)$ represents the distance of gene g_i to its closest cluster. Now compute the silhouette width of gene g_i as

$$S(g_i) = \frac{b(g_i) - a(g_i)}{max\{a(g_i), b(g_i)\}}. \tag{4.8}$$

d) Compute silhouette index by finding the average of $S(i)$ over $i = 1, 2, \ldots, G$, where G is the total number of genes:

$$S = average\{(S(g_i)\}. \tag{4.9}$$

The value of silhouette index varies from -1 to 1 with higher values indicating better clustering. We observe from Table 4.3 that the silhouette index for clusters produced by GCA is superior then the values for clusters produced by other algorithms.

iii. Z-score

For evaluating the quality of clusters produced by different algorithms, we need an objective external criterion. We obtain a statistical rating of the relative

Table 4.2: Homogeneity values for GCA and other comparable algorithms

Datasets	Method Applied	No. of Clusters	Threshold value	Homogeneity
Dataset 2	k-means	16	NA	0.671
	SOM	16	4 × 4 grid	0.710
	CLICK	3	Default value	0.549
	DCCA	15	NA	0.818
	GCA	**16**	NA	**0.778**
Dataset 3	k-means	119	NA	0.553
	SOM	47	10 × 12 grid	0.865
	DCCA	4	NA	0.818
	GCA	**119**	NA	**0.844**
Dataset 4	k-means	8	NA	0.781
	SOM	8	2 × 4 grid	0.772
	CLICK	3	Default value	0.676
	DCCA	10	NA	0.834
	GCA	**8**	NA	**0.878**
Dataset 5	k-means	4	NA	0.512
	SOM	4	2 × 2 grid	0.562
	CLICK	6	Default value	0.729
	DCCA	10	NA	0.812
	GCA	**10**	NA	**0.901**
Dataset 6	k-means	4	NA	0.551
	SOM	4	2 × 2 grid	0.553
	CLICK	5	Default value	0.483
	DCCA	10	NA	0.813
	GCA	**4**	NA	**0.898**
Dataset 7	k-means	5	NA	0.577
	SOM	6	2 × 3 grid	0.514
	CLICK	5	Default value	0.501
	DCCA	43	NA	0.699
	GCA	**5**	NA	**0.815**

Table 4.3: Silhouette Index for GCA and other comparable algorithms

Datasets	Method Applied	No. of Clusters	Silhouette Index
Dataset 2	MOGA-SVM (RBF)	5	0.443
	MOGA (without SVM)	5	0.439
	FCM	6	0.387
	Average linkage	4	0.439
	SOM	6	0.368
	DCCA	15	0.838
	GCA	**17**	**0.87**
Dataset 4	MOGA-SVM (RBF)	6	0.451
	MOGA (without SVM)	6	0.487
	FCM	5	0.405
	Average linkage	6	0.412
	SOM	7	0.482
	CLICK	3	0.179
	DCCA	10	0.910
	GCA	**8**	**0.933**
Dataset 5	MOGA-SVM (RBF)	4	0.431
	MOGA (without SVM)	4	0.401
	FCM	4	0.364
	Average linkage	5	0.315
	k-means	10	0.652
	SOM	9	0.536
	CLICK	6	0.449
	DCCA	10	0.609
	GCA	**10**	**0.745**
Dataset 6	MOGA-SVM (RBF)	6	0.415
	MOGA (without SVM)	6	0.395
	FCM	8	0.299
	Average linkage	4	0.356
	SOM	6	0.324
	GCA	**14**	**0.631**

gene-expression activity shown by the genes associated in each cluster and the GO terms. In order to validate our clustering result, we employ z-score [GR02] as the measure of agreement. Z-score [GR02] is calculated by investigating the relation (mutual information) between a clustering obtained by an algorithm and the functional annotation of the genes in the cluster. To compute this, we use the Saccharomyces Genome Database (SGD) annotation of yeast genes, along with the gene ontology (GO) developed by the Gene Ontology Consortium [ABB+00]. A higher value of z-score indicates that genes are better clustered by function, indicating a more biologically relevant clustering result. We use the Gibbons ClusterJudge [GR02] tool to calculate z-scores. The concept of z-score computation is as follows:

1. First, parse annotation from SGD of *S. cerevisiae* genes with GO attributes in such a way that a gene-attribute table is produced in which a '1' in the position (i, j) indicates that the gene i is known to possess attribute j, and a '0' indicates lack of knowledge about whether gene i possesses attribute j or not.

2. From the gene-attribute table, construct a contingency table for each cluster-attribute pair.

3. Compute total mutual information between the cluster result C and all the attributes A_is as:

$$MI(C, A_1, A_2, \cdots A_{N_A}) = \sum_i MI(C, A_i) = N_A H_C + \sum_i H_{A_i} - \sum_i H_{A_i,C}$$

where $H_{A_i,C}$ is the entropy for each cluster-attribute pair, H_C is the entropy for the clustering result independent of attributes, and H_{A_i} is the entropy for each of the N_A attributes in the contingency table independent of clusters.

Z-score [GR02] of a clustering is computed as follows:

1. Compute Mutual Information (MI) for the clustered data (MI_{real}) by using the attribute database derived from GO/SGD;

2. Obtain a clustering by randomly assigning genes to clusters of uniform size. Compute mutual information (MI_{random}), repeating until a distribution of values is obtained;

Table 4.4: z-scores for GCA, SOM and k-means for Dataset 1

Method Applied	No. of Clusters	z-score
k-means	30	12.56
SOM	30	14.44
GCA	30	16.81

Table 4.5: z-scores for GCA, SOM, DCCA, k-means and UPGMA for reduced set of Dataset 7. DCCA is a divisive partitional algorithm reported in [BD08]

Method Applied	No. of Clusters	z-score
UPGMA	8	6.67
k-means	5	8.14
DCCA	8	9.41
SOM	8	8.16
GCA	5	9.62

3. Compute z-score as $z = (MI_{real} - MI_{random})/s_{random}$ where, mean of the MI-values computed for randomly obtained cluster is MI_{random} and standard deviation of these MI-values is s_{random}).

The z-score represents a standardized distance between the MI value obtained by clustering and those MI values obtained by random assignment of genes to clusters. The larger the z-score, the greater the distance. Higher z-scores indicate that the clustering results are more significantly related to gene function.

Also, we see in Table 4.4 that GCA performs better than other algorithms in terms of z-score measure of cluster validity. A higher z-score value indicates more biologically relevant clusters. The z-score values of clusters produced by GCA along with those produced by other algorithms for Dataset 1 and Dataset 7 are given in Table 4.4 and Table 4.5, respectively. We make the observation from the tables that GCA can cluster better than the other algorithms in terms of z-score and can hence give more biologically significant clusters.

4.5.3 Biological Significance

The biological relevance of a cluster can be verified based on the gene ontology (GO) annotation database located at *http://db.yeastgenome.org/cgi-bin/GO/goTermFinder*. It is used to test the functional enrichment of a group of genes in terms of three structured controlled ontologies, *viz.*, associated biological processes, molecular functions and biological components. The functional enrichment of each GO category in each of the clusters obtained is calculated by its *p-value*. The *p-value* is computed using a cumulative hypergeometric distribution. It measures the probability of finding the number of genes involved in a given GO term (i.e., function, process, component) within a cluster. From a given GO category, the probability p of getting k or more genes within a cluster of size n, is defined as [THC+99]:

$$p = 1 - \sum_{i=0}^{k-1} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}}$$

where f and g denote the total number of genes within a category and within the genome respectively. The genes in a cluster are evaluated for the statistical significance by computing the p-value for each GO category. This signifies how well the genes in the cluster match with the different GO categories. p-value represents the probability of observing the number of genes from a specific GO functional category within each cluster. A low p-value indicates the genes belonging to the enriched functional categories are biologically significant in the corresponding clusters.

To compute the p-value, we used the software FuncAssociate [B+03]. FuncAssociate [B+03] computes the hypergeometric functional enrichment score based on Molecular Function and Biological Process annotations. The resulting scores are adjusted for multiple hypothesis testing using Monte Carlo simulations. FuncAssociate is a Web-based tool that accepts as input a list of genes and returns a list of GO attributes that are over-represented (or under-represented) among the genes in the input list.

To test the biological significance of the clusters obtained by GCA, we use a reduced form of Dataset 3 The dataset is reduced by filtering out low variance and low entropy genes from the data The enriched functional categories for each cluster obtained by the GCA method on the reduced form of Dataset 3 are listed in Table 4 6 The functional enrichment of each GO category in each of the clusters is calculated by its p-value Of the 16 clusters obtained from the dataset, the cluster C6 contains several enriched categories on 'ribosome' The highly enriched category in C6 is the 'ribosome' with a p-value of 3 6e-13 The GO category 'ribonucleoprotein complex' is also highly enriched in this cluster with p-value of $1\ 1 \times 10^{-12}$ Cluster C1 contains genes involved in different biological processes Cluster C2 contains genes involved in different ribosomal functions C2 contains several enriched categories on 'biogenesis' The highly enriched categories in C2 are the 'ribosome biogenesis and assembly' with p-value of $1\ 5 \times 10^{-11}$, 'ribonucleoprotein complex biogenesis and assembly' with p-value of $2\ 8 \times 10^{-10}$ and 'ribosomal large subunit biogenesis and assembly' with p-value of $4\ 9 \times 10^{-07}$ Cluster C5 contains genes involved in energy synthesis The highest enriched category in C5 is 'oxidative phosphorylation' with p-value of $1\ 4 \times 10^{-14}$ C5 also contains several enriched categories on 'mitochondria' Cluster C6 contains the highly enriched cellular components of 'non-membrane-bounded organelle' and 'intracellular non-membrane-bounded organelle' with a p-value of $2\ 7 \times 10^{-14}$ each In the cluster C7 all the functionally enriched categories are from Biological Process annotation with 'trehalose metabolic process' with a p-value of $2\ 5 \times 10^{-09}$ being the highly enriched one C7 contains genes involved in the functions of metabolism Cluster C8 contains several enriched categories on 'catabolic process' with 'cellular catabolic process' having a p-value of 1×10^{-05} being the highly enriched category C8 contains functional categories on energy synthesis and metabolic pathways Cluster C9 contains genes involved in metabolic pathways with 'mitochondrial respiratory chain' having the highest p-value of 4×10^{-07} From the Table 4 6, we can conclude that GCA shows a good enrichment of functional categories and therefore project a good biological significance

Table 4.6: P-value of Dataset 3

Cluster	P-value	GO number	GO category
C1	1e-10	GO:0006119	oxidative phosphorylation
	5.4e-10	GO:0006091	generation of precursor metabolites and energy
	2.8e-08	GO:0022900	electron transport chain
	2.8e-08	GO:0022904	respiratory electron transport chain
	2.8e-08	GO:0042773	ATP synthesis coupled electron transport
	2.8e-08	GO:0042775	organelle ATP synthesis coupled electron transport
	2.8e-08	GO:0055114	oxidation reduction
	7.2e-08	GO:0005739	mitochondrion
	7.9e-08	GO:0044455	mitochondrial membrane part
	1.5e-07	GO:0015078	hydrogen ion transmembrane transporter activity
	1.9e-07	GO:0005743	mitochondrial inner membrane
	2.8e-07	GO:0015077	monovalent inorganic cation transmembrane transporter activity
	3.3e-07	GO:0019866	organelle inner membrane
	3.5e-07	GO:0031966	mitochondrial membrane
	7.6e-07	GO:0005740	mitochondrial envelope
	1.7e-06	GO:0016310	phosphorylation
	2e-06	GO:0005746	mitochondrial respiratory chain
	2.5e-06	GO:0006793	phosphorous metabolic process
	2.5e-06	GO:0006796	phosphate metabolic process
	1.2e-05	GO:0022890	inorganic cation transmembrane transporter activity
	1.4e-05	GO:0044429	mitochondrial part
	2.1e-05	GO:0031967	organelle envelope
	2.2e-05	GO:0031975	envelope

Cluster	P-value	GO number	GO category
C1	5.8e-05	GO:0006122	mitochondrial electron transport, ubiquinol to cytochrome c
	7.6e-05	GO:0000276	mitochondrial proton-transporting ATP synthase complex, coupling factor F(o)
	7.6e-05	GO:0006123	mitochondrial electron transport, cytochrome c to oxygen
	7.6e-05	GO:0045263	proton-transporting ATP synthase complex, coupling factor F(o)
	8.5e-05	GO:0016491	oxidoreductase activity
	9.5e-05	GO:0009060	aerobic respiration
C2	1.5e-11	GO:0042254	ribosome biogenesis and assembly
	4e-11	GO:0005730	nucleolus
	2.8e-10	GO:0022613	ribonucleoprotein complex biogenesis and assembly
	3e-10	GO:0043228	non-membrane-bounded organelle
	3e-10	GO:0043232	intracellular non-membrane-bounded organelle
	4.9e-07	GO:0042273	ribosomal large subunit biogenesis and assembly
	9.4e-07	GO:0006364	rRNA processing
	1.1e-06	GO:0016072	rRNA metabolic process
	1.8e-06	GO:0031981	nuclear lumen
	1.7e-05	GO:0030529	ribonucleoprotein complex
C3	7.7e-10	GO:0042254	ribosome biogenesis and assembly
	8.6e-10	GO:0022613	ribonucleoprotein complex biogenesis and assembly
	3.4e-05	GO:0042273	ribosomal large subunit biogenesis and assembly
	3.5e-05	GO:0004410	homocitrate synthase activity

Cluster	P-value	GO number	GO category
C5	1.4e-14	GO:0006119	oxidative phosphorylation
	8.5e-14	GO:0044455	mitochondrial membrane part
	6.3e-11	GO:0015078	hydrogen ion transmembrane transporter activity
	9.8e-11	GO:0006091	generation of precursor metabolites and energy
	1.4e-10	GO:0015077	monovalent inorganic cation transmembrane transporter activity
	1.8e-09	GO:0005753	mitochondrial proton-transporting ATP synthase complex
	1.8e-09	GO:0045259	proton-transporting ATP synthase complex
	3.8e-09	GO:0005743	mitochondrial inner membrane
	7.2e-09	GO:0019866	organelle inner membrane
	1e-08	GO:0015985	energy coupled proton transport, down electrochemical gradient
	1e-08	GO:0015986	ATP synthesis coupled proton transport
	1.3e-08	GO:0006754	ATP biosynthetic process
	1.3e-08	GO:0046034	ATP metabolic process
	2e-08	GO:0022890	inorganic cation transmembrane transporter activity
	2.6e-08	GO:0005746	mitochondrial respiratory chain
	2.6e-08	GO:0009144	purine nucleoside triphosphate metabolic process
	2.6e-08	GO:0009145	purine nucleoside triphosphate biosynthetic process
	2.6e-08	GO:0009205	purine ribonucleoside triphosphate metabolic process

Cluster	P-value	GO number	GO category
C5	2.6e-08	GO:0009206	purine ribonucleoside triphosphate biosynthetic process
	4e-08	GO:0009199	ribonucleoside triphosphate metabolic process
	4e-08	GO:0009201	ribonucleoside triphosphate biosynthetsic process
	4.4e-08	GO:0016310	phosphorylation
	5.8e-08	GO:0009142	nucleoside triphosphate biosynthetic process
	6.2e-08	GO:0008324	cation transmembrane transporter activity
	7e-08	GO:0016469	proton-transporting two-sector ATPase complex
	7.1e-08	GO:0031966	mitochondrial membrane
	8.3e-08	GO:0009141	nucleoside triphosphate metabolic process
	1.6e-07	GO:0005740	mitochondrial envelope
	1.6e-07	GO:0006818	hydrogen transport
	1.6e-07	GO:0015992	proton transport
	2.7e-07	GO:0015075	ion transmembrane transporter activity
	6.4e-07	GO:0022900	electron transport chain
	6.4e-07	GO:0022904	respiratory electron transport chain
	6.4e-07	GO:0042773	ATP synthesis coupled electron transport
	6.4e-07	GO:0042775	organelle ATP synthesis coupled electron transport
	6.4e-07	GO:0055114	oxidation reduction
	6.6e-07	GO:0006793	phosphorous metabolic process
	6.6e-07	GO:0006796	phosphate metabolic process

Cluster	P-value	GO number	GO category
C5	8.1e-07	GO:0009152	purine ribonucleotide biosynthetic process
	9.1e-07	GO:0009150	purine ribonucleotide metabolic process
	1e-06	GO:0015672	monovalent inorganic cation transport
	1.1e-06	GO:0009260	ribonucleotide biosynthetic process
	1.3e-06	GO:0009259	ribonucleotide metabolic process
	1.6e-06	GO:0006164	purine nucleotide biosynthetic process
	1.9e-06	GO:0006163	purine nucleotide metabolic process
	2.5e-05	GO:0000275	mitochondrial proton-transporting ATP synthase complex, catalytic core F(1)
	2.5e-05	GO:0045261	proton-transporting ATP synthase complex, catalytic core F(1)
	2.8e-05	GO:0044429	mitochondrial part
	4.3e-06	GO:0046933	hydrogen ion transporting ATP synthase activity, rotational mechanism
	4.7e-06	GO:0031967	organelle envelope
	4.9e-06	GO:0031975	envelope
	8.2e-06	GO:0022891	substrate-specific transmembrane transporter activity
	1.5e-05	GO:0005739	mitochondrion
	2e-05	GO:0046961	hydrogen ion transporting ATPase activity, rotational mechanism

Cluster	P-value	GO number	GO category
C5	2.5e-05	GO:0009165	nucleotide biosynthetic process
	2.7e-05	GO:0022857	transmembrane transporter activity
	2.8e-05	GO:0016491	oxidoreductase activity
	3.6e-05	GO:0019829	cation-transporting ATPase activity
	4.1e-05	GO:0005754	mitochondrial proton-transporting ATP synthase, catalytic core
	4.1e-05	GO:00452671	proton-transporting ATP synthase, catalytic core
	4.3e-05	GO:0022892	substrate-specific transporter activity
	5.3e-05	GO:0005751	mitochondrial respiratory chain complex IV
	5.3e-05	GO:0006123	mitochondrial electron transport, cytochrome c to oxygen
	5.3e-05	GO:0045277	respiratory chain complex IV
	6.9e-05	GO:0033178	proton-transporting two-sector ATPase complex, catalytic domain
C6	2.7e-14	GO:0043228	non-membrane-bounded organelle
	2.7e-14	GO:0043232	intracellular non-membrane-bounded organelle
	3.6e-13	GO:0005840	ribosome
	1.1e-12	GO:0030529	ribonucleoprotein complex
	1.5e-12	GO:0042254	ribosome biogenesis and assembly

Cluster	P-value	GO number	GO category
C6	1.7e-12	GO:0022613	ribonucleoprotein complex biogenesis and assembly
	2.2e-12	GO:0022626	cytosolic ribosome
	9.5e-12	GO:0003735	structural constituent of ribosome
	3e-11	GO:0044445	cytosolic part
	1.7e-10	GO:0033279	ribosomal subunit
	3.7e-09	GO:0005198	structural molecule activity
	2.6e-08	GO:0044249	cellular biosynthetic process
	3.1e-08	GO:0022625	cytosolic large ribosomal subunit
	4.4e-08	GO:0005730	nucleolus
	3.4e-07	GO:0006412	translation
	6.8e-07	GO:0015934	large ribosomal subunit
	1.1e-06	GO:0009058	biosynthetic process
	1.8e-06	GO:0010467	gene expression
	4.1e-06	GO:0006996	organelle organization and biogenesis
	6.5e-06	GO:0009059	macromolecule biosynthetic process
	1.6e-05	GO:0032991	macromolecular complex
	1.6e-05	GO:0006364	rRNA processing
	1.9e-05	GO:0016072	rRNA metabolic process
	2.5e-05	GO:0022627	cytosolic small ribosomal subunit
	2.6e-05	GO:0019843	rRNA binding
	3e-05	GO:0005829	cytosol
	5.5e-05	GO:0044452	nucleolar part
	8e-05	GO:0031981	nuclear lumen

Cluster	P-value	GO number	GO category
C7	2.5e-09	GO:0005991	trehalose metabolic process
	1.9e-08	GO:0005975	carbohydrate metabolic process
	2.6e-08	GO:0044262	cellular carbohydrate metabolic process
	5.8e-08	GO:0005992	trehalose biosynthetic process
	5.8e-08	GO:0046351	disaccharide biosynthetic process
	1.9e-07	GO:0046164	alcohol catabolic process
	9.4e-07	GO:0005996	monosaccharide metabolic process
	1.1e-06	GO:0009946	alpha, alpha-trehalose-phosphate synthase complex (UDP-forming)
	2.5e-06	GO:0046365	monosaccharide catabolic process
	2.7e-06	GO:0016052	carbohydrate catabolic process
	2.7e-06	GO:0044275	cellular carbohydrate catabolic process
	4.3e-06	GO:0006096	glycolysis
	6e-06	GO:0019200	carbohydrate kinase activity
	6.3e-06	GO:0005984	disaccharide metabolic process
	7.1e-06	GO:0006066	alcohol metabolic process
	8e-06	GO:0019318	hexose metabolic process
	1.9e-05	GO:0006007	glucose catabolic process
	2.1e-05	GO:0006006	glucose metabolic process
	3e-05	GO:0019320	hexose catabolic process
	4.3e-05	GO:0004186	carboxypeptidase C activity
	4.3e-05	GO:0043043	peptide biosynthetic process
	4.3e-05	GO:0046937	phytochelatin metabolic process
	4.3e-05	GO:0046938	phytochelatin biosynthetic process
	7.2e-05	GO:0044265	cellular macromolecule catabolic process

Cluster	P-value	GO number	GO category
C8	1e-05	GO:0044248	cellular catabolic process
	1.8e-05	GO:0006099	tricarboxylic acid cycle
	1.8e-05	GO:0046356	acetyl-CoA catabolic process
	1.9e-05	GO:0009056	catabolic process
	2e-05	GO:0015980	energy derivation by oxidation of organic compounds
	3.1e-05	GO:0006084	acetyl-CoA metabolic process
	3.1e-05	GO:0009109	coenzyme catabolic process
	3.1e-05	GO:0004867	serine-type endopeptidase inhibitor activity
	3.5e-05	GO:0051187	cofactor catabolic process
	8e-05	GO:0006091	generation of precursor metabolites and energy
C9	4e-07	GO:0005746	mitochondrial respiratory chain
	1.9e-06	GO:0004061	peroxidase activity
	1.9e-06	GO:0016684	oxidoreductase activity, acting on peroxide as acceptor
	5.2e-06	GO:0006091	generation of precursor metabolites and energy
	5.4e-06	GO:0016209	antioxidant activity
	9.4e-06	GO:0005743	mitochondrial inner membrane
	1.4e-05	GO:0006099	tricarboxylic acid cycle
	1.4e-05	GO:0046356	acetyl-CoA catabolic process
	1.4e-05	GO:0019866	organelle inner membrane
	2.4e-05	GO:0006084	acetyl-CoA catabolic process
	2.4e-05	GO:0009109	coenzyme catabolic process
	2.7e-05	GO:0051187	cofactor catabolic process
	3.1e-05	GO:0016491	oxidoreductase activity
	5.3e-05	GO:0044429	mitochondrial part
	6.2e-05	GO:0005740	mitochondrial envelope

4.6 Discussion

This chapter presents a parameter-less clustering technique that uses a dynamically calculated threshold to assign cluster membership. Our experimental results show that the clusters obtained are similar to those obtained by [TSM+99]. The GCA method also obtains better Rand index, homogeneity, silhouette and z-score values than several competitors, showing that GCA can cluster gene expression data effectively. Unlike hierarchical algorithms, GCA does not build a tree of clusters but a set of disjoint clusters. In contrast with SOM [Koh95], it does not assume the number of clusters and spatial structure, but determines the cluster number and structure based on the dataset itself. Also, the clusters obtained by GCA is found to be of high biological significance.

Chapter 5

Coherent Pattern Extraction using Maximal Frequent Patterns

This chapter presents a frequent itemset nearest neighbor based technique for clustering gene expression data. It attempts to find finer clusters over the gene expression data by integrating the nearest neighbor clustering technique with frequent itemset discovery. The advantage of using frequent itemset discovery is that it can capture relations among more than two genes while normal similarity measures can calculate the proximity between only two genes at a time. We experimented with FINN using real-life datasets and we observe that it can find the finer clustering of the dataset.

5.1 Introduction

Association rule learning is a popular and well researched method for discovering interesting associations and/or correlation relationships among large set of data items Association rules show attribute value conditions that occur frequently together in a given dataset Association rule mining has received considerable attention since its introduction in [ATS93] A typical and widely-used example of association rule mining is Market Basket Analysis The market-basket problem assumes we have some large number of items, e g , "bread", " milk", "butter" Customers fill their market baskets with some subset of the items, and we get to know what items people buy together An example rule for the supermarket could be $\{milk, bread\} \rightarrow \{butter\}$ meaning that if milk and bread is bought, customers also buy butter Marketers use this information to position items, and control the way a typical customer traverses the store

Association rules provide information of this type in the form of "if-then" statements These rules are computed from the data and, unlike the if-then rules of logic, association rules are probabilistic in nature Association mining analysis is a two part process First, is the identication of sets of items or itemsets within the dataset Second, the subsequent derivation of inferences or rules from these itemsets

Association rules follow the form $X \rightarrow Y$ where X and Y are disjoint sets of items (or itemsets) i e , X and Y are subsets of the set of items A in the transaction database T X is called the antecedent and Y the consequent of the rule The intended meaning of such a rule is that data instances that contain X are likely to contain Y as well The extent to which the rule applies to a given dataset can be measured using various metrics including support and confidence The support of a rule is the probability of X and Y occurring togethei in an instance, *P(X and Y)* The confidence of a rule is the conditional probability of Y given X, $P(Y \mid X)$ Here, probability is taken to be the observed frequency in the underlying dataset An itemset $X \subset A$ is said to *frequent* in T w r t support s, if $support(X) \geq s$ A frequent set is a *maximal frequent set* if it is a frequent set and no superset of this is a frequent set Association rules are required to satisfy a user-specified minimum support and a user-specified minimum confidence at

the same time. To achieve this, association rule generation is a two-step process. First, minimum support is applied to find all frequent itemsets in a database. In a second step, these frequent itemsets and the minimum confidence constraint are used to form rules. While the second step is straight forward, the first step needs more attention. We next present a review of some selected association mining techniques.

5.2 Related Work

A review of frequent pattern mining strategies is given in [ADRB+09]. This section discusses various methods for gene association analysis in DNA microarray gene expression data.

5.2.1 Apriori Algorithm

The Apriori algorithm [ATS93] is a pioneering algorithm for association rule mining; it finds all frequent itemsets whose supports are above a threshold. It is based on the fact that all subsets of a frequent itemset are also frequent. The algorithm first makes one pass over the dataset and finds the large items. Then the algorithm makes many passes over the data. Each pass starts with the seed set of large itemsets which are used to generate new potentially large itemsets called candidate itemsets. Then, support for each candidate itemset is found during the pass over the data and actual large itemsets are determined. These large itemsets become the seed for the next pass. This process continues till no more large itemsets can be found. The algorithm is very easy to implement and finds all possible frequent itemsets. However, it is expensive from the view point of storage as well as execution time.

5.2.2 AprioriTid Algorithm

AprioriTid algorithm [JPZ03b] is a modification of the Apriori algorithm. It generates the candidates using the same candidate generating function as Apriori. The main feature of the algorithm is that the original database is not used after

the first pass Instead a data structure C_k' is used Each member of the set C_k' is of the form $< TID, \{X_k\} >$, where X_k is a potentially large k-itemset present in the transaction with the identifier TID For $k = 1$, C_k' is the database itself with each item i replaced by itemset $\{i\}$. For $k > 1$, the member of C_k' corresponding to a transaction t is $< t\,TID, \{c \in C_k \mid c$ contained in $t\} >$ If a transaction does not contain any candidate set, C_k' does not have any entry for that transaction So the number of entries in C_k' gets reduced in successive passes resulting in fewer transactions to be scanned in each subsequent pass One shortcoming of the algorithm is the creation and updation of C_k', which takes considerable amount of execution time It differs from Apriori in that it scans the database once and uses a better data structure for the rest of the iterations It suffers from the similar disadvantages as Apriori and in addition requires extra memory and extra disk space for the data structure Moreover, to maintain the data structure extra time is required

5.2.3 AprioriHybrid Algorithm

AprioriHybrid algorithm [JPZ03b] is basically a fusion of Apriori and AprioriTid It uses Apriori for the first few passes and AprioriTid for the remaining passes based on some threshold value, *i e* , when it finds that candidates can be stored in memory, it uses AprioriTid It has the advantages of both the algorithms and is superior to both However, it also suffers from the disadvantages of both the algorithms

5.2.4 FP-Tree Growth Algorithm

The FP-growth algorithm [HPY00] finds frequent itemsets without candidate generation The algorithm is based on a data structure called FP(Frequent Pattern)-tree, which is a prefix tree of the transactions of the database such that each path represents a set of transactions that share the same prefix The algorithm first scans the database once to find frequent items in the database Infrequent items are removed from the database and items in the transactions are rearranged in the descending order of frequency Then, the least frequent items are removed

from the transactions, resulting in a reduced (projected) database This projected database is processed to find frequent itemsets The process is repeated with the next least frequent item The FP-tree contains all necessary information about the transactions and the frequent itemsets So to find any information about the transactions and the frequent itemsets, just the tree needs to be searched The FP-growth algorithm is one of the most efficient algorithms for finding frequent itemsets from large databases The FP-tree algorithm [HPY00] does not rely on a candidate generation step and is therefore faster than the Apriori algorithm However, the algorithm takes much time to construct the FP-tree, especially for higher dimensions

Association rules, used widely in the area of market basket analysis, can be applied to the analysis of expression data as well Association rules can reveal biologically relevant associations between different genes or between environmental effects and gene expression In the analysis of gene expression data, the items in an association rule can represent genes that are strongly expressed or repressed, as well as relevant facts describing the cellular environment of the genes (e g a diagnosis for a tumor sample that was profiled, or a drug treatment given to cells in the sample before profiling) An example of an association rule mined from expression data might be $\{cancer\} \rightarrow \{gene\ A \uparrow,\ gene\ B \downarrow,\ gene\ C \uparrow\}$, meaning that, for the data set that was mined, in most profile experiments where the cells used were cancerous, gene A was measured as being up (i e highly expressed), gene B was down (i e highly repressed), and gene C was up, altogether

Recently, several authors have proposed the use of association rules for the analysis of gene expression data [CH03, TA02, CSCR+06] in order to extract associations and relationships among subsets of genes This approach avoids some of the drawbacks of standard clustering algorithms and has been successful in extracting new and informative gene relationships A major disadvantage of the association rules discovery method is the large number of rules that are generated This becomes a major problem in many applications In several studies, post-processing pruning methods have been proposed to reduce the number of generated rules For example, in the context of gene expression, Creighton and Hanash [CH03] impose constraints on the size of the rules, extracting only those formed by seven or more genes while Tuzhilin and Adomavicius [TA02] propose

several post-processing operators to select and explore interesting rules from the whole set. In [CSCR+06], another method for the integrative analysis of microarray data based on the association rules discovery technique is presented. The approach integrates gene annotations and expression data to discover intrinsic associations among both data sources based on co-occurrence patterns. Filter options have been used to eliminate irrelevant and redundant associations. This option drastically reduces the number of associations to be examined. In [GWBO+07], the authors propose a new similarity measure that can be applied together with hierarchical clustering leading to grouped similar patterns. The mining part first constructs a compact data structure called Gene Profile tree (or GP-tree), from which the frequent co-regulated gene profiles are extracted.

5.3 Motivation

In this chapter, we present a finer clustering method that integrates a traditional clustering technique with frequent itemset discovery. The gene expression dataset is encoded in binary with respect to correlated genes. Frequent itemset mining is then run on this data to discover the maximal frequent set(s). This maximal frequent set gives the core genes in a cluster. Cluster expansion proceeds with this set of core genes using a shared neighbor approach. Previous authors have applied frequent itemset mining to gene expression data. However, to the best of our knowledge, an approach similar to the one reported in this chapter has not yet been explored in the domain of gene expression datasets. The advantage of our method is that it produces finer clustering of the dataset. Also, it avoids redundant checking and guarantees to form clusters. The advantage of using frequent itemset discovery is that it can capture relations among more than two genes while normal similarity measures can calculate the proximity between only two genes at a time.

5.4 Frequent Itemset Mining and Nearest Neighbor Clustering (FINN)

The FINN algorithm exploits frequent itemsets and uses a nearest neighbor approach for clustering gene sets. Most work related to the application of association rule mining on gene expression profiles relies on discretization of the data before applying any data mining technique. Although discretization may imply loss of information, it also alleviates noise [CH03], [CSCR+06]. The FINN algorithm works in three phases. In the first phase, the gene expression data D_G is transformed into a 0-1 transaction matrix. The second phase finds maximal frequent itemsets using a frequent itemset mining algorithm such as Apriori or FP-tree Growth algorithm. The third phase is dedicated to the task of clustering using a shared nearest neighbor based approach.

5.4.1 Phase I: Transformation from Gene Expression Matrix to Transaction Matrix

The gene expression dataset D_G is a $G \times T$ matrix of expression values where G is the number of rows (genes) and T is the number of columns (time points) as shown in Equation 5.1. Using our dissimilarity measure between the genes, we build a $G \times G$ dissimilarity matrix for the whole dataset. We introduce some definitions as we proceed with the description of our method.

Definition 5.1. Nearest Neighbor of a gene
A gene g_i is the nearest neighbor of a gene g_j if $DBK(g_i, g_j) \leq \theta_1$, where θ_1 is the nearest neighbor threshold and DBK is our dissimilarity measure discussed in Chapter 3.

From the nearest neighbor lists, we build the $G \times G$ gene-gene transaction matrix, T_G, of zeroes and ones (Equation 5.2). For each gene g_i, a T-pattern of 0's and 1's is obtained with 1 if a gene g_j is neighbor of g_i and 0 otherwise as given in (Equation 5.3).

$$D_G = \begin{bmatrix} a_{11}\ a_{12} \cdots\cdots a_{1T} \\ a_{21}\ a_{22} \cdots\cdots a_{2T} \\ \vdots \\ a_{G1}\ a_{G2} \cdots\cdots a_{GT} \end{bmatrix} \tag{5.1}$$

$$T_G = \begin{bmatrix} t_{11}\ t_{12} \cdots\cdots t_{1G} \\ t_{21}\ t_{22} \cdots\cdots t_{2G} \\ \vdots \\ t_{G1}\ t_{G2} \cdots\cdots t_{GG} \end{bmatrix} \tag{5.2}$$

$$T_G = t_{ij} = \begin{cases} 1\ if\ DBK(g_i, g_j) \leq \theta_1, where\ i = 1, 2, \cdots G; \\ \qquad\qquad\qquad\qquad\qquad\qquad j = 1, 2, \cdots G\ and\ i \neq j. \\ 0\ otherwise \end{cases} \tag{5.3}$$

Pruning

Those transactions are pruned to satisfy the following conditions.

i. In the transaction matrix, the value of t_{ij}, where $i = j$ is set to zero since a gene does not contribute to frequent itemset generation with itself.

ii. In this transaction matrix, if for a particular row i the value of t_{ij} across all j conditions are zero and the same applies for column j and all i rows, that i^{th} row and j^{th} column both are discarded.

These two steps reduce the size of the transaction matrix considerably.

Phase II now uses T_G to calculate the frequent itemset using FP-tree Growth algorithm.

5.4.2 Phase II: Maximal Frequent Itemset Generation

In this phase, we use the FP-tree Growth algorithm to generate maximal frequent itemset(s) (MFIS) at support threshold $sup_c\%$. The gene-gene $G \times G$ transaction

matrix, T_G is fed as input along with a user defined support threshold to obtain frequent itemsets. The maximal frequent itemset obtained from this phase gives us the set of core genes. The identification of core genes is done as follows.

- If only one MFIS is obtained at $sup_c\%$ support, the genes within that set become the set of core genes for a particular cluster.
- If more than one MFIS is obtained at $sup_c\%$ support and there is a chain of genes (items) from one MFIS to the other, the genes are merged together into the set of core genes for a particular cluster.
- If more than one MFIS is obtained at $sup_c\%$ support and there is no chain of genes (items) from one MFIS to the other, each MFIS gives the set of core genes for a different cluster.

This set of core genes provides the seeds for cluster expansion, giving the core clustering of the dataset. Different clustering approaches such as hierarchical or density based clustering can be applied on these core genes to get the final cluster. The next phase gives a detailed overview of the clustering process.

The following definitions provide the foundation for the clustering process.

Definition 5.2. Density of a gene
The density of a gene g_i is the number of nearest neighbors of that gene in the gene-gene transaction matrix, T_G.

$$Density(g_i) = \sum_{j=1}^{G} t_{ij}, \text{ where } t_{ij} = 1 \tag{5.4}$$

Definition 5.3. Core genes
A set of core genes C_{r_i} that gives a cluster C_i is defined by an $MFIS_i$ i.e., a maximal frequent itemset generated by the FP-tree Growth algorithm [HPY00]. Assume $MFIS\_set$ is the set of k maximal frequent itemsets generated by FP-Tree growth algorithm. Then, the set of core genes, C_{r_i} may be obtained as follows:

Case 1. If k = 1, $C_{r_i} = MFIS_i$, where $i = 1$ and $MFIS\_set = MFIS_i$.

Case 2. If $k > 1$ and $MFIS_i \bigcap MFIS_j \neq \phi$, for $i \neq j$, and $i = 1, 2, \cdots, k$, $j = 1, 2, \cdots, k$. Then, $MFIS_i = MFIS_i \bigcup MFIS_j$ and $k = k - 1$.
Finally, we have,
$MFIS\_set = MFIS_1, MFIS_2, \cdots, MFIS_k$. Then, $C_{r_i} = MFIS_i$ for $i = 1, 2, \cdots, k$.

Here, each C_{r_i} will give the set of core genes of different clusters and the total number of clusters given by this $MFIS\_set$ is k.
For better understanding of the above cases, we take the help of an example as given next.

Case 1. Let $MFIS\_set = \{1, 2, 4\}$, hence $k = 1$ and $C_{r_1} = \{1, 2, 4\}$ which is the set of core genes of a particular cluster.

Case 2. Let $MFIS\_set = \{1, 2, 3, 4\}, \{6, 7\}, \{8, 9\}, \{1, 4, 10, 11\}$ where $k = 4$. Since, $MFIS_1$ and $MFIS_4$ have the genes 1 and 4 as common, therefore we take the union of these MFISs to obtain,
$MFIS\_set = \{1, 2, 3, 4, 10, 11\}, \{6, 7\}, \{8, 9\}$, now $k = 3$.
Therefore, $C_{r_1} = \{1, 2, 3, 4, 10, 11\}$, $C_{r_2} = \{6, 7\}$, and $C_{r_3} = \{8, 9\}$.
Here, C_{r_1} consists of the core genes of a cluster. Similarly, C_{r_2} and C_{r_3} are core genes of two different clusters. Thus, we obtain three different clusters for the given example $MFIS\_set$.

Definition 5.4. Shared Neighbors
Let C_{r_i} be the set of core genes and $C_{r_i} = \{g_1, \cdots, g_x\}$. A gene g_q is said to be the shared neighbor of each of the core genes in C_{r_i}, if it satisfies the following condition:

$$sn(C_{r_i}, g_q) = \begin{array}{l} DBK(g_1, g_q) \leq \beta \ \wedge DBK(g_2, g_q) \\ \leq \beta \wedge \cdots \wedge DBK(g_x, g_q) \leq \beta \end{array} \tag{5.5}$$

where β is the shared neighbor threshold.

Definition 5.5. Cluster
A cluster, C, can be defined as the set of core genes along with their shared neighbors.

Definition 5.6. Noise genes
A gene g_q is said to be a noise gene, if it has no nearest neighbor gene g_m, where $g_m \in G$.

The following lemmas provide the foundation of FINN.

Lemma 5.1. A gene belonging to an MFIS has nearest neighbors.

Proof. A gene g_j can be a member of an $MFIS_i$ iff g_j is frequent over T_G at $s\%$ support. Therefore, g_j has nearest neighbors to it and hence the proof. □

Lemma 5.2. Seeds selected for cluster expansion cannot be noise.

Proof. Let g_{ij} be the j^{th} gene in C_{r_i} and g_{ij} is a seed, i.e., $g_{ij} \in C_{r_i}$, where $C_{r_i} = MFIS_i$. Then g_{ij} has nearest neighbors according to Lemma 5.1. Again, according to Definition 5.6, a gene with nearest neighbors cannot be a noise gene and hence the proof. □

5.4.3 Phase III: Clustering

We use a shared neighbor approach to expand the cluster from the core clustering to obtain the final clusters. The clustering procedure is initiated from the set of core genes, C_{r_i}, $(i = 1, \cdots, k)$, identified in Phase II. First, these genes are labeled. The set of core genes are classified as follows.

If $C_{r_i} = \{MFIS_i\}$ and $MFIS_i = \{g_1, g_2, \cdots, g_x\}$,

Label $\{g_1, g_2, \cdots, g_x\}$ with the same cluster_id.

For a labeled C_{r_i} of cardinality x, an arbitrary unclassified gene g_q is a shared neighbor if g_q is a shared neighbor of each of the genes of that C_{r_i} w.r.t. β. A major advantage of FINN is that it eliminates exhaustive neighbor search over T_G. If g_q has dissimilarities less than a given *shared neighbor threshold* (β) with each of the core genes of C_{r_i}, g_q is labeled with the same cluster_id as that of the core genes of that C_{r_i} and grouped into the same cluster. This process of cluster expansion is iterated until there are no more genes that can be merged into this

cluster. The cluster thus obtained gives a final cluster. This process repeats for all C_{r_i}, where $i = 1, \cdots, k$. Finall, we obtain k clusters.

Once cluster expansion terminates, the row and column of each classified gene in the transaction matrix T_G are discarded from further consideration. This step reduces the number of items (genes) which have to be checked for itemset generation in the next iteration. The process then restarts phase II with the new compact transaction matrix T_G.

The steps of the FINN approach are given below.

i. Calculate the $G \times G$ dissimilarity matrix using our dissimilarity measure and generate the $G \times G$ gene-gene transaction matrix.

ii. Generate maximal frequent itemsets ($MFIS\_set$) using FP-tree algorithm on T_G.

iii. Classify each of the set of core genes, C_{r_i} ($MFIS_i$) with the same cluster_id.

iv. Select an unclassified gene, g_q, and classify it with the same cluster_id as that of C_{r_i}, if g_q is a shared neighbor of each of the core genes in C_{r_i}.

v. Repeat step iv till no more genes satisfy the shared neighbor condition

vi. Discard the rows and columns of the classified genes from the gene-gene transaction matrix.

vii. Increment i and goto step iv.

viii. Repeat through step ii. till all genes in T_G are classified.

The algorithm for FINN is given in Figure 5.1. The input to the algorithm is the gene dataset, D_G; the number of genes, G; nearest neighbor threshold, θ_1, support count, sup_c; and shared neighbor threshold, β. The first line of the algorithm calls the $create\_transaction\_matrix(\theta, G)$ module to create the transaction matrix according to Figure 5.2. The module $find\_DBK(i, j)$ calculates the DBK distance between genes i and j. the $MFIS\_set$ will hold the maximal frequent itemsets generated by the FP-Tree growth algorithm. The $call\_FP\_Tree(sup\_c)$ module of Figure 5.1 calls the FP Tree Growth algorithm of

[HPY00] to generate maximal frequent itemset(s) (MFIS(s)) at support threshold *sup_c* Each MFIS is a set of frequent genes The k in the figure holds the total number of MFIS generated by the FP Tree Growth algorithm The *check_MFIS*$(MFIS\_set, k)$ module given in Figure 5 3, gives the set of core genes for different clusters from the different maximal frequent sets stored in $MFIS\_set$ The total number of core gene sets is stored in *actual_count*, which will finally be used to generate *actual_count* number of clusters The module *get_MFIS_tokens*$(i, MFIS)$ inserts the i^{th} MFIS into $MFIS\_tokens[i]$ which is of length k The *get_individual_token*$(l, MFIS\_tokens[i])$ extracts each individual gene l from the i^{th} MFIS The *process_MFIS*( ) combines the MFIS as explained in Case 2 of Definition 5 3 to obtain the set of core genes The *get_token_length*$(MFIS\_tokens[increment])$ module gives the total number of genes of each $MFIS\_tokens[increment]$ and stores it in *token_length* The module *cluster*$(MFIS\_tokens[increment], token\_length, cl\_id, \beta)$ of Figure 5 4 generates the clusters using the shared neighbor clustering described before The results of clustering using the FINN method using our dissimilarity measure are reported in Section 5 5 1

5.5 Performance Evaluation

We implemented the FINN method is implemented in Java in Windows environment and evaluated it using the real-life datasets discussed in Chapter 3

5.5.1 Results of FINN Clustering

We exhaustively tested the FINN approach on all the datasets in Table 3 5 The similarity matrix is first computed and the transaction matrix is obtained from it When the method is executed on Dataset 1, the clusters obtained agree well with the functional classification of [CCW+98] Of the different clusters obtained from Dataset 2, two are shown in this chapter The first cluster along with its core genes of Dataset 2 is shown in Figure 5 5 and Figure 5 6 The second cluster results are shown in Figure 5 7 and Figure 5 8

When we execute the method on Dataset 4, we obtain eight clusters Some

$FINN(D_G, G, \theta_1, sup\_c, \beta)$

create_transaction_matrix(θ_1, G),

do

$MFIS\_set$ = "",

k = -1,

$actual\_count$ = 0,

$k = call\_FP\_Tree(sup\_c)$,

IF $k \geq 1$ do

$actual\_count = check\_MFIS(MFIS\_set, k)$,

$increment$ = 0,

do

$token\_length = get\_token\_length(MFIS\_tokens[increment])$,

$cluster(MFIS\_tokens[increment], token\_length, cl\_id, \beta)$,

$increment++$,

cl_id++,

WHILE $actual\_count - increment \geq 1$,

End IF

$sup\_c = sup\_c$-5,

WHILE $sup\_c \geq 10$,

Figure 5.1 Algorithm of FINN

```
create_transcation_matrix(θ1 G)
  FOR i from 0 to G do
    FOR j from 0 to G do
      dissimilarity = find_DBK(i, j),
      IF dissimilarity ≤ θ1 do
        transaction_matrix[i][j] = 1,
      ELSE
        transaction_matrix[i][j] = 0,
      End IF
    End FOR
  End FOR
```

Figure 5 2 Algorithm for computing the transaction matrix

of the clusters obtained are shown in Figure 5 10 and 5 12 and their respective core genes are shown in Figure 5 9 and Figure 5 11 From these results, we can also conclude that the core genes give the overall trend of the cluster Therefore, this approach can also be used to detect the embedded clusters in the dataset

5.5.2 Cluster Quality

In this section, we validate the results obtained by the FINN approach using Average Homogeneity [SS00], Silhouette index [Rou87] and z-score [GR02] measures of cluster validity The homogeneity and silhouette index values for FINN along with some of the other algorithms are given in Tables 5 1 and 5 5 respectively

To test the performance of the clustering algorithm, we compare the clusters identified by our method with the results from k-means, UPGMA, DCCA and SOM The result of applying the z-score measure on Dataset 2 is shown in Table 5 3 Table 5 3 clearly shows that FINN outperforms k-means, DCCA and SOM w r t the cluster quality The z-score values obtained from clustering the reduced set of Dataset 3 is given in Table 5 4 The Dataset is reduced by using the technique in [HCML05] As can be seen in the table, our method performs better than k-means, hierarchical clustering, DCCA and SOM

```
check_MFIS(MFIS_set, count1)
  MFIS_tokens[count1]
  count = count1;
  FOR i from 0 to count1-1 do
    MFIS_tokens[i] = get_MFIS_tokens(i, MFIS_set);
  End FOR
  IF count > 1 do
    FOR i from 0 to count1-1 do
      FOR j from (i + 1) to count1-1 do
        FOR l from 0 to MFIS_tokens[i].length-1 do
        token1 = get_individual_token(l, MFIS_tokens[i]);
          FOR m from 0 to MFIS_tokens[j].length-1 do
            token2 = get_individual_token(m, MFIS_tokens[j]);
            IF token1 == token2 do
              count = count - 1;
              process_MFIS(m, i, j, l, MFIS_tokens, actual_count);
            End IF
          End FOR
        End FOR
      End FOR
    End FOR
  End IF
  Return count;
```

Figure 5.3: Algorithm for computing the core genes

```
cluster(MFIS_token, count, cl_id, β)
  FOR x from 0 to count-1 do
    tokens[x] = get_individual_token(x, MFIS_token),
    no = tokens[x],
    g_no classified = 1,
    g_no cluster_id = cl_id
  End FOR
  FOR x from 0 to G do
    Flag = 0,
    IF g_x classified == 0 do
      FOR i from 0 to count-1 do
        IF (find_DBK(x, tokens[i]) ≥ β)
          Flag = 1,
        End IF
      End FOR
      IF Flag == 0 do
        g_x classified = 1,
        g_x cluster_id = cl_id,
      End IF
    End IF
  End FOR
```

Figure 5 4 Shared neighbor clustering algorithm

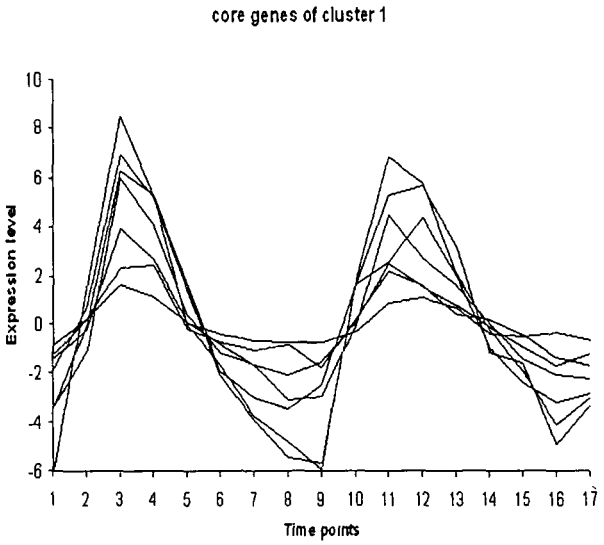


Figure 5.5: The core genes of cluster 1 of Dataset 2

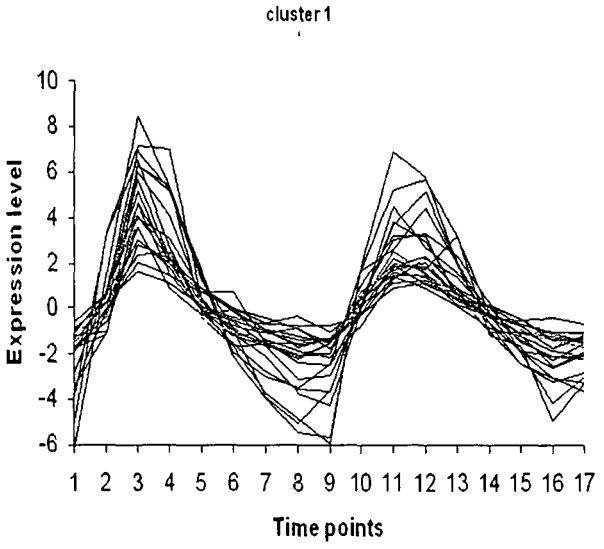


Figure 5.6: Final cluster 1 based on the core genes of Figure 5.5 of Dataset 2

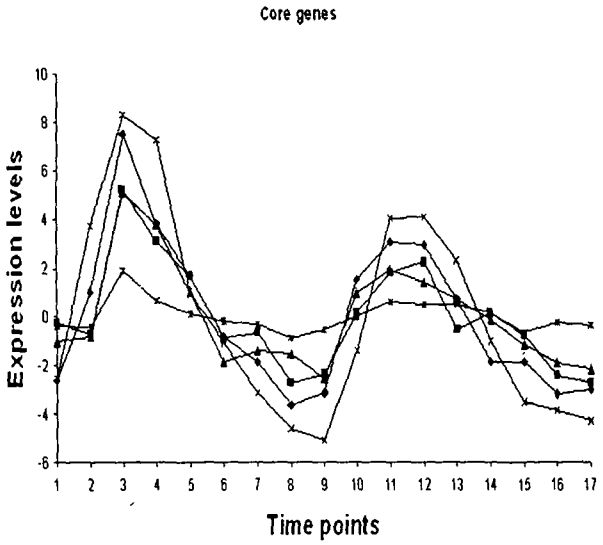


Figure 5.7: The core genes of cluster 2 of Dataset 2

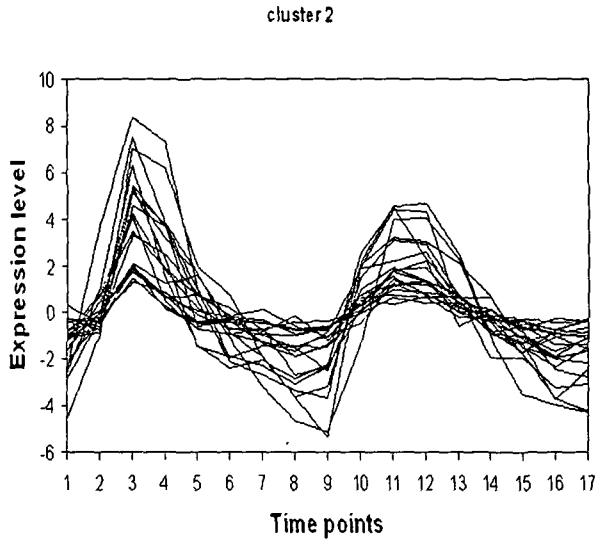


Figure 5.8: The final cluster 2 based on the core genes of Figure 5.7 of Dataset 2

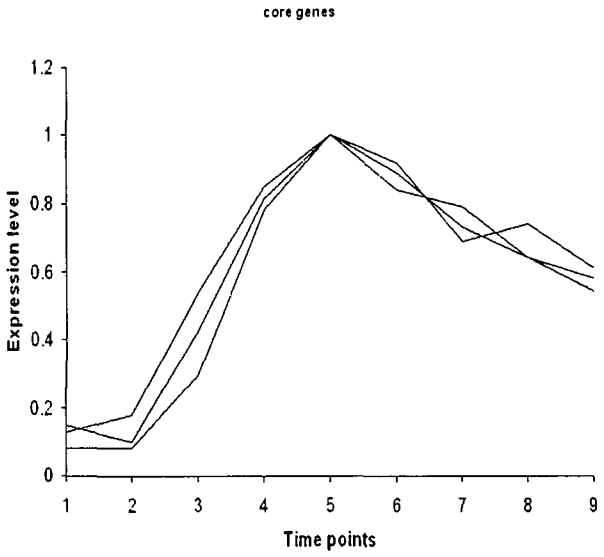


Figure 5.9: The Core genes at s=40% of Dataset 4

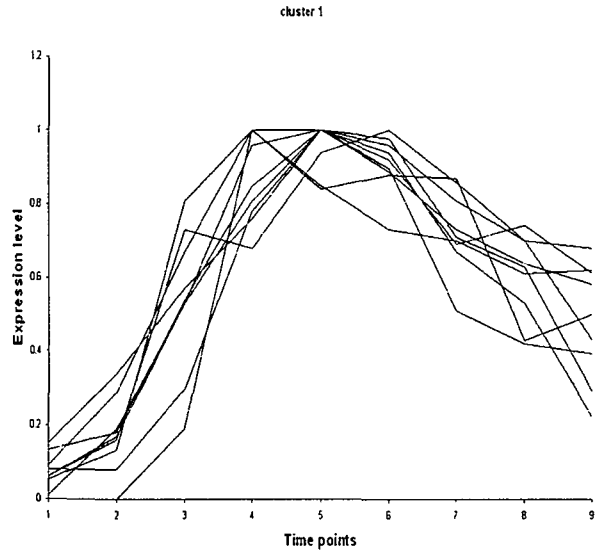


Figure 5.10: The final cluster 1 obtained from the core genes of Dataset 4

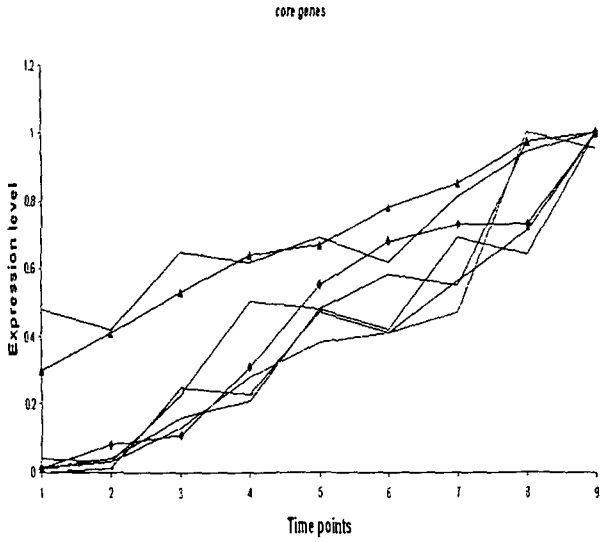


Figure 5.11: The Core genes at s=40% of Dataset 4

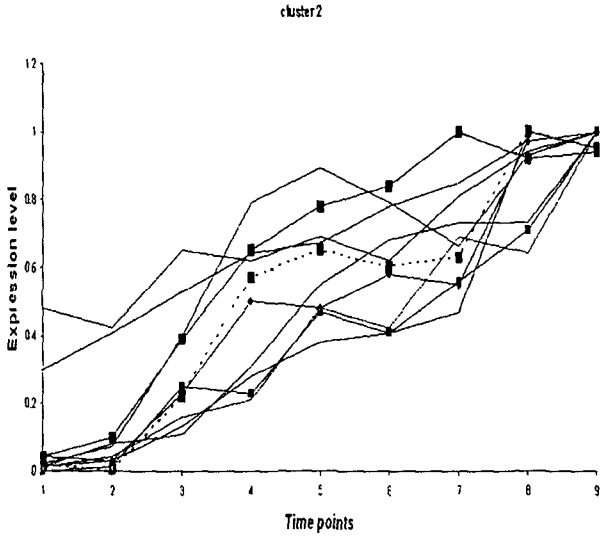


Figure 5.12: The final cluster 2 obtained from the core genes of Dataset 4

Table 5.1: Homogeneity values for FINN and other comparable algorithms

Datasets	Method Applied	No. of Clusters	Threshold value	Homogeneity
Dataset 2	k-means	16	NA	0.671
	SOM	16	4 × 4 grid	0.710
	CLICK	3	Default value	0.549
	DCCA	15	NA	0.818
	FINN	**16**	NA	**0.778**
Dataset 3	k-means	119	NA	0.553
	SOM	47	10 × 12 grid	0.865
	DCCA	4	NA	0.818
	FINN	**119**	NA	**0.844**
Dataset 4	k-means	8	NA	0.781
	SOM	8	2 × 4 grid	0.772
	CLICK	3	Default value	0.676
	DCCA	10	NA	0.834
	FINN	**8**	NA	**0.878**
Dataset 5	k-means	4	NA	0.512
	SOM	4	2 × 2 grid	0.562
	CLICK	6	Default value	0.729
	DCCA	10	NA	0.812
	FINN	**10**	NA	**0.901**
Dataset 6	k-means	4	NA	0.551
	SOM	4	2 × 2 grid	0.553
	CLICK	5	Default value	0.483
	DCCA	10	NA	0.813
	FINN	**4**	NA	**0.898**
Dataset 7	k-means	5	NA	0.577
	SOM	6	2 × 3 grid	0.514
	CLICK	5	Default value	0.501
	DCCA	43	NA	0.699
	FINN	**5**	NA	**0.815**

Table 5.2: Silhouette Index for FINN and other comparable algorithms

Datasets	Method Applied	No. of Clusters	Silhouette Index
Dataset 2	MOGA-SVM (RBF)	5	0.443
	MOGA (without SVM)	5	0.439
	FCM	6	0.387
	Average linkage	4	0.439
	SOM	6	0.368
	DCCA	15	0.838
	FINN	**16**	**0.855**
Dataset 4	MOGA-SVM (RBF)	6	0.451
	MOGA (without SVM)	6	0.487
	FCM	5	0.405
	Average linkage	6	0.412
	SOM	7	0.482
	CLICK	3	0.179
	DCCA	10	0.910
	FINN	**8**	**0.928**
Dataset 5	MOGA-SVM (RBF)	4	0.431
	MOGA (without SVM)	4	0.401
	FCM	4	0.364
	Average linkage	5	0.315
	k-means	10	0.652
	SOM	9	0.536
	CLICK	6	0.449
	DCCA	10	0.609
	FINN	**10**	**0.69**
Dataset 6	MOGA-SVM (RBF)	6	0.415
	MOGA (without SVM)	6	0.395
	FCM	8	0.299
	Average linkage	4	0.356
	SOM	6	0.324
	FINN	**7**	**0.747**

Table 5.3: z-scores for k-means, DCCA, SOM and FINN for Dataset 2

Method Applied	No. of Clusters	z-score
k-means	16	11.6
DCCA	14	6.995
SOM	16	6.46
FINN	16	12.1

Table 5.4: z-scores for UPGMA, k-means, DCCA, SOM and FINN for reduced set of Dataset 3

Method Applied	No. of Clusters	z-score	Total no. of genes
UPGMA	8	8.7	614
k-means	8	9.4	614
DCCA	8	7.22	614
SOM	8	6.6	614
FINN	8	12.12	614

5.5.3 Biological significance

The functional enrichment of each GO category in each of the clusters obtained is calculated by its *p-value*. To compute the p-values for the clusters obtained by FINN, we used the software FuncAssociate [B+03]. To restrict the size of the chapter, we report only functional categories with p-value $< 7 \times 10^{-05}$. The highly enriched categories for the various clusters are shown in Table 5.5. Cluster C1 contains genes involved in ATP activity. Some of the highly enriched functional categories in C1 are 'helicase activity', 'ATP-dependent helicase activity', 'ATPase activity coupled', 'ATPase activity' et al. with p-values of 6.4×10^{-26}, 6.7×10^{-25}, 6.2×10^{-20}, 4.7×10^{-18}, respectively. Cluster C2 contains genes involved in cell cycle. Among the various enriched functional categories in C2, the ones scoring the highest are 'DNA replication', 'DNA metabolic process', 'cell cycle' and 'DNA-dependent DNA replication' with p-values of 1.3×10^{-11}, 2.2×10^{-11}, 1.3×10^{-10} and 4.6×10^{-10} respectively. Cluster C4 consists of genes responsible for cell wall functions. Cluster C5 contains genes involved in different components and functions of nucleus. This cluster also contains highly enriched

GO attributes with 'nuclear nucleosome' scoring the highest in terms of p-value (3.5×10^{-27}). The other enriched attributes are 'nucleosome' with a p-value of 7.7×10^{-26}, 'nucleosome assembly' with a p-value of 7.1×10^{-22}, 'nuclear chromatin' with a p-value of 9.5×10^{-20}, 'chromatin' with a p-value of 1.8×10^{-18} and so on Cluster C6 has various genes contributing to different phases of cell cycle with 'cell cycle' being the highly enriched category with a p-value of 6.3×10^{-11}. From the results of Table 5.5, we arrive at the conclusion that the genes in a cluster obtained by FINN seem to be involved in similar functions.

5.6 Discussion

From our exhaustive experiments with FINN over the datasets mentioned in Table 3.5, we come to the conclusion that by varying the value of β, the quality of the clusters can be further increased. The support count in the frequent itemset generation has a pivotal role in the detection of the core genes. With the increase in the support count, more compact sets of core genes can be obtained. Moreover, for high values of support count, frequent itemset generation also becomes faster. Taking these factors into count, more compact clusters may be obtained. To test the performance of the clustering algorithm, we compare the clusters obtained by our method and those obtained by several other methods, and the result was found satisfactory. The clusters detected by FINN is also found to be biologically significant in terms of p-value.

Table 5.5: P-values of Dataset 7

Cluster	P-value	GO number	GO category
C1	6.4e-26	GO:0004386	helicase activity
	6.7e-25	GO:0008026	ATP-dependent helicase activity
	6.2e-20	GO:0042623	ATPase activity, coupled
	4.7e-18	GO:0016887	ATPase activity
	2.6e-16	GO:0017111	nucleoside-triphosphatase activity
	5.4e-16	GO:0016462	pyrophosphatase activity
	5.4e-16	GO:0016818	hydrolase activity, acting on acid anhydrides, in phosphorus-containing anhydrides
	5.8e-16	GO:0016817	hydrolase activity, acting on acid anhydrides
	7.9e-13	GO:0005524	ATP binding
	8.6e-13	GO:0032559	adenyl ribonucleotide binding
	2e-12	GO:0030554	adenyl nucleotide binding
	5.6e-12	GO:0032553	ribonucleotide binding
	5.6e-12	GO:0032555	purine ribonucleotide binding
	1.2e-11	GO:0017076	purine nucleotide binding
	2.6e-11	GO:0003678	DNA helicase activity
	4.5e-11	GO:0000166	nucleotide binding
	3.5e-11	GO:0000722	telomere maintenance via recombination
	5e-10	GO:0016787	hydrolase activity
	6.7e-08	GO:0006312	mitotic recombination
	8.1e-08	GO:0003676	nucleic acid binding
	8e-07	GO:0003824	catalytic activity
	8.9e-05	GO:0006310	DNA recombination
C2	1.3e-11	GO:0006260	DNA replication
	2.2e-11	GO:0006259	DNA metabolic process
	1.3e-10	GO:0007049	cell cycle
	4.6e-10	GO:0006261	DNA-dependent DNA replication

Cluster	P-value	GO number	GO category
C2	4.2e-08	GO:0051301	cell division
	8.2e-08	GO:0005657	replication fork
	1.3e-07	GO:0022402	cell cycle process
	1.4e-07	GO:0000079	regulation of cyclin-dependent protein kinase activity
	1.6e-07	GO:0006281	DNA repair
	2.2e-07	GO:0022403	cell cycle phase
	7.2e-07	GO:0006298	mismatch repair
	7.2e-07	GO:0045005	maintenance of fidelity during DNA-dependent DNA replication
	9e-07	GO:0006974	response to DNA damage stimulus
	1.1e-06	GO:0005694	chromosome
	1.5e-06	GO:0009719	response to endogenous stimulus
	1.6e-06	GO:0005935	cellular bud neck
	1.8e-06	GO:0043549	regulation of kinase activity
	1.8e-06	GO:0045859	regulation of protein kinase activity
	2.1e-06	GO:0051338	regulation of transferase activity
	2.2e-06	GO:0030894	replisome
	2.2e-06	GO:0043601	nuclear replisome
	6e-06	GO:0006273	lagging strand elongation
	8.9e-06	GO:0006310	DNA recombination
	9.2e-06	GO:0043596	nuclear replication fork
	9.6e-06	GO:0045934	negative regulation of nucleobase, nucleoside, nucleotide and nucleic acid metabolic process
	9.8e-06	GO:0005933	cellular bud
	9.8e-06	GO:0030427	site of polarized growth
	1e-05	GO:0051329	interphase of mitotic cell cycle
	1.1e-05	GO:0051325	interphase
	1.1e-05	GO:0005634	nucleus

Cluster	P-value	GO number	GO category
C2	1.3e-05	GO:0051052	regulation of DNA metabolic process
	1.3e-05	GO:0016538	cyclin-dependent protein kinase regulator activity
	1.6e-05	GO:0044427	chromosomal part
	1.9e-05	GO:0000228	nuclear chromosome
	2.5e-05	GO:0000082	G1/S transition of mitotic cell cycle
	2.8e-05	GO:0003677	DNA binding
	3e-05	GO:0031324	negative regulation of cellular metabolic process
	3e-05	GO:0000278	mitotic cell cycle
	3.1e-05	GO:0009892	negative regulation of metabolic process
	4.3e-05	GO:0032301	MutSalpha complex
	5.9e-05	GO:0006271	DNA strand elongation during DNA replication
	5.9e-05	GO:0022616	DNA strand elongation
	6e-05	GO:0051276	chromosome organization and biogenesis
	6.7e-05	GO:0051726	regulation of cell cycle
C4	1.3e-05	GO:0009277	fungal-type cell wall
	3.1e-05	GO:0005618	cell wall
	3.3e-05	GO:0030312	external encapsulating structure
C5	3.5e-27	GO:0000788	nuclear nucleosome
	7.7e-26	GO:0000786	nucleosome
	7.1e-22	GO:0006334	nucleosome assembly
	9.5e-20	GO:0000790	nuclear chromatin
	1.8e-18	GO:0000785	chromatin
	2.4e-16	GO:0065004	protein-DNA complex assembly

Cluster	P-value	GO number	GO category
C5	2.2e-15	GO:0031497	chromatin assembly
	4.5e-15	GO:0006323	DNA packaging
	5.3e-15	GO:0006333	chromatin assembly or disassembly
	2.4e-14	GO:0044454	nuclear chromosome part
	1.6e-13	GO:0000228	nuclear chromosome
	1.7e-12	GO:0044427	chromosomal part
	4.9e-12	GO:0006325	establishment and/or maintenance of chromatin architecture
	7.5e-12	GO:0005694	chromosome
	1.5e-10	GO:0065003	macromolecular complex assembly
	3.4e-09	GO:0003677	DNA binding
	5.5e-09	GO:0022607	cellular component assembly
	6.5e-09	GO:0051276	chromosome organization and biogenesis
	9e-07	GO:0043228	non-membrane-bounded organelle
	9e-07	GO:0043232	intracellular non-membrane-bounded organelle
	2.6e-06	GO:0044428	nuclear part
	8.6e-06	GO:0043234	protein complex
	1.6e-05	GO:0006281	DNA repair
	1.9e-05	GO:0003676	nucleic acid binding
	2.4e-05	GO:0006996	organelle organization and biogenesis
	2.5e-05	GO:0045816	negative regulation of transcription from RNA polymerase II promoter, global

Cluster	P-value	GO number	GO category
C5	3.7e-05	GO:0006358	regulation of transcription from RNA polymerase II promoter, global
	3.7e-05	GO:0006974	response to DNA damage stimulus
	3.8e-05	GO:0006259	DNA metabolic process
	4.7e-05	GO:0009719	response to endogenous stimulus
	6.8e-05	GO:0016043	cellular component organization and biogenesis
C6	6.3e-11	GO:0007049	cell cycle
	1.2e-10	GO:0000278	mitotic cell cycle
	1.8e-10	GO:0022402	cell cycle process
	1.3e-09	GO:0007067	mitosis
	1.5e-09	GO:0000087	M phase of mitotic cell cycle
	1.9e-09	GO:0022403	cell cycle phase
	9.6e-08	GO:0051301	cell division
	1.5e-07	GO:0000279	M phase
	2.3e-07	GO:0005935	cellular bud neck
	9.7e-07	GO:0005933	cellular bud
	9.7e-07	GO:0030427	site of polarized growth

Chapter 6

Finding Coherent Patterns using a Density Based Approach

This chapter presents a Density based Clustering Algorithm, DGC, for clustering gene expression data. DGC uses a regulation based discretization technique to transform the gene expression data into 3 discrete levels. It then uses the discretized data to find coherent patterns using a density based clustering approach. DGC is independent of any proximity measure, however, we establish the effectiveness of DGC based on DBK measure introduced in Chapter 3.

6.1 Introduction

A cluster can be defined as a region over the gene space, in which the local density is higher than its surrounding region To identify such a region, we need to calculate local densities of genes in space The density of genes is governed by two factors (a) the typical distances among the genes, and (b) the number of neighbors of a gene, indicative of the dimension in which the points are embedded

6.2 Related Work

Density based clustering algorithms identify dense areas in the object space Clusters are hypothesized as high density areas separated by sparsely dense areas

6.2.1 Kernel Density Clustering Method

A kernel density clustering method for gene expression profile analysis is reported in [SZCS03] It assumes no parametric statistical model and does not rely on any specific probability distribution Hyper-spherical uniform kernels of variable radius are used and density estimate of the data points are found The distance between two clusters (or observations) i and j is computed [SAS99] as follows

$$d(x_i, x_j) = \left\{ \begin{array}{l} \frac{1}{2}\left(\frac{1}{f(x_i)} + \frac{1}{f(x_j)}\right) \; if \; d(x_i, x_j) \leq R \\ \infty \; otherwise \end{array} \right\} \qquad (6\ 1)$$

where R is the user-specified radius and $f(x)$ is the estimated density at x [SAS99] The method is robust and less sensitive to outliers However, accurate density estimation and assignment of cluster membership require multiple data points in near-neighborhoods and thus density estimation is less accurate when cluster size is small

6.2.2 Density-based Hierarchical Clustering

In [JPZ03a], the authors propose the Density-based Hierarchical Clustering method (DHC) that uses a density-based approach to identify co-expressed gene groups

from gene expression data It considers clusters as high dimensional dense areas where the genes are attracted to each other DHC uses two-level hierarchical structures (attraction tree and density tree) to organize the cluster structure of the data set The attraction tree reflects relationships among genes in the dense area Each node in the attraction tree represents a gene and its parent is the attractor of it The highest density gene becomes the root of the tree The attraction tree becomes complicated for large datasets and hence the cluster structure is summarized in a density tree Each node of the density tree represents a dense area Initially the whole dataset is considered as a single dense area represented by the root node of the density tree This dense area is then split into several sub-dense areas based on some criteria and each sub-dense area is represented by a child node of the root node The sub-dense areas are further split till each sub-dense area contains a single cluster DHC is suitable for detecting highly connected clusters but is computationally expensive and is dependent on two global parameters

An alternative to this is to define the similarity of points in terms of their shared nearest neighbors This idea was first introduced by Jarvis and Patrick [JP73]

6.2.3 Nearest Neighbor based Density Estimation for Clustering Gene Expression Data

In [CJM04], a k-nearest neighbor based density estimation technique is exploited The density based algorithm proposed by [CJM04] works in three phases density estimation for each gene, rough clustering using core genes and cluster refinement using border genes Density of a gene is calculated by the sum of similarities among its k nearest neighbors Core genes are high density genes and the method proceeds by clustering core genes to form rough clusters Once rough clusters are formed, the border genes are assigned to the most relevant cluster

6.2.4 Clustering based on Density and Shared Nearest Neighbor Measure

In [SAP06], the authors present a density and shared nearest neighbor based clustering method The similarity measure used is that of Pearson's correlation and the density of a gene is given by the sum of its similarities with its neighbors The shared nearest neighbors of the dense genes are found and merged into the same cluster The merging is done efficiently using a data structure called the P-tree [Per01]

6.3 Motivation

Density-based approach discovers clusters of arbitrary shapes even in the presence of noise However, density-based clustering techniques suffer from high computational complexity with increase in dimensionality (even if spatial index structure is used) and input parameter dependency In this chapter, we present a density based clustering method that uses a regulation based cluster expansion process It overcomes the problem of maintaining the pattern information usually linked with the different clustering approaches due to traditional similarity measures The advantage of our method is that it produces quality clustering and can handle noisy datasets

6.4 DenGeneClus (DGC)

DGC works in two phases The first phase normalizes and discretizes the gene expression data with minimum information loss while the second phase is dedicated to clustering the discretized normalized data

6.4.1 Phase I: Normalization and Discretization

This phase is a two step process The first step deals with normalization of the gene expression data to have mean 0 and standard deviation 1 Expression

data having a low variance across conditions as well as data having more than 3-fold variation are filtered out in this step. The second step, i.e., discretization is performed on this normalized expression data where the regulation pattern, i.e., up- or down- regulation in each of the conditions for a particular gene plays an important role. An example of a discretized matrix obtained from the data in Figure 6.1 is shown in Figure 6.2. Discretization is carried out as follows: Suppose, G^* is the set of all genes and T^* is the set of all conditions. Let $\{g_i\} \in G^*$ be the i^{th} gene and $\{t_j\} \in T^*$ be the j^{th} condition. The expression value of gene $\{g_i\}$ at condition $\{t_j\}$ is given by $\varepsilon_{i,j}$. The discretization step gives us the regulation pattern of genes across conditions. For a particular gene, the regulation pattern is computed for all conditions except the first condition based on the previous condition value For the first condition, t_1, its discretized value is directly based on $\varepsilon_{i,1}$. For ${t_{j+1}}^{th}$ condition, its discretized value is computed w.r.t. the ${t_j}^{th}$ condition, i.e., $\varepsilon_{i,j+1}$ and $\varepsilon_{i,j}$. Here, $\varepsilon_{i,1}$ is the expression value for a gene g_i at condition t_1, $\varepsilon_{i,j}$ is the expression value for a gene g_i at condition t_j and similarly, $\varepsilon_{i,j+1}$ is the expression value for a gene g_i at t_{j+1}. While discretizing, following two cases occur:

Case 1: For condition t_1 (i.e., the first condition).

The discretized value of gene g_i at condition, t_1

$$\xi_{i,1} = \begin{cases} 1 & \text{if } \varepsilon_{i,1} > 0 \\ 0 & \text{if } \varepsilon_{i,1} = 0 \\ 2 & \text{if } \varepsilon_{i,1} < 0. \end{cases}$$

Case 2: For the conditions $(T^* - \{t_1\})$:

The discretized value of gene g_i at t_j

$$\xi_{i,j+1} = \begin{cases} 1 & \text{if } \varepsilon_{i,j} < \varepsilon_{i,j+1} \\ 0 & \text{if } \varepsilon_{i,j} = \varepsilon_{i,j+1} \\ 2 & \text{if } \varepsilon_{i,j} > \varepsilon_{i,j+1} \end{cases}$$

where $\xi_{i,j}$ is the discretized value of gene g_i at condition t_j $(j = 1,..(T-1))$. Each gene now has a regulation pattern $(\wp)$ of 0, 1, and 2 across the conditions or time points.

Once $\wp$ of each gene is obtained, the second phase, i.e., the clustering process is initiated.

-0.26188	-0.26188	0.662408	1.097367	0.553668
-1.34928	-1.34928	-0.26188	0.009968	-0.2347
0.281818	0.009968	-0.80558	-0.26188	-0.26188
0.825517	0.825517	0.009968	0.281818	0.281818
-1.51239	-1.51239	-1.07743	0.281818	-1.45802
1.097367	1.097367	1.641067	2.184767	1.097367

Figure 6.1: Example dataset

2	0	1	1	2
2	0	1	1	2
1	2	2	1	0
1	0	2	1	0
2	0	1	1	2
1	0	1	1	2

Figure 6.2: Discretized matrix

6.4.2 Phase II: Clustering of genes

The clustering of genes is initiated with the finding of the maximal matching genes with respect to regulation pattern.

i. A Density Based Notion of Clusters

Clusters consist of genes having similar expression patterns across conditions while *noise genes* are those that do not belong to any of the clusters. The basic idea behind recognizing a cluster is that within each cluster we have a typical density (of genes with similar expression patterns) that is considerably higher than that outside the cluster. Furthermore, the density within the areas of noise is lower than the density in any of the clusters. In the following, we try to formalize this intuitive notion of clusters and noise in a database D_G of genes. The key idea is that for each gene in a cluster, the neighborhood must contain at least σ number of genes that have similar expression pattern (regPattern). The shape of a neighborhood is determined by the choice of a distance function for two genes g_i and g_j, denoted by $D(g_i, g_j)$. Note that our approach works with any distance measure and hence there is provision for selecting the appropriate similarity function for a given application. In this chapter, we give results for our own dissimilarity measure, DBK, discussed in detail in Chapter 3.

ii. Basis of the Clustering Approach

The *regulation matching*, *order preservation* and *proximity* are the three fundamental pillars based on which the clustering technique (DenGeneClus or DGC) is designed.

i. Regulation Matching: For a particular gene g_i, the maximal matching regulation pattern (defined later) is found. All genes having the same maximal matching regulation pattern w.r.t. g_i are grouped into the same cluster.

ii. Order Preservation: We preserve order based on [BDCKY02] in the following way. For a condition set $t \subset T^*$ and a gene $g_i \in G^*$, t can be ordered in a way such that the expression values are in ascending order. In order

ranking, we search for expression levels of genes in ascending order within a cluster. Such a pattern arises, for example, if the experiments in t represent distinct stages in the progress of a disease or in a cellular process and the expression levels of all genes in a cluster vary across the stages in the same way [BDCKY02].

Each gene has a rank (Rank) which gives the permutation order of that gene across conditions t. The rank is calculated according to the expression values of a gene across conditions. In other words, the elements of the rank pattern are given by their rank in ascending order of their expression values.

iii. Proximity: The proximity between any two genes g_i and g_j is given by $D(g_i, g_j)$ where D is any proximity measure such as Euclidean distance, Pearson's Correlation and DBK.

The identification of clusters is based on the following definitions. The definitions are given based on the notion of density available in [EKSX96].

Definition 6.1. Match
Let $\wp_{g_i}$ and $\wp_{g_j}$ be the regulation patterns of two genes g_i and g_j. Then, the match (M) between g_i and g_j is given by the number of agreements $No\_Agreements$ (i.e., the number of condition-wise common regulation values excluding condition 1) between the two regulation patterns, i.e.,

$$M(g_i, g_j) = No\_Agreements(\wp_{g_i}, \wp_{g_j}).$$

Definition 6.2. Maximal Match
Gene g_i is referred to as maximally matched (MM) with gene g_j, if the number of agreements between $(\wp_{g_i}, \wp_{g_j}) \geq \delta$ where $g_j \in \{G^* - g_i\}$ and δ is a user-defined threshold.

Definition 6.3. Maximal Matching Regulation Pattern
If a gene g_i maximally matches with gene g_j, then the regulation pattern $\wp'_{g_i}$ and $\wp'_{g_j}$ formed by taking the subset of conditions where both $\wp_{g_i}$ and $\wp_{g_j}$ match is referred to as the Maximal Matching Regulation Pattern (MMRP).

MMRP of genes g_i and g_j is computed as follows.

$$\wp'_{g_i} = \wp'_{g_j} = \begin{cases} 1 & \text{if } \wp_{g_i,t} = \wp_{g_j,t} = 1 \\ 0 & \text{if } \wp_{g_i,t} = \wp_{g_j,t} = 0 \\ 2 & \text{if } \wp_{g_i,t} = \wp_{g_j,t} = 2 \\ x & \text{otherwise.} \end{cases}$$

Here t refers to the conditions $(t = 2, 3, \cdots, T-1)$.
Each gene has a rank which gives the permutation order of that gene across conditions $t \subset T^*$. The rank is calculated according to the expression values of a gene across conditions, *i.e.*, the elements of the rank pattern are given by their ranking in ascending order of their expression values. The rank of a gene is calculated as follows:

1. For a gene g_i, find $\wp'_{g_i}$.
2. Rank g_i in ascending order according to the expression values where $\wp'_{g_i,t} \neq x$.

To understand the rank computation, let us refer to the example given in figure 6.1. Here, the rows represent the genes $g_1, g_2, \cdots, g_6$ and the columns represent the corresponding conditions (excluding condition 1 as stated before).

$$\wp_{g_1} = 2 \quad 0 \quad 1 \quad 1 \quad 2$$

$$\wp_{g_2} = 2 \quad 0 \quad 1 \quad 1 \quad 2$$

$$\wp_{g_3} = 1 \quad 2 \quad 2 \quad 1 \quad 0$$

$$\wp_{g_4} = 1 \quad 0 \quad 2 \quad 1 \quad 0$$

$$\wp_{g_5} = 2 \quad 0 \quad 1 \quad 1 \quad 2$$

$$\wp_{g_6} = 1 \quad 0 \quad 1 \quad 1 \quad 2$$

Matching among pairs of genes are given below.

$$M(g_1, g_2) = 4 \qquad M(g_1, g_3) = 1 \qquad M(g_1, g_4) = 2$$

$$M(g_1, g_5) = 4 \qquad M(g_1, g_6) = 4 \qquad M(g_2, g_3) = 1$$

$$M(g_2, g_4) = 2 \qquad M(g_2, g_5) = 4 \qquad M(g_2, g_6) = 4$$

$$M(g_3, g_4) = 3 \qquad M(g_3, g_5) = 1 \qquad M(g_3, g_6) = 1$$

$$M(g_4, g_5) = 2 \qquad M(g_4, g_6) = 2 \qquad M(g_5, g_6) = 4$$

Suppose $\delta = 3$, then Maximal Matching of pairs of genes are as follows.

$$MM(g_1, g_2) = 4 \qquad MM(g_1, g_5) = 4 \qquad MM(g_1, g_6) = 4$$

$$MM(g_2, g_5) = 4 \qquad MM(g_2, g_6) = 4$$

$$MM(g_3, g_4) = 3 \qquad MM(g_5, g_6) = 4$$

Thus, MMRP is given below.

$$\wp'_{g_1} = 0 \quad 1 \quad 1 \quad 2$$

$$\wp'_{g_2} = 0 \quad 1 \quad 1 \quad 2$$

$$\wp'_{g_5} = 0 \quad 1 \quad 1 \quad 2$$

$$\wp'_{g_6} = 0 \quad 1 \quad 1 \quad 2$$

$$\wp'_{g_3} = x \quad 2 \quad 1 \quad 0$$

$$\wp'_{g_4} = x \quad 2 \quad 1 \quad 0$$

From the example given above, it is clear that the MMRP of g_1, g_2, g_5 and g_6 are same, as well as the MMRP of g_3 and g_4 are same.

Genes 1, 2, 5 and 6 have the MMRP over conditions 2, 3, 4, 5. Rank order over these four conditions are computed w.r.t. their expression values ($\varepsilon_{i,j}$, $i = 1, 2, 5, 6$ and $j = 2, 3, 4, 5$, where i refers to gene i and j refers to condition j) and ranks as follows.

$$Rank(g_1) = 1 \quad 3 \quad 4 \quad 2$$

$$Rank(g_2) = 1 \quad 2 \quad 3 \quad 4$$

$$Rank(g_5) = 1 \quad 3 \quad 4 \quad 2$$

$$Rank(g_6) = 1 \quad 2 \quad 3 \quad 1$$

Similarly, genes 3 and 4 can be found to have the MMRP over conditions 3, 4, 5 and ranks obtained are as follows.

$$Rank(g_3) = 1 \quad 2 \quad 2$$

$$Rank(g_4) = 1 \quad 2 \quad 2$$

Definition 6.4. θ-neighborhood
The θ-neighborhood of a gene g_i, denoted by $N_\theta(g_i)$ is defined by

$$N_\theta(g_i) = \{g_j \in G^* \mid D(g_i, g_j) \leq \theta\}$$

where D may be any distance measure such as Euclidean, Pearson's correlation and our dissimilarity measure, DBK.

Definition 6.5. Core Gene
A gene g_i is said to be a core gene w.r.t. θ if there is at least one gene g_j such that

i. $g_j \in N_\theta(g_i)$,

ii. $\mid N_\theta(g_i) \mid \geq \sigma$,

iii. $Rank(g_i) \approx Rank(g_j)$, and

iv. $\wp'_{g_i} \approx \wp'_{g_j}$

where σ is a user defined threshold for the minimum number of genes in the θ-neighborhood of g_i.

Definition 6.6. Directly Reachable Gene
A gene g_i is directly reachable from gene g_j w.r.t. θ if

i. g_j is a core gene,

ii. $g_i \in N_\theta(g_j)$, and

iii. $\wp'_{g_i} \approx \wp'_{g_j}$.

Direct reachability relation of a gene is symmetric for pairs of core genes. However, in case of a pair consisting of a core and a non-core gene, it may not be symmetric.

Definition 6.7. Reachable Gene
A gene p is said to be reachable from gene q w.r.t. θ, if there is a chain of genes $P_1, P_2, \cdots, P_n$, where $P_1 = q$, $P_n = p$ such that P_{i+1} is direct reachable from P_i.

Thus, reachability relation is a canonical extension of direct reachability [EKSX96]. This relation is transitive, but not symmetric. However, over this gene expression domain reachability is symmetric for core genes.

Definition 6.8. Connected Genes
A gene g_i is said to be connected to another gene g_j if both g_i and g_j are reachable from another gene g_q w.r.t. θ.

Connectivity is a symmetric relation.

Definition 6.9. Cluster
A cluster C w.r.t. θ is a non-empty subset of G^* and $\mid C \mid \geq \sigma$ (σ is a user-defined threshold) satisfying the following conditions:

i. $\forall g_i, g_j$ if $g_i \in C$ and g_j is reachable from g_i w.r.t. θ then, $g_j \in C$(reachability).

ii. $\forall g_i, g_j \in C : g_i$ is connected to g_j w.r.t. θ (connectivity).

Therefore, a cluster can be defined as a set of reachable and/or connected genes.

Definition 6.10. Noise
Let C be the set of clusters w.r.t. parameter θ. Noise is defined as the set of genes not belonging to any cluster $C_i \in C$. In other words,

$$noise = \{g_i \in G^* \mid \forall i : g_i \notin C_i\}.$$

Also, a gene g_i is said to be a noise gene if it does not satisfy the θ-neighborhood condition i.e.,

$$\mid N_\theta(g_i) \mid = \phi$$

Note that any cluster C_i w.r.t. θ contains at least two genes (i.e. $\sigma = 2$) to satisfy the core gene condition.

iii. Identifying core genes

The clustering process starts with the identification of Core genes according to Definition 6.5. Cluster expansion starts from a core gene and finds all reachable genes from it.

iv. Finding maximal coherent clusters

Cluster identification starts with an arbitrary gene and finds the MMRP with other unclassified genes. For regulation pattern matching, two genes are matched w.r.t. regulation across conditions starting from condition 2. Condition 1 is not considered since its regulation is w.r.t. the expression level rather than the previous condition. If the arbitrary gene is a core gene, cluster expansion proceeds with this core gene, and finds reachable and connected genes from this core gene. All reachable and connected genes in a particular iteration of the clustering process are grouped into the same cluster. The process then recursively continues until all genes are classified. This expansion process can be summarized in terms of the following steps.

i. Start with an arbitrary unclassified gene g_i and find its rank order and regulation pattern.

ii. Call $get\_Core(g_i)$.

iii. For each core gene g_i.

    a) Find all reachable and connected genes w.r.t. g_i.
    b) Classify all those genes with the same Cluster-id.

iv. Repeat steps ii. - iii.until no more core gene is found.

v. Repeat steps i. to iv. till all genes are classified.

The cluster creation process is given in Figure 6.3 and the cluster expansion process is given in Figure 6.4. Here, $get\_core(g_i)$ is a function which checks the core condition as stated in Definition 6.5 and returns a true value if g_i is core; otherwise, it returns a false value. Assuming G^* be the set of genes and C the

DGC_cluster_creation()

Precondition: All genes in D_G are unclassified

```
FOR all g_i ∈ G do
  Compute ℘(g_i);
END FOR
FOR i = 0 to G do
  IF g_i.classified ≠ CLASSIFIED then
    Compute ℘'(g_i) & Rank(g_i);
    IF get_core(g_i) == TRUE then
      expand_cluster(g_i, cluster_id);
      cluster_id = cluster_id + 1;
    END IF
  END IF
END FOR
```

Figure 6.3: Algorithm for cluster formation

expand_cluster(g_i, cluster_id)

```
IF g_i.classified == CLASSIFIED then
  RETURN;
END IF
g_i.classified = CLASSIFIED;
g_i.cluster_id = cluster_id;
FOR j = 0 to G do
  IF g_i ≠ g_j
    IF ℘'_{g_i} ≈ ℘'_{g_j} && g_j ∈ N_θ(g_i) then
      IF get_core(g_j) == TRUE then
        expand_cluster(g_j, cluster_id);
      END IF
      g_j.classified = CLASSIFIED;
      g_j.cluster_id = cluster_id;
    END IF
  END IF
END FOR
```

Figure 6.4: Algorithm for cluster expansion

set of clusters, the following lemmas are introduced to provide the basis of our clustering algorithm. Intuitively they state that given the parameter θ, we can discover a cluster in a two-step approach. First, an arbitrary gene which satisfies the core gene condition is chosen as the seed. Second, all genes reachable from the seed are retrieved. These two steps result in a cluster containing the seed.

Lemma 6.1. Let g_i be a core gene in G, then the set $X = \{x | x \in G^*$ and x is reachable from g_i w.r.t. $\theta\}$ is a cluster w.r.t. θ.

Proof. Let $g_i \in G^*$ be a core gene and let $g_i \in C_i$ (where C_i is a cluster). Again, let $x \in G^*$ be any gene in C_i, which is not reachable from g_i. However, as per Definition 6.9 (condition i.), x must be reachable from g_i. Therefore it contradicts, hence the proof. □

Lemma 6.2. Assume genes g_i, $g_j \in G^*$ and C_1, C_2 are two clusters where $g_i \in C_1$ and $g_j \in C_2$, then g_i and g_j are not connected.

Proof. Consider genes $g_i \in C_1$ and $g_j \in C_2$ where g_i and g_j are connected. Then as per Definition 6.9 (condition ii.), g_i and g_j must belong to the same cluster. Thus we come to a contradiction and hence the proof. □

Lemma 6.3. Let gene $g_i \in G^*$ and C be the set of all clusters. If g_i is a noise gene, then $g_i \notin C$.

Proof. Let gene $g_i \in G^*$ be a noise gene and let $g_i \in C_i$ where $C_i \in C$. Now, g_i must be reachable w.r.t. θ from at least one core gene g_q where $g_q \in C_i$ as per Definition 6.9 (reachability condition). But this violates the noise condition as defined in Definition 6.10 and hence the proof. □

Observation 1. Any core gene $g_i \in C_k$ (where $i = 1, 2, \cdots, m$ and C_k is a cluster) w.r.t. θ have the same MMRP and rank with the other core genes in C_k.

Observation 2. All genes in a cluster C_k have same MMRP with the core gene(s) $\in C_k$.

The result of clustering using DGC with our dissimilarity measure is reported in Section 6.5.1

6.5 Performance Evaluation

We implement DGC in Java in the Windows environment. We use the real-life datasets given in Table 3.5 to evaluate the methods.

6.5.1 Results

We exhaustively test DGC on the given datasets with $\sigma = 2$. The value of σ is taken to be 2 since we search exhaustively for the different patterns. We use Euclidean distance and our dissimilarity measure, DBK, for D and the value of $\theta = 2$. On experimentation with various real-life and synthetic datasets, the method is found to detect biologically significant clusters. We compare our algorithm with that of the k-means, hierarchical clustering (UPGMA), CLICK, DCCA, SOM algorithms. The k-means and UPGMA algorithms are evaluated using the built-in MATLAB implementation. CLICK and SOM algorithms are run using the implementation provided by the Expander tool [SMKS03]. CLICK is run with the default parameter provided by Expander. Expander is also used for finding the homogeneity of k-means clustering. For k-means, k is varied from 2 to 30 by increments of two. The results obtained by our method over a reduced form of Dataset 3 are shown in Figure 6.5. The dataset is reduced by filtering out low variance and low entropy genes from the data. We note here that the clusters obtained by our algorithm are detected automatically and unlike k-means no input parameter for number of clusters is needed. We test k-means with $k = 16, 20, 30, 40, 48$. Since our method gives a total of 47 clusters (when Euclidean distance is used) and 44 clusters (when our dissimilarity measure is used) for the reduced form of Dataset 3, we also test k-means algorithm for k = 44 and 47 respectively. Similarly, UPGMA algorithm is tested for cutoff = 43, 44, 47 and also for other values. Some of the clusters obtained by our method over full Dataset 3 are shown in Figure 6.6. A total of 118 clusters are generated from the full Dataset 3. In Figure 6.7 the clusters generated by k-means on the reduced form of Dataset 3 are given. In Figure 6.8 and Figure 6.9, clusters generated from the reduced form and full form of Dataset 3 using UPGMA at cutoff= 46 and 176 are shown, respectively. In Figure 6.10 some of the clusters

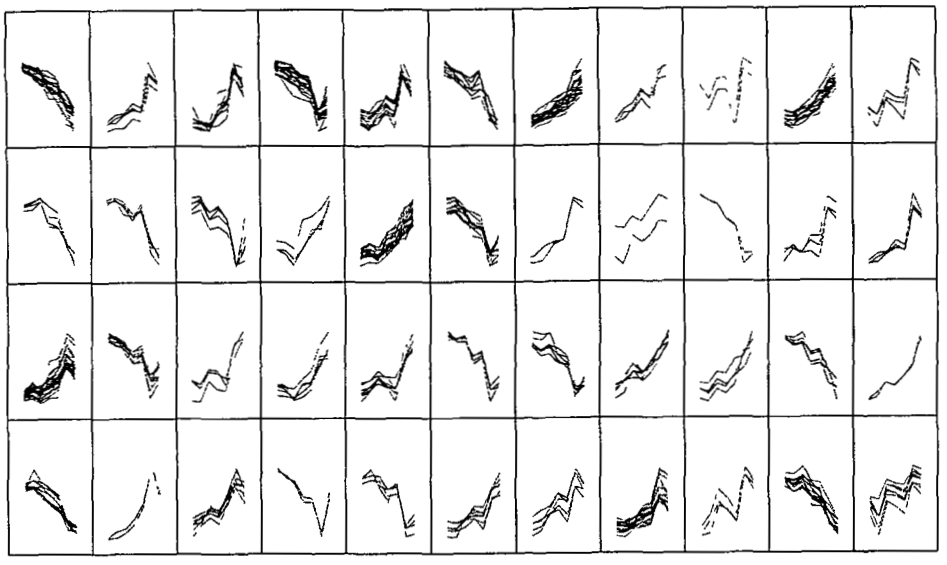


Figure 6.5: Result of DGC on the reduced form of Dataset 3 using our dissimilarity measure

generated from the full Dataset 2 using DGC method (with DBK measure) are shown and in Figure 6 11 the clusters identified from Dataset 6 using DGC (with DBK measure) are shown. Finally, to validate the cluster results, three cluster validity measures, viz., z-score, homogeneity and silhouette index are used and the results were compared with the different clustering algorithms.

6.5.2 Cluster Quality

In this section the performance of DGC is demonstrated on the six publicly available benchmark microarray data sets stated earlier in Table 3.5. We report a comparative study of several widely used microarray clustering algorithms. The performance of DGC is judged by Silhouette index [Rou87] and Average Homogeneity score [SS00]. Table 6.1, Table 6.2 and Table 6.4 respectively show the homogeneity and silhouette values for the different clustering algorithms on the real-life datasets. It can be seen from Table 6 1 and Table 6.2 that the homogeneity values of DGC is superior for all the datasets. Table 6.4 shows that the silhouette values of DGC is superior to those obtained by other methods.

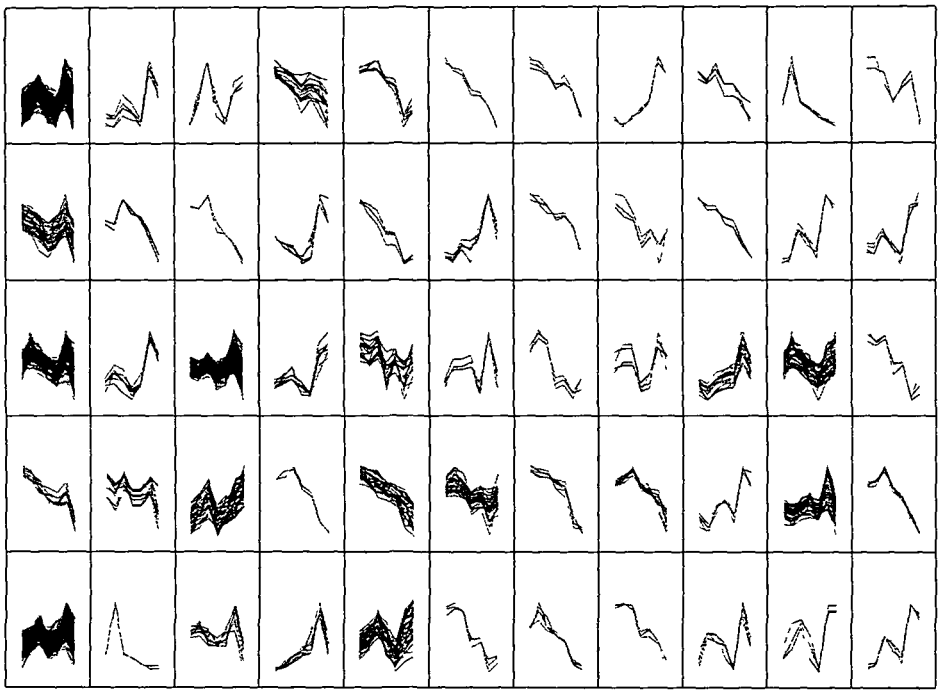


Figure 6.6: Result of DGC on the full Dataset 3 using our dissimilarity measure

However, for Dataset 4 the silhouette value of DGC is less than DCCA. This is because DCCA has two more clusters than DGC. DGC detects eight clusters in the Dataset which is also the number of clusters detected in [WFM+98]. Moreover, the z-score value of DGC for the Dataset 2 is more than DCCA. This concludes that the clusters detected by DGC is more relevant than DCCA.

For validating DGC, we employ z-score [GR02] as the measure of agreement. A higher value of z indicates that genes are better clustered by function, indicating a more biologically relevant clustering result. Z-score [GR02] is calculated by investigating the relation between a clustering result and the functional annotation of the genes in the cluster. In this section, the reported z-score is averaged over 50 repeated experiments. The result of applying the z-score on the reduced form of Dataset 3 is shown in Table 6.5. Table 6.5 clearly shows that our method outperforms k-means, DCCA and SOM w.r.t. cluster quality. Table 6.6 shows the z-score values when our method, DGC, is executed at different values of θ. It is observed that DGC gives better clustering at $\theta = 2$ for the full Dataset 3.

Table 6.1: Homogeneity values for DGC and other comparable algorithms for Datasets 2, 3 and 4.

Datasets	Method Applied	No. of Clusters	Threshold value	Homogeneity
Dataset 2	k-means	4	NA	0.553
	k-means	5	NA	0.591
	k-means	6	NA	0.601
	k-means	16	NA	0.771
	k-means	29	NA	0.787
	k-means	30	NA	0.8
	SOM	4	2×2 grid	0.624
	SOM	9	3×3 grid	0.723
	SOM	25	7×7 grid	0.792
	SOM	41	8×8 grid	0.840
	SOM	33	10×10 grid	0.823
	CLICK	3	Default value	0.549
	DCCA	15	NA	0.818
	DGC	**17**	$\theta = \mathbf{2}$	**0.877**
Dataset 3	k-means	119	NA	0.553
	SOM	47	10×12 grid	0.865
	DCCA	4	NA	0.818
	DGC	**119**	$\theta = 2$	**0.887**
Dataset 4	k-means	8	NA	0.781
	SOM	4	2×2 grid	0.668
	SOM	6	2×3 grid	0.753
	SOM	8	2×4 grid	0.772
	CLICK	3	Default value	0.676
	DCCA	10	NA	0.834
	DGC	**8**	$\theta = 4$	**0.928**

Table 6.2: Homogeneity values for DGC and other comparable algorithms for Datasets 5, 6 and 7.

Datasets	Method Applied	No. of Clusters	Threshold value	Homogeneity
Dataset 5	k-means	4	NA	0.512
	k-means	6	NA	0.680
	k-means	10	NA	0.755
	SOM	4	2×2 grid	0.562
	SOM	6	2×3 grid	0.641
	SOM	10	2×5 grid	0.775
	CLICK	6	Default value	0.728
	DCCA	10	NA	0.825
	DGC	**10**	$\theta = 0.87$	**0.901**
Dataset 6	k-means	6	NA	0.633
	k-means	10	NA	0.682
	k-means	14	NA	0.769
	SOM	6	2×3 grid	0.707
	SOM	9	3×3 grid	0.571
	SOM	14	2×7 grid	0.775
	CLICK	5	Default value	0.483
	DCCA	10	NA	0.813
	DGC	**9**	$\theta = 1.5$	**0.928**
	DGC	**14**	$\theta = 1$	**0.989**
Dataset 7	k-means	7	NA	0.507
	SOM	4	2×2 grid	0.453
	SOM	6	2×3 grid	0.514
	SOM	9	3×3 grid	0.535
	CLICK	5	Default value	0.501
	DCCA	43	NA	0.699
	DGC	**7**	$\theta = 5$	**0.883**
	DGC	**5**	$\theta = 4.5$	**0.906**

Table 6.3: Silhouette Index for DGC and other comparable algorithms for Datasets 2 and 4.

Datasets	Method Applied	No. of Clusters	Silhouette Index
Dataset 2	MOGA-SVM (RBF)	5	0.443
	MOGA (without SVM)	5	0.439
	FCM	6	0.387
	Average linkage	4	0.439
	SOM	6	0.368
	DCCA	15	0.838
	DGC at $\theta = 2$	**17**	**0.851**
Dataset 4	MOGA-SVM (RBF)	6	0.451
	MOGA (without SVM)	6	0.487
	FCM	5	0.405
	Average linkage	6	0.412
	SOM	6	0.482
	SOM	8	0.357
	k-means	8	0.554
	CLICK	3	0.179
	DCCA	10	0.910
	DGC at $\theta = 4$	**8**	**0.777**

Table 6.4: Silhouette Index for DGC and other comparable algorithms for Datasets 5, 6 and 7.

Datasets	Method Applied	No. of Clusters	Silhouette Index
Dataset 5	MOGA-SVM (RBF)	4	0.431
	MOGA (without SVM)	4	0.401
	FCM	4	0.364
	Average linkage	5	0.315
	k-means	10	0.652
	SOM	10	0.536
	CLICK	6	0.449
	DCCA	10	0.609
	DGC at $\theta = 0.87$	**10**	**0.871**
Dataset 6	MOGA-SVM (RBF)	6	0.415
	MOGA (without SVM)	6	0.395
	FCM	8	0.299
	Average linkage	4	0.356
	SOM	6	0.324
	DGC at $\theta = 1$	**14**	**0.9**
	DGC at $\theta = 1.5$	**9**	**0.722**
Dataset 7	k-means	7	0.361
	CLICK	5	0.077
	SOM	6	0.51
	DCCA	43	0.908
	DGC at $\theta = 5$	**7**	**0.524**

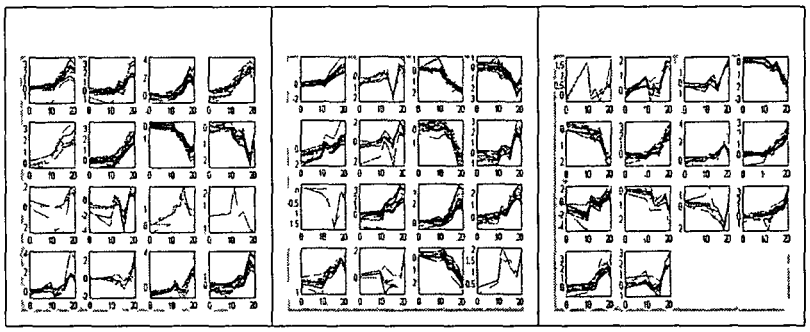


Figure 6 7 Result of k-means on the reduced form Dataset 3 at cutoff = 46

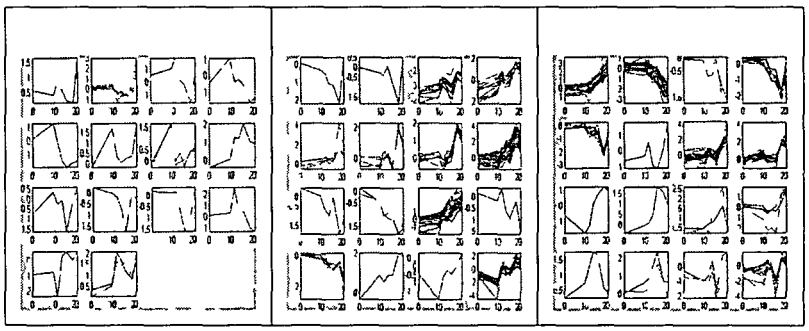


Figure 6 8 Result of UPGMA on the reduced form Dataset 3 at cutoff = 46

The z-score values of several clustering algorithms over the full Dataset 3 are reported in Table 6 7 From the table we conclude that DGC performs better than k-means, hierarchical clustering and DCCA We note here that unlike k-means our method does not require the number of clusters as an input parameter It detects the clusters present in the dataset automatically and gives the rest as noise Also, UPGMA requires the parameter cutoff as input to the algorithm

The z-score values of DGC for datasets 2 and 7 is compared with DCCA and other algorithms and is shown in Table 6 8 and Table 6 9 respectively It is observed from them that DGC performs better than its competitors in terms of z-score

Since ClusterJudge can operate only on yeast datasets, we could not find the z-score values for Datasets 4, 5 and 6 Therefore, we could not compare the z-

Table 6.5: z-scores for DGC and other methods for the reduced form of Dataset 3.

Method Applied	No. of Clusters	z-score
k-means	44	8.6
DCCA	2	-0.995
SOM	35	4.46
DGC at $\theta = 2$	**44**	**12.6**

Table 6.6: z-scores for DGC at different values of θ for the full Dataset 3.

DGC at	No. of Clusters	z-score
$\theta = 0.7$	176	8
$\theta = 1$	128	9.6
$\theta = 1.5$	120	10.6
$\theta = 2$	**119**	**13.2**
$\theta = 2.7$	121	12.9
$\theta = 3.2$	120	11.3
$\theta = 3.7$	120	12.5
$\theta = 4.7$	120	10.5

Table 6.7: z-scores for UPGMA, k-means, DCCA and DGC for the full Dataset 3.

Method Applied	No. of Clusters	z-score	Total no. of genes
UPGMA	119	9.7	6089
k-means	119	8.6	6089
DCCA	4	12.1	6089
DGC at $\theta = 0.7$	177	9.12	6089
DGC at $\theta = 1$	129	7.02	6089
DGC at $\theta = 1.5$	121	11.2	6089
DGC at $\theta = 2$	**119**	**13.8**	**6089**
DGC at $\theta = 2.7$	121	11.2	6089

Table 6.8: z-scores for DCCA, k-means, SOM and DGC for the Dataset 2.

Method Applied	No. of Clusters	z-score	Total no. of genes
DCCA	15	4.41	384
k-means	15	4.48	384
k-means	17	4.56	384
SOM	15	4.66	384
SOM	18	5.11	384
DGC at $\theta = 2$	**17**	**5.51**	**384**

Table 6.9: z-scores for DCCA, k-means, CLICK, SOM and DGC for the Dataset 7.

Method Applied	No. of Clusters	z-score	Total no. of genes
DCCA	43	7.06	698
k-means	7	10.8	698
CLICK	5	11.3	698
SOM	4	11.9	698
SOM	6	11.7	698
SOM	9	10.7	698
DGC at $\theta = 4.5$	7	11.747	698
DGC at $\theta = 5$	**7**	**16.55**	**698**

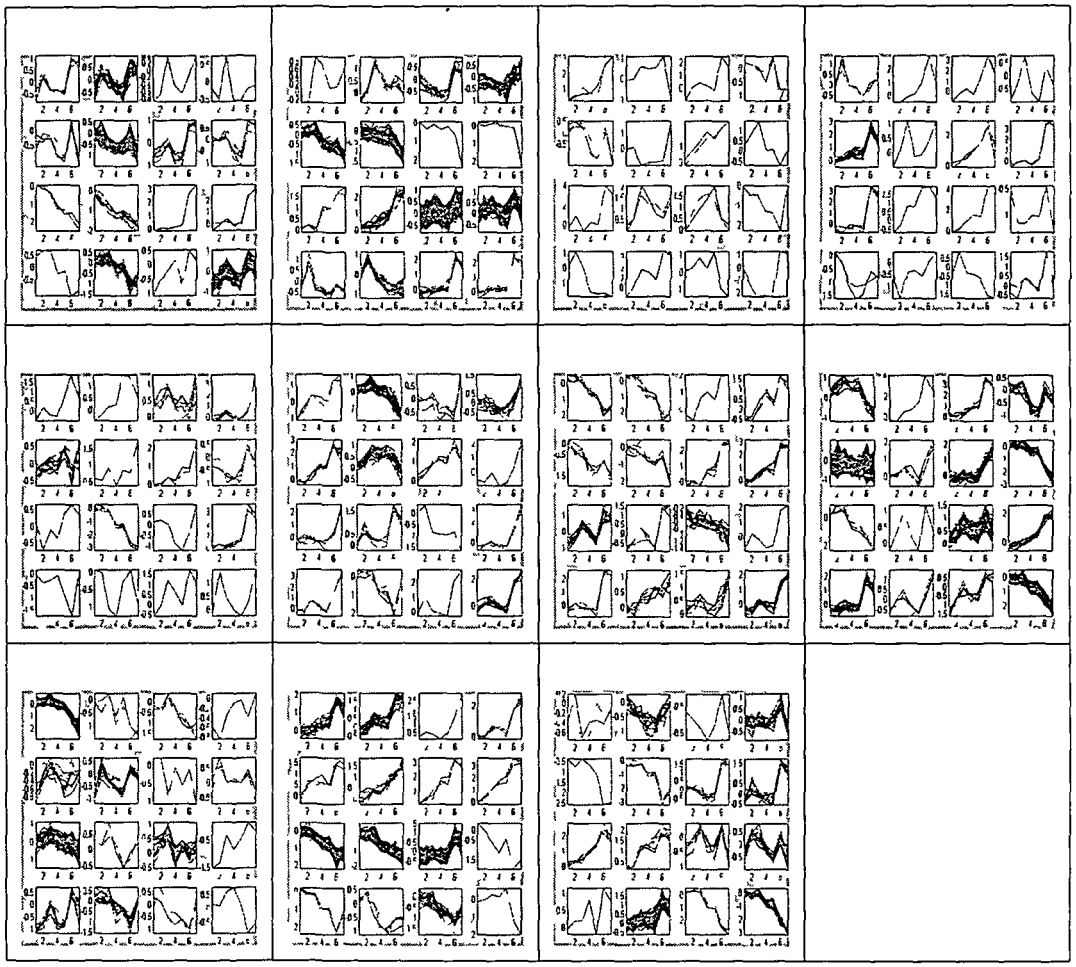


Figure 6 9 Result of UPGMA on the full Dataset 3 at cutoff = 176

score values of DGC with those of clusters generated by other methods for these three datasets

6.5.3 Biological significance

The functional enrichment of each GO category in each of the clusters obtained is calculated by its *p-value* To compute the p-value, we use the software FuncAssociate [B+03] We report functional categories with p-value $< 7 \times 10^{-05}$ in order to restrict the size of the chapter Of the seventeen clusters obtained from Dataset

2, the highly enriched categories are shown in Table 6.10. As can be seen in cluster C3, the highly enriched categories of 'cell cycle', 'DNA metabolic process', 'DNA replication' and 'chromosome' have p-values of 1.1×10^{-25}, 7.6×10^{-22}, 4.1×10^{-21}, 3.4×10^{-18}, respectively. The highly enriched categories in cluster C6 are the 'cellular bud' and the 'cell cycle' with p-values of 3×10^{-12} and 8.4×10^{-12}, respectively. The genes in clusters C3 and C6 are involved in cell cycle. Cluster C9 have genes involved in DNA replication. The cluster C10 contains highly enriched categories such as 'spindle', 'cytoskeletal part' and 'microtubule cytoskeleton' with p-values of 1.3×10^{-12}, 1.8×10^{-11}, 1.9×10^{-11}, respectively. C10 contains genes whose functions are related to various phases of cell cycle. Cluster C13 contains genes belonging to various functional categories related to synthesis of various amino acids. From the results of Table 6.10, we see that the clusters obtained by DGC shows a high enrichment of functional categories.

We also compare the results obtained by DGC with those by DCCA. To restrict the size, we report the p-values of only one of the clusters obtained from DCCA in Table 6.11. This cluster corresponds to the cluster C10 of DGC reported in Table 6.10. We observe that the p-values for the functional categories of DGC are smaller than those obtained by DCCA. For example, the enriched category 'spindle' has a p-value of 1.3×10^{-12} in the result of DGC whereas for DCCA, the value is 9.0×10^{-12}. Similarly, for the GO attribute 'microtubule cytoskeleton', DGC obtains a p-value of 1.9×10^{-11} whereas DCCA obtains 1.2×10^{-10}. Also, for 'cytoskeleton', the p-value of DGC is 8.9×10^{-11} and for DCCA it is 1.7×10^{-07}. This trend continues for other GO attributes as well. Therefore, we can conclude that the clustering solution obtained by DGC is more biologically significant than that of DCCA.

6.6 Discussion

This chapter presents a regulation based density pattern matching approach that does not require the number of clusters as an input parameter. The clusters obtained by DGC on the six microarray data sets have been found to be functionally enriched based on the p-values. The clusters obtained have been validated

Table 6 10 p-value of Dataset 2

Cluster	P-value	GO number	GO category
C1	6 4e-06	GO 0051301	cell division
	1 2e-05	GO 0019887	protein kinase regulator activity
	1 3e-05	GO 0019207	kinase regulator activity
C2	5 7e-06	GO 0005933	cellular bud
	5 6e-05	GO 0044429	mitochondrial part
	6 6e-05	GO 0031966	mitochondrial membrane
C3	1 1e-25	GO 0007049	cell cycle
	7 6e-22	GO 0006259	DNA metabolic process
	4 1e-21	GO 0006260	DNA replication
	3 4e-18	GO 0005694	chromosome
	2 2e-17	GO 0044427	chromosomal part
	5 6e-17	GO 0006261	DNA-dependent DNA replication
	1e-16	GO 0022402	cell cycle process
	3 6e-15	GO 0022403	cell cycle phase
	8 7e-15	GO 0006281	DNA repair
	1 5e-14	GO 0000278	mitotic cell cycle
	2 6e-14	GO 0005657	replication fork
	8 6e-14	GO 0006974	response to DNA damage stimulus
	2 5e-13	GO 0000228	nuclear chromosome
	3 1e-13	GO 0009719	response to endogenous stimulus
	1 9e-12	GO 0045934	negative regulation of nucleobase, nucleoside, nucleotide and nucleic acid metabolic process
	1 9e-11	GO 0051276	chromosome organization and biogenesis
	2 1e-11	GO 0006273	lagging strand elongation

Cluster	P-value	GO number	GO category
C3	2.8e-11	GO:0044454	nuclear chromosome part
	4e-11	GO:0031324	negative regulation of cellular metabolic process
	4.3e-11	GO:0009892	negative regulation of metabolic process
	5.3e-11	GO:0005634	nucleus
	7.6e-11	GO:0051301	cell division
	8.5e-11	GO:0030894	replisome
	8.5e-11	GO:0043601	nuclear replisome
	1.1e-10	GO:0007064	mitotic sister chromatid cohesion
	1.3e-10	GO:0048523	negative regulation of cellular process
	1.7e-10	GO:0048519	negative regulation of biological process
	3e-10	GO:0006271	DNA strand elongation during DNA replication
	3e-10	GO:0022616	DNA strand elongation
	1.1e-09	GO:0006323	DNA packaging
	1.5e-09	GO:0000079	regulation of cyclic dependent protein kinase activity
	1.7e-09	GO:0005933	cellular bud
	2.1e-09	GO:0006342	chromatin silencing
	2.1e-09	GO:0031507	heterochromatin formation
	2.1e-09	GO:0045814	negative regulation of gene expression, epigenetic
	2.5e-09	GO:0043596	nuclear replication fork
	2.5e-09	GO:0005935	cellular bud neck
	3.8e-09	GO:0007062	sister chromatid cohesion
	4.2e-09	GO:0016458	gene silencing

Cluster	P-value	GO number	GO category
C3	4.2e-09	GO:0040029	regulation of gene expression, epigenetic
	5.8e-09	GO:0051325	interphase
	6.1e-09	GO:0051726	regulation of cell cycle
	1.7e-08	GO:0003677	DNA binding
	2.7e-08	GO:0031497	chromatin assembly
	3.1e-08	GO:0000819	sister chromatid segregation
	3.4e-08	GO:0051052	regulation of DNA metabolic process
	3.6e-08	GO:0000279	M phase
	4.2e-08	GO:0045892	attribute negative regulation of transcription, DNA-dependent
	4.3e-08	GO:0051329	interphase of mitotic cell cycle
	4.5e-08	GO:0006139	nucleobase, nucleoside, nucleotide and nucleic acid metabolic process
	4.6e-08	GO:0051253	negative regulation of RNA metabolic process
	7.9e-08	GO:0030427	site of polarized growth
	8.9e-08	GO:0006333	chromatin assembly or disassembly
	1e-07	GO:0016481	negative regulation of transcription
	1e-07	GO:0043228	non-membrane-bounded organelle
	1e-07	GO:0043232	intracellular non-membrane-bounded organelle
	1.1e-07	GO:0043549	regulation of kinase activity
	1.1e-07	GO:0045859	regulation of protein kinase activity
	1.1e-07	GO:0000793	condensed chromosome
	1.3e-07	GO:0006310	DNA recombination

Cluster	P-value	GO number	GO category
C3	1.3e-07	GO:0000731	DNA synthesis during DNA repair
	1.4e-07	GO:0051338	regulation of transferase activity
	1.5e-07	GO:0050794	regulation of cellular process
	3.7e-07	GO:0006950	cresponse to stress
	5.3e-07	GO:0006298	mismatch pair
	5.3e-07	GO:0045005	maintenance of fidelity during DNA-dependent DNA replication
	6.4e-07	GO:0000798	nuclear cohesin complex
	6.4e-07	GO:0008278	cohesin complex
	7.7e-07	GO:0050789	regulation of biological process
	1.2e-06	GO:0006338	chromatin remodeling
	1.2e-06	GO:0019887	protein kinase regulator activity
	1.3e-06	GO:0051053	negative regulation of DNA metabolic process
	1.4e-06	GO:0019219	regulation of nucleobase, nucleoside, nucleotide and nucleic acid metabolic process
	1e.4-06	GO:0007067	mitosis
	1.5e-06	GO:0000070	mitotic sister chromatid segregation
	1.6e-06	GO:0007059	chromosome segregation
	1.7e-06	GO:0000087	M phase of mitotic cell cycle
	1.8e-06	GO:0019207	kinase regulator activity
	2.1e-06	GO:0006289	nucleotide-excision repair
	2.2e-06	GO:0000794	condensed nuclear chromosome

Cluster	P-value	GO number	GO category
C3	3.5e-06	GO:0007534	gene conversion at mating-type locus
	3.5e-06	GO:0016538	cyclin-dependent protein kinase regulator activity
	4.2e-06	GO:0006284	base-excision repair
	4.3e-06	GO:0006270	DNA replication initiation
	6.4e-06	GO:0043566	structure-specific DNA binding
	7.4e-06	GO:0000075	cell cycle checkpoint
	7.4e-06	GO:0000082	G1/S transition of mitotic cell cycle
	1.2e-05	GO:0003690	double stranded DNA binding
	1.5e-05	GO:0006348	chromatin silencing at telomere
	1.5e-05	GO:0031509	telomeric heterochrmatin formation
	1.5e-05	GO:0006275	regulation of DNA replication
	1.5e-05	GO:0007533	mating type switching
	1.5e-05	GO:0000217	DNA secondary structure binding
	1.6e-05	GO:0050896	response to stimulus
	1.8e-05	GO:0030466	chromatin silencing at silent mating-type cassette
	2e-05	GO:0050790	regulation of catalytic activity
	2e-05	GO:0065007	biological regulation
	2.6e-05	GO:0031323	regulation of cellular metabolic process
	2.7e-05	GO:0000135	septin checkpoint
	2.7e-05	GO:0005658	alpha DNA polymerase: primase complex
	2.7e-05	GO:0031565	cytokinesis checkpoint
	3.2e-05	GO:0065009	regulation of molecular function

Cluster	P-value	GO number	GO category
C3	3.7e-05	GO:0016568	chromatin modification
	3.9e-05	GO:0051320	S phase
	4.4e-05	GO:0043283	biopolymer metabolic process
	4.6e-05	GO:0003006	reproductive developmental process
	4.6e-05	GO:0007530	sex determination
	4.6e-05	GO:0007531	mating type determination
	5.2e-05	GO:0019222	regulation of metabolic process
	6e-05	GO:0032502	developmental process
	6.8e-05	GO:0000400	four-way junction DNA binding
	6.8e-05	GO:0008622	epsilon DNA polymerase complex
C6	3e-12	GO:0005933	cellular bud
	8.4e-12	GO:0007049	cell cycle
	5.3e-11	GO:0005935	cellular bud neck
	5.5e-11	GO:0030427	site of polarized growth
	1.2e-08	GO:0051301	cell division
	5.7e-08	GO:0022402	cell cycle process
	1.7e-07	GO:0000278	mitotic cell cycle
	6.1e-07	GO:0022403	cell cycle phase
	1.5e-06	GO:0000142	cellular bud neck contractile ring
	1.5e-06	GO:0005826	contractile ring
	3.5e-06	GO:0007067	mitosis
	4e-06	GO:0000087	M phase of mitotic cell cycle
	7.1e-06	GO:0044430	cytoskeletal part
	1.2e-05	GO:0000910	cytokinesis
	1.8e-05	GO:0032153	cell division site
	1.8e-05	GO:0032155	cell division site part
	2.1e-05	GO:0005856	cytoskeleton

Cluster	P-value	GO number	GO category
C9	6 7e-09	GO 0042555	MCM complex
	6 5e-07	GO 0005656	pre-replicative complex
	6 8e-07	GO 0006267	pre-replicative complex assembly
	6e-06	GO 0051052	regulation of DNA metabolic process
	6e-06	GO 0003688	DNA replication origin binding
	7 3e-06	GO 0031261	DNA replication preinitiation complex
	2 3e-05	GO 0006270	DNA replication initiation
	6 9e-05	GO 0006260	DNA replication
C10	1 3e-12	GO 0005819	spindle
	1 8e-11	GO 0044430	cytoskeletal part
	1 9e-11	GO 0015630	microtubule cytoskeleton
	8 9e-11	GO 0005856	cytoskeleton
	3 6e-10	GO 0000278	mitotic cell cycle
	4e-10	GO 0007020	microtubule nucleation
	7 2e-10	GO 0007049	cell cycle
	7 7e-10	GO 0007017	microtubule-based process
	6 8e-05	GO 0008622	epsilon DNA polymerase complex
	2 1e-09	GO 0005200	structural constituent of cytoskeleton
	3 7e-09	GO 0000226	microtubule cytoskeleton organization and biogenesis
	4 2e-09	GO 0005874	microtubule
	5 3e-09	GO 0007059	chromosome segregation
	1 7e-08	GO 0000775	chromosome, pericentric region
	2 5e-08	GO.0043228	non-membrane-bounded organelle
	2 5e-08	GO 0043232	intracellular non-membrane-bounded organelle
	4 4e-08	GO 0007010	cytoskeleton organization and biogenesis

Cluster	P-value	GO number	GO category
C10	4.5e-08	GO:0022402	cell cycle process
	5e-08	GO:0051301	cell division
	1.3e-07	GO:0000776	kinetochore
	1.4e-07	GO:0005815	microtubule organizing center
	1.4e-07	GO:0005816	spindle pole body
	2.2e-07	GO:0000070	mitotic sister chromatid segregation
	2.2e-07	GO:0005694	chromosome
	2.4e-07	GO:0000922	spindle pole
	3.6e-07	GO:0005822	inner plaque of spindle pole body
	3.9e-07	GO:0000819	sister chromatid segregation
	6.3e-07	GO:0044427	chromosomal part
	9.7e-07	GO:0007067	mitosis
	1.1e-06	GO:0000087	M phase of mitotic cell cycle
	2.2e-06	GO:0000780	condensed nuclear chromosome, pericentric region
	2.6e-06	GO:0005876	spindle microtubule
	3e-06	GO:0000779	condensed chromosome, pericentric region
	3.1e-06	GO:0019237	centromeric DNA binding
	9.6e-06	GO:0000279	M phase
	1e-05	GO:0022403	cell cycle phase
	1.4e-05	GO:0000228	nuclear chromosome
	1.4e-05	GO:0003777	microtubule motor activity
	2.7e-05	GO:0000794	condensed nuclear chromosome
	3.7e-05	GO:0000778	condensed nuclear chromosome kinetochore

Cluster	P-value	GO number	GO category
C10	3.9e-05	GO:0044450	microtubule organizing center part
	4.5e-05	GO:0007052	mitotic spindle organization and biogenesis
	4.6e-05	GO:0044454	nuclear chromosome part
	4.8e-05	GO:0006996	organelle organization and biogenesis
	4.9e-05	GO:0000777	condensed chromosome kinetochore
	5e-05	GO:0000793	condensed chromosome
	5.8e-05	GO:0007051	spindle organization and biogenesis
	6.2e-05	GO:0000928	gamma-tubulin small complex, spindle pole body
	6.2e-05	GO:0000930	gamma-tubulin complex
	6.2e-05	GO:0008275	gamma-tubulin small complex
C13	4e-07	GO:0009086	methionine biosynthetic process
	7.2e-07	GO:0000097	sulfur amino acid biosynthetic process
	1e-06	GO:0006555	methionine metabolic process
	1.7e-06	GO:0044272	sulfur compound biosynthetic process
	1.8e-06	GO:0000096	sulfur amino acid metabolic process
	1.9e-06	GO:0009067	aspartate family amino acid biosynthetic process
	4.5e-06	GO:0009066	aspartate family amino acid metabolic process
	7.7e-06	GO:0006790	sulfur metabolic process
	2.2e-05	GO:0019344	cysteine biosynthetic process
	3e-05	GO:0006534	cysteine metabolic process

Cluster	P-value	GO number	GO category
C13	4.2e-05	GO:0008652	amino acid biosynthetic process
	5.2e-05	GO:0009309	amine biosynthetic process
	5.3e-05	GO:0044271	nitrogen compound biosynthetic process
	3e-05	GO:0006534	cysteine metabolic process
	4.2e-05	GO:0008652	amino acid biosynthetic process
	5.2e-05	GO:0009309	amine biosynthetic process
	5.3e-05	GO:0044271	nitrogen compound biosynthetic process

Table 6.11: p-value of cluster 3 obtained by DCCA over Dataset 2

Cluster	P-value	GO number	GO category
C3	9.0e-12	GO:0005819	spindle
	1.2e-10	GO:0015630	microtubule cytoskeleton
	4.0e-09	GO:0007017	microtubule-based process
	1.4e-08	GO:0005874	microtubule
	1.6e-08	GO:0000226	microtubule cytoskeleton organization and biogenesis
	2.1e-08	GO:0007010	cytoskeleton organization and biogenesis
	2.7e-08	GO:0007059	chromosome segregation
	4.9e-08	GO:0044430	cytoskeletal part
	8.9e-08	GO:0007020	microtubule nucleation
	1.7e-07	GO:0005856	cytoskeleton
	2.5e-07	GO:0005200	structural constituent of cytoskeleton
	5.1e-07	GO:0000278	mitotic cell cycle
	6.0e-07	GO:0005822	inner plaque of spindle pole body
	5.1e-06	GO:0005876	spindle microtubule
	1.0e-05	GO:0005875	microtubule associated complex

Cluster	P-value	GO number	GO category
C3	1.0e-05	GO:0005815	microtubule organizing center
	1.0e-05	GO:0005816	spindle pole body
	1.6e-05	GO:0000922	spindle pole
	2.4e-05	GO:0003777	microtubule motor activity
	2.4e-05	GO:0022402	cell cycle process
	2.9e-05	GO:0000775	chromosome, pericentric region
	4.0e-05	GO:0007067	mitosis
	4.5e-05	GO:0000087	M phase of mitotic cell cycle
	6.5e-05	GO:0044450	microtubule organizing center part
	7.2e-05	GO:0000778	condensed nuclear chromosome kinetochore
	8.7e-05	GO:0007052	mitotic spindle organization and biogenesis

by homogeneity, silhouette index and z-score measures of cluster validation. The regulation based cluster expansion maintains the pattern information in a simple regulation pattern. From our experimental results we conclude that the clustering solution obtained by DGC has higher degree of biological significance than the algorithms with which we compared it. The next chapter introduces an incremental clustering algorithm based on DGC that is capable of handling large incremental gene expression datasets.

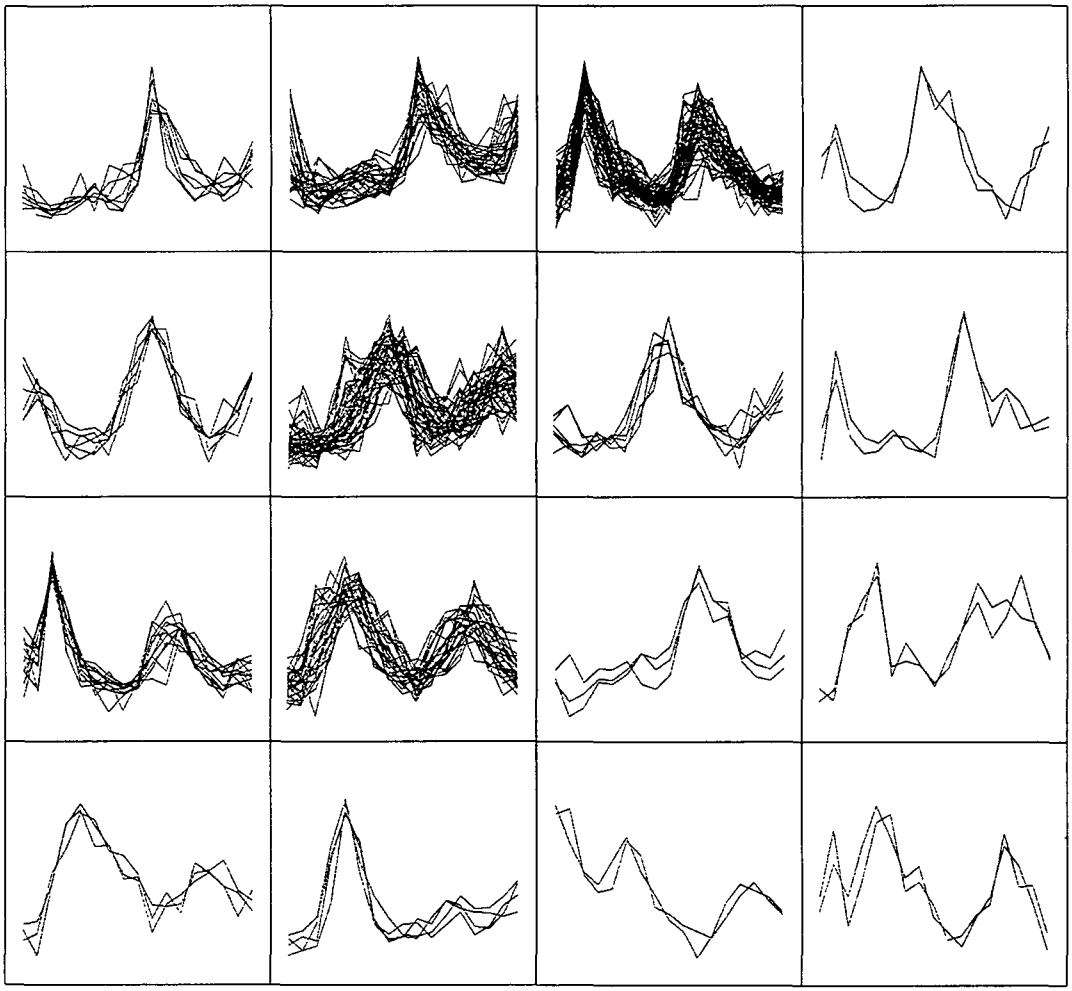


Figure 6.10: Some clusters generated using DGC on Dataset 2. A total of 17 clusters were detected.

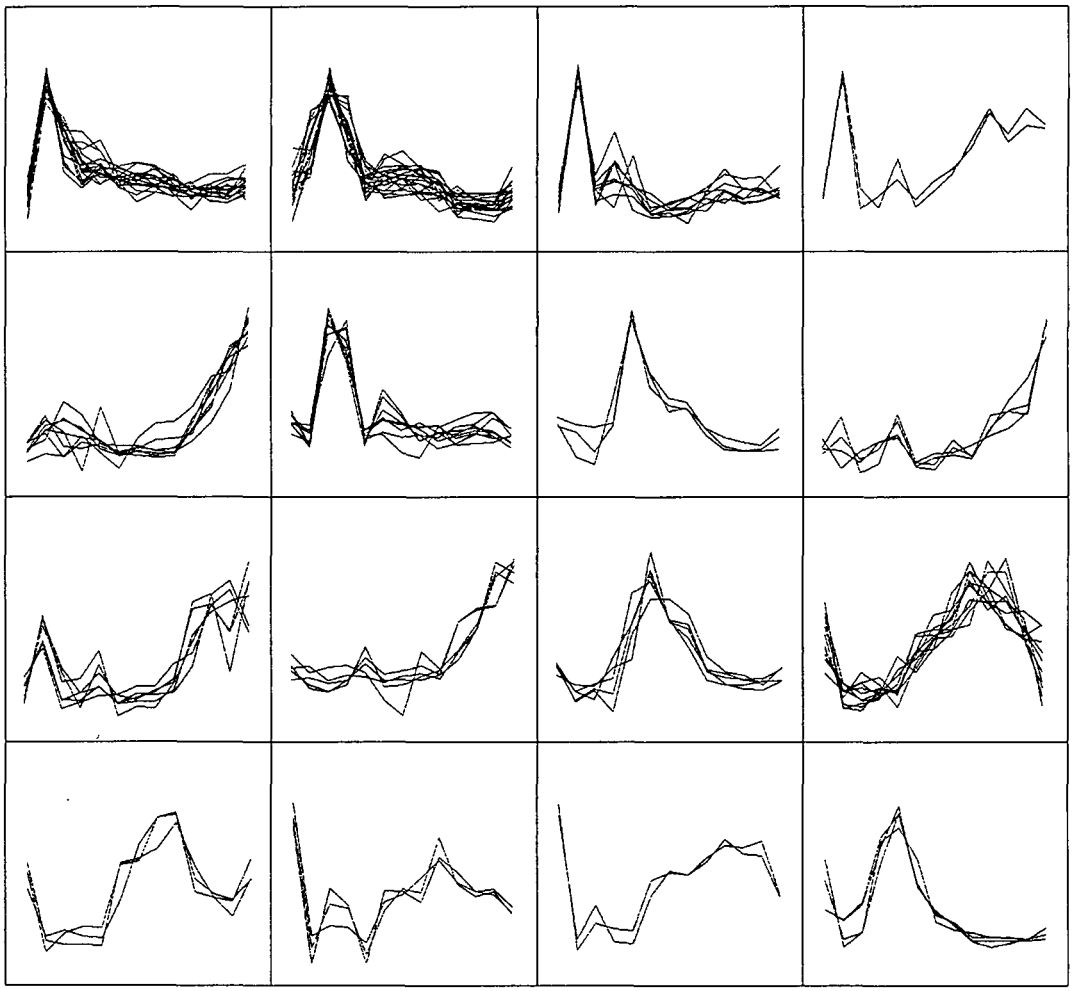


Figure 6.11: The clusters obtained by DGC on Dataset 6.

Chapter 7

incDGC: An Incremental Clustering Approach

In this chapter, we introduce an incremental density based gene clustering technique (incDGC) which is designed based on our existing density based clustering technique i.e., DGC which has been discussed in Chapter 6. The incDGC approach uses cluster profile information to cluster genes incrementally. We compare it's performance with that of a few other methods using real-life datasets and find that it detects biologically relevant clusters.

7.1 Introduction

The current information explosion, fuelled by the availability of the World Wide Web and the huge numbers of microarray experiments being continuously conducted, has led to ever-increasing volumes of gene expression data. Therefore, there is a need for incremental clustering so that updates can be clustered in an incremental manner. Though a lot of research has focussed on incremental clustering for other application domains, there has not been much study of incremental clustering in the context of gene expression data.

7.2 Related Work

We now present a review of some selected incremental clustering algorithms.

7.2.1 Incremental DBSCAN

In [EKS+98], the authors present an incremental clustering approach based on the DBSCAN [EKSX96] algorithm. The main idea behind the algorithm is that the insertion or deletion of an object affects the current clustering only in the neighborhood of this object. Density connections may surface or get removed depending on whether an object is added or deleted, respectively. Incremental DBSCAN yields the same result as DBSCAN executed over the whole updated database.

7.2.2 Incremental Clustering Algorithm (C^2ICM)

In [Can93], the authors propose an incremental clustering for dynamic processing. Documents are clustered by assigning them to clusters of the seed that covers them the most. Updations (additions and deletions) are handled by checking the newly inserted data, the *ragbag* cluster (documents not covered by any seed) and the members of the $falsified$ clusters and assigned to the most appropriate seed.

7.2.3 HIREL: An Incremental Clustering Algorithm for Relational Datasets

In [TS08], the authors present a one pass clustering algorithm for relational datasets HIREL is a multi-phase clustering algorithm where the dataset is first divided into a set of micro-clusters so that the variance of each cluster is equal to or less than some threshold A hierarchical dendrogram is built based on the micro-clusters to optimize the result The micro-clusters are indexed by a balanced search tree S to facilitate the assignment of new data to the appropriate cluster

7.2.4 Rough Set based Data Clustering

Rough set theory has been employed in the incremental approach for clustering interval datasets in [ANS06] It groups the given dataset into a set of overlapping clusters by employing a rough variant of the Leader algorithm [ANS06] The algorithm generates cluster abstractions in a single scan and is robust to outliers In [CCFM97], the authors present an incremental clustering model for information retrieval applications [CHNW96] and [FAAM97] also report efficient methods for modifying a set of association rules

7.2.5 Incremental Genetic k-means Algorithm (IGKA)

In [LLF+04b], an incremental genetic k-means algorithm (IGKA) has been presented IGKA calculates the objective value called Total Within-Cluster Variation (TWCV) and clusters centroids incrementally whenever the mutation probability is small IGKA converges to the global optimum In the Genetic k-means Algorithm (GKA) proposed in [KM99] a genetic algorithm is hybridized with the k-means algorithm and therefore GKA converges to the global optimum faster than other genetic algorithms In [LLF+04a], the authors present a faster version of GKA (FGKA) that efficiently evaluates the TWCV, avoids illegal string termination overhead and simplifies the mutation operator IGKA inherits all the advantages of FGKA and outperforms FGKA when the mutation probability is

small The cost of calculating the centroids in FGKA is more expensive when the mutation probability is small than when it is calculated incrementally in IGKA The Hybrid Genetic k-means Algorithm (HGKA) in [LLF+04b] combines the advantages of both IGKA and FGKA and obtains an even better performance However it is very difficult to obtain the threshold value which is dataset dependent

7.2.6 Best Incremental Ranked Subset (BIRS)

In [RRAR06], an incremental gene selection algorithm that reduces search space complexity using a wrapper-based method is presented This method works on the ranking directly In BIRS [RRAR06], genes are first ranked w r t an evaluation measure Then, the set of genes is updated by crossing the ranking from the beginning to the last ranked gene Classification accuracy with the first gene in the list is obtained and it is marked as selected The classification rate is again obtained and the second gene is selected depending on whether the classification accuracy is significantly better The process is repeated till the last gene on the ranked list is processed The algorithm returns the best subset formed and it does not contain irrelevant or redundant genes

7.3 Motivation

Due to the huge number of microarray experiments being conducted regularly, whenever new gene expression data becomes available it is highly desirable to perform updates (i e , incorporate the new results to existing clusters) with these newly arrived genes incrementally Therefore, we propose an incremental clustering method, *incDGC*, based on DGC

7.4 incDGC: Incremental DGC

The DGC algorithm as discussed in Chapter 6 can be used for clustering static gene expression data Due to the density based nature of DGC, the insertion of

a gene affects the current clustering only in the neighborhood of this gene. We find that the incremental algorithm yields the same result as DGC. A significant achievement would be if we could simply and incrementally update the clustering obtained by DGC (on the old database) to handle the new updates. We examine the parts of an existing set of clusters affected by an update and present the algorithm called incDGC for incremental updates of a set of clusters after insertions. The incremental clustering problem can be stated as follows: for an update of y genes in D_{upd}, incDGC maintains a collection of k clusters such that as each input gene is presented, either it is assigned to one of the current k clusters, or it starts a new cluster, or it merges two or more existing clusters into one, or that it is a noise gene.

The changes in the set of clusters in the gene dataset D_G are restricted to the neighborhood $N_\theta(g_i)$, of an inserted gene g_i. The previously identified core genes retain their core property, but non-core genes (border genes or noise genes) in $N_\theta(g_i)$ may become core. Thus new density connections may surface; that is, a new chain $g_1, \cdots, g_n$, $g_1 = r, g_n = s$ may arise with g_{j+1} directly density reachable from g_j for two genes r and s which were not density reachable before the insertion of g_i. Thus one of the genes g_j for $j < n$ must be contained in $N_\theta(g_i)$. Figure 7.1 shows an example dataset of genes illustrated in 2D and the gene g_i is to be inserted. Each of the points represents a gene. The genes a and b are density connected w.r.t. θ and $\sigma = 4$ without using any gene $\in N_\theta(g_i)$. On the other hand genes r and s are density connected via *the genes* $\in N_\theta(g_i)$ if the gene g_i is present. If g_i is not present r and s are not connected and they belong to different clusters. Thus the cluster memberships of r and s are dependent on the presence or absence of g_i. The insertion of a gene g_i may result in a change of cluster membership of genes in θ-neighborhood of g_i and all genes density reachable from one of these genes in $D_G \cup g_i$. While inserting g_i the following cases may occur.

1. Fusion

   If g_i is reachable from exactly one cluster C_i, g_i and possibly some noise genes are fused into cluster C_i.

2. Creation

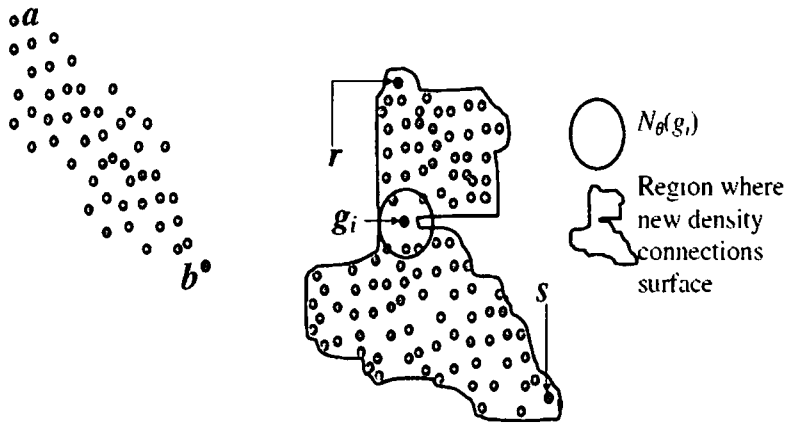


Figure 7.1: Example dataset of genes

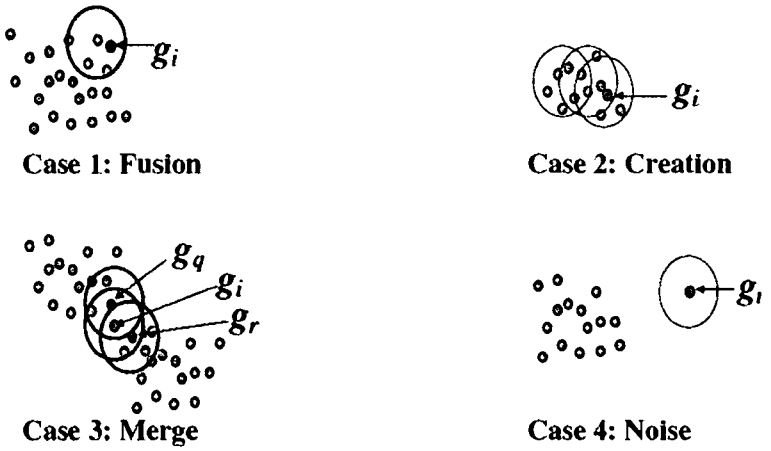


Figure 7.2: The different cases of insertion

g_i may become core w.r.t some other noise or unclassified gene(s) and may lead to the formation of a new cluster.

3. Merge
   Gene $g_q \in N_\theta(g_i)$ and g_q becomes core after insertion of g_i. Also, gene g_r is core and $g_r \in N_\theta(g_q)$. If both g_q and g_r belong to different clusters then all these clusters as well as g_i are merged to form one cluster.

4. Noise
   g_i is neither a core gene nor it is density reachable from any other core gene. Moreover, insertion of g_i does not produce any new core genes. Then g_i is noise gene and no density-connections are changed.

The four cases given above are depicted in Figure 7.2 for 2D illustration where $\sigma = 4$. The incDGC algorithm starts with a newly inserted gene g_i and finds its regulation pattern. Each of the clusters obtained by a call to the *DGC_cluster_creation*() (given in previous chapter) over the old database has a cluster MMRP. The updated database D_{upd} contains genes from both the old database D_G and incremental database D_I, *i.e.*, $D_{upd} = D_G \cup D_I$ and $\mid D_{upd} \mid = G + y$, where, G is the total number of genes and y is the total number of genes in the incremental dataset, D_I.
The steps of incDGC are given below.

1. Call DGC to create the clusters on D_G.

2. Represent each cluster by the cluster MMRPs.

3. For each of the unclassified genes, g_i, in the updated database, D_{upd}

   (a) Find the regulation pattern of g_i.

   (b) g_i is matched with each of the cluster MMRPs obtained from D_G.

   (c) If g_i matches with the MMRP of exactly one cluster C_i then incDGC proceeds with the gene g_i (and can be a viable case either for case 1, 2 or 4 above). In the θ-neighborhood of g_i, only those genes which belong to C_i or are unclassified become the seeds for cluster expansion.

(d) If g_i matches more than one cluster, then, the seeds for cluster expansion are genes in these clusters as well as unclassified genes belonging to θ-neighborhood of g_i (Case 3)

(e) If g_i matches none of the clusters, then, either case 2 or 4 may occur

4 Step 3 is repeated till all genes in D_{upd} are classified

For a gene expression database of G genes and y inserted genes, we derive the following theorems and lemmas

Theorem 1 incDGC has time complexity $O(G+y)$ in the worst case

Proof Assume m clusters have been detected by DGC in the database D_G of size G For an insertion of y genes, the cardinality of the updated database D_{upd} becomes $(G+y)$ For finding matching profile(s), incDGC compares the newly inserted genes with m profiles where $m << G$ This results in a complexity of $O(m)$ Once matching profile(s) is identified, neighborhood processing starts Let, x be the number of genes in a cluster Let $g_i \in \{D_{upd} - D_G\}$ be an inserted gene Assume g_i matches k clusters ($k \leq m$) Then the neighborhood query searches $((x \times k) + z)$ genes where $(x \times k) << G$ and z is the set of unclassified genes $\in D_{upd}$ This gives a complexity of $O((x \times k) + z)$ Once the neighborhood of g_i is identified, the four cases discussed above are checked Out of the four, the merging case is more costly taking at most $O(x \times k)$ time Therefore,

$$\textit{total time complexity} = O(m) + O((x \times k) + z) + O(x \times k)$$

$$= O((x \times k) + z)$$

In the worst case, $k = m$,

$$\textit{total time complexity} = O((x \times m) + z) \approx O(G + y)$$

Observation 1 Clustering obtained by incDGC is the same as the clustering obtained by DGC

Lemma 7.1. Let g_i be an inserted gene. Let there be two other genes $g_x \in C_1$ and $g_y \in C_2$ where C_1, C_2 are two clusters. If g_i becomes core and both g_x and g_y are reachable from g_i, then C_1 and C_2 are merged.

Proof. Suppose $g_x \in C_1$ and $g_y \in C_2$ and the inserted gene g_i is a core gene. Also, let g_x and g_y be reachable from g_i. Then g_x is density connected to g_y. Thus, as per Definition 6.9, g_x and g_y belong to the same cluster, i.e., clusters C_1 and C_2 should be merged. Hence the proof. □

Lemma 7.2. Let g_i be an inserted gene and genes $g_x \in C_1$ and $g_y \in C_2$ where C_1, C_2 are two clusters. If g_i is not core and g_i is reachable from both g_x and g_y, then g_i belongs to either C_1 or C_2.

Proof. Let, g_i be an inserted gene and assume g_i is not core. Also, let g_i be reachable from both the clusters $g_x \in C_1$ and $g_y \in C_2$. Then according to Definition 6.9, $g_i \in C_1$ and $g_i \in C_2$. However, as per *Lemma* 7.1, C_1, C_2 cannot be merged as g_i is not core. Therefore g_i can be included in any of C_1 or C_2. Hence the proof. □

A significant advantage of incDGC is that genes in the $N_\theta(g_i)$ having MMRP different from that of g_i are not considered for cluster expansion. This in turn reduces the computational cost of the algorithm significantly.

7.5 Performance Evaluation

We have implemented incDGC using Java in Windows environment and tested it on the real-life datasets given in Table 3.5.

7.5.1 Cluster Quality

To assess the quality of incDGC, we need an objective external criterion. We perform a statistical rating of the relative gene-expression activity in each cluster. In order to validate our clustering result, we use z-score [GR02] as the measure of agreement. A higher value of z indicates that genes are better clustered by function, indicating a more biologically relevant clustering result. Z-score [GR02]

is calculated by investigating the relation between a clustering result and the functional annotation of the genes in the cluster We have used Gibbons Cluster-Judge [GR02] tool to calculate z-score To test the performance of the clustering algorithm, we compare the clusters identified by our method with the results from k-means, UPGMA, SOM, DCCA and CLICK We average the z-score value over 50 repeated experiments The result of applying the z-score on the reduced form of Dataset 3 is shown in Table 7 1 In this table DGC is compared with the well known k-means and agglomerative hierarchical algorithm, UPGMA Table 7 1 clearly shows that our method outperforms both k-means and UPGMA w r t the cluster quality The z-score values obtained from clustering the full Dataset 3 is given in Table 7 2 We observe from the table that our method performs better than k-means and hierarchical clustering We note here that unlike k-means, our method does not require the number of clusters as an input parameter It detects the clusters present in the dataset automatically and identifies the rest as noise Also, UPGMA requires the parameter cutoff as input to the algorithm From both tables we see that DGC gives better cluster set at $\theta = 2$ for Dataset 3 The z-score for DGC and incDGC are shown in Table 7 1 and Table 7 2, respectively We see from the tables that incDGC discovers all the clusters as DGC Table 7 3 and Table 7 4 demonstrate that the clusters detected by DGC and incDGC for Dataset 2 and Dataset 7 are same respectively

We do not report the homogeneity and silhouette index values for incDGC since they are the same as for DGC We thus conclude that the clusters detected by incDGC are same as those detected by DGC

7.5.2 Execution Time Performance

We compare the execution times of DGC and incDGC by increasing the size of the dataset with updates of 500 genes for each iteration The execution time for both algorithms is illustrated in Figure 7 3 We see that incDGC is much more efficient than DGC We also see that with increase in the size of the updated database, the performance of DGC degrades unlike incDGC

Table 7.1: z-scores for incDGC, k-means at k=16 and 46 and UPGMA using average linkage at cutoff = 16 and 46 for the reduced form of Dataset 3

Method Applied	No. of Clusters	z-score	Total no. of genes
UPGMA	16	0.285	614
k-means	16	-0.366	614
UPGMA	46	1.69	614
DCCA	2	-0.995	614
k-means	46	0.193	614
DGC at $\theta = 0.7$	46	5.38	614
DGC at $\theta = 1$	44	6.55	614
DGC at $\theta = 1.5$	44	6.41	614
DGC at $\theta = 2$	44	7.07	614
DGC at $\theta = 2.7$	44	6.58	614
incDGC	**44**	**7.07**	**614**

Table 7.2: z-scores for incDGC, and UPGMA using average linkage at cutoff = 176 for the full Dataset 3

Method Applied	No. of Clusters	z-score	Total no. of genes
UPGMA	119	9.7	6089
k-means	119	8.6	6089
DCCA	4	12.1	6089
DGC at $\theta = 0.7$	176	9.12	6089
DGC at $\theta = 1$	128	7.02	6089
DGC at $\theta = 1.5$	120	11.2	6089
DGC at $\theta = 2$	**119**	**12**	**6089**
DGC at $\theta = 2.7$	121	11.2	6089
incDGC	**119**	**12**	**6089**

Table 7.3: z-scores for DCCA, k-means, SOM, DGC and incDGC for Dataset 2

Method Applied	No. of Clusters	z-score	Total no. of genes
DCCA	15	4.41	384
k-means	15	4.48	384
k-means	17	4.56	384
SOM	15	4.66	384
SOM	18	5.11	384
DGC at $\theta = 2$	**17**	**5.51**	**384**
incDGC	**17**	**5.51**	**384**

Table 7.4: z-scores for DCCA, k-means, CLICK, SOM, DGC and incDGC for the Dataset 7

Method Applied	No. of Clusters	z-score	Total no. of genes
DCCA	43	7.06	698
k-means	7	10.8	698
CLICK	5	11.3	698
SOM	4	11.9	698
SOM	6	11.7	698
SOM	9	10.7	698
DGC at $\theta = 4.5$	7	11.747	698
DGC at $\theta = 5$	**7**	**16.55**	**698**
incDGC	**7**	**16.55**	**698**

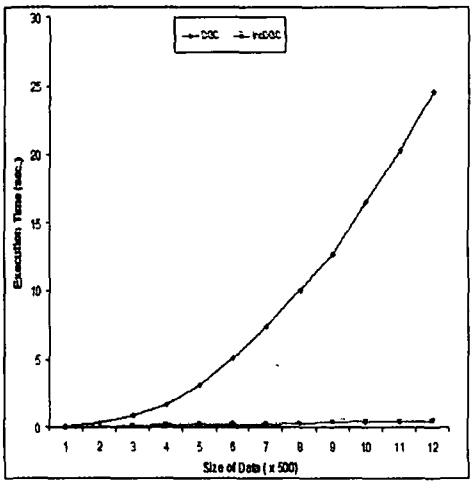


Figure 7.3: Execution Times of DGC and incDGC with increase in the size of dataset

7.6 Discussion

This chapter presents an incremental clustering algorithm (incDGC) based on DGC. incDGC does not require the number of clusters as input. The clusters obtained by incDGC are superior to those obtained by k-means, UPGMA, DCCA, SOM and CLICK based on z-score for three real datasets. The regulation based cluster expansion that we use also helps in maintaining the pattern information by using a simple pattern matching approach. The incDGC algorithm brings down the cost of performing DGC on the whole database after insertions are carried out. The number of neighborhood queries are scaled down very effectively than if DGC were allowed to run on the whole updated data. Moreover, the incDGC always gives the same result as DGC run on the whole updated database and is also much faster than DGC.

Chapter 8

Conclusions and Future work

8.1 Conclusions

In this thesis, we have developed four clustering algorithms for identification of coherent patterns in gene expression data. Clustering algorithms are dependent on the proximity measure used. Gene expression data are usually either up- or down- regulated across conditions. For gene expression data, capturing this regulation information is important. Moreover, the measure should also be robust in the presence of noise. We have developed an effective dissimilarity measure which addresses the above mentioned issues. Since gene expression data have a non-Gaussian distribution we have developed a graph based clustering algorithm which detects the clusters in non-Gaussian gene space. Our main objective is the detection of quality clusters. To this end we have developed a frequent itemset nearest neighbor based algorithm which gives finer clustering of the dataset. Finer clusters contain highly coherent genes. Density based clustering algorithms are known to detect quality clusters. We have developed a density based gene clustering technique that finds superior cluster sets than those obtained by k-means, UPGMA, DCCA, SOM, and CLICK based on cluster quality metrics such as z-score and coherence computation. With the increase in the size of gene databases and due to continuous updations of gene data, it is highly desirable if the newly inserted genes can be clustered incrementally. We have developed an incremental density based clustering approach which can handle such updations

in the gene dataset incrementally. All the clustering algorithms presented have been validated over several real-life datasets and found to be satisfactory.

8.2 Future Work

The work reported in this thesis can be expanded and improved in many different ways. Below, we list some ideas for future work.

- Clustering samples via genes as features is one of the key issues in problems such as class discovery, normal and tumor tissue classification and drug treatment evaluation [THC+99]. In this thesis, we have used gene-based clustering and it is desirable to experiment with sample based clustering in future.

- Since DGC finds clusters in subsets of conditions, one may be able to exploit a biclustering approach to make it more useful. Standard clustering algorithms group genes whose expression levels are similar across all conditions. However, a group of genes involved in the same biological process might only be co-expressed in a small subset of experimental conditions. In this sense, methods that can pull out subsets of genes associated with small subsets of experiments are likely to be useful. Much research has focused on biclustering approaches although they are still mainly focused on finding sets of related genes based only on expression data. Biological knowledge is still incorporated as a subsequent step to expression data analysis. In our future work, we plan to incorporate biological knowledge into gene expression data to detect the presence of sets of genes that share a similar expression pattern and common biological properties, such as function or regulatory mechanism.

- The use of external information is a helpful strategy in any data mining task. In association analysis of gene expression data, we may be able to use prior biological knowledge in many phases. The use of external information, such as gene annotations, is a useful strategy in any data mining task and may be incorporated into the data mining techniques such as clustering or

association mining In future work, a method for the integrative analysis of microarray data based on the association rules discovery technique might be used to automatically extract intrinsic associations among gene annotations and expression patterns These relationships will provide valuable information for the analysis and interpretation of gene expression datasets

- Furthermore, it will be useful to investigate how biological information can be integrated in the gene expression database during the clustering process A lot of research to analyze microarray data is frequently based on the application of clustering algorithms to establish sets of co-expressed genes However, these algorithms do not incorporate available information about genes and gene products, they just take into account experimental measurements Therefore, each set of co-clustered genes has to be further examined with the aim of discovering common biological connections among them In this way, biological information is incorporated as a subsequent process to the analysis of expression data However, simultaneously expressed genes may not always share the same function or regulatory mechanism Even when similar expression patterns are related to similar biological roles, discovering these biological connections among co-expressed genes is not a trivial task and requires substantial additional work A future direction of work could be to integrate the analysis of gene expression dataset with biological information about the different functions performed by the genes from sources such as Gene Ontology (GO) and finally use a semi-supervised clustering on the integrated dataset to identify the co-regulated genes

- It is also of utmost importance to validate if the final association rules generated are significant from a biological point of view Therefore significant biological validation methods must be used in order to validate the results obtained are biologically significant and will be of use to biologists

- As a future direction of our work, we plan to incorporate annotation information along with gene expression data while mining for coherent patterns

Bibliography

[ABB+00] M. Ashburner, C. Ball, J. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, and S.S. Dwight. Gene ontology· tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[ABN+99] U. Alon, N. Barkai, Notterman, D. A., K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide array. In *Proceedings of National Academy of Sciences*, volume 96(12), pages 6745–6750, USA, 1999.

[ADRB+09] R. Alves, S. Domingo, Rodriguez-Baena, S. Jesus, and Aguilar-Ruiz. Gene association analysis: a survey of frequent pattern mining from gene expression data. *Briefings in Bioinformatics*, 2009.

[AMS94] R. Agarwal, H. Mannila, and R. Shrikant. Fast algorithms for mining association rules in large databases. In *Proceedings of* 20^{th} *International Conference on Very Large Databases*, pages 487–499, Chile, 1994.

[ANS06] S. Asharaf, M. Narasimha, and S.K. Shevade. Rough set based incremental clustering of interval data. *Pattern Recognition Letters*, 27:515–519, 2006.

[ATS93] R. Agrawal, Imielinski T., and A. Swami. Mining association rules between set of items in large databases. In *Proceedings of ACM SIGMOD Conference on Management of Data*, pages 207–216, 1993.

[B+03] F. G. Berriz et al. Characterizing gene sets with funcassociate. *Bioinformatics*, 19:2502–2504, 2003.

[BD08] A. Bhattacharya and R. De. Divisive correlation clustering algorithm (dcca) for grouping of genes: detecting varying patterns in expression profiles. *Bioinformatics*, 24(11):1359–1366, 2008.

[BDCKY02] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: The order-preserving submatrix problem. In *Proc. Of the 6th Annual Int. Conf. on Computational Biology*, pages 49–57, New York, USA, 2002. ACM Press.

[BDSY99] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3-4):281–297, 1999.

[Bez81a] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

[Bez81b] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

[BG03] M.J. Beal and Z. Ghahramani. The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. In *Proceedings of the 7th Valencia International Meeting on Bayesian Statistics*, volume 63(4), pages 453–464, Spain, 2003.

[Bic01] D.R Bickel. Robust cluster analysis of dna microarray data: An application of nonparametric correlation dissimilarity. In *Proceedings of the Joint Statistical Meetings of the American Statistical Association (Biometrics Section)*, 2001.

[BMM07] S. Bandyopadhyay, A. Mukhopadhyay, and U. Maulik. An improved algorithm for clustering gene expression data. *Bioinformatics*, 23(21):2859–2865, 2007.

[BPC02] A. Bellaachia, D. Portnoy, and A. G. Chen, Y.and Elkahloun. E-cast: A data mining algorithm for gene expression data. In *Proceedings of the BIOKDD02: Workshop on Data Mining in Bioinformatics (with SIGKDD02 Conference)*, page 49, 2002.

[Can93] F. Can. Incremental clustering for dynamic information processing. *ACM Transactions on Information Systems*, 11(2):143–164, 1993.

[CCFM97] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 626–635, New York, NY, USA, 1997. ACM.

[CCW+98] R. J. Cho, M. Campbell, E. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. Wolfsberg, A. Gabrielian, D. Landsman, and D. Lockart. A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell*, 2(1):65–73, 1998.

[CDE+98] S. Chu, J. DeRisi, M. Eisen, J. Nulholand, D. Botstein, P. Brown, and I. Herskowitz. The transcriptional program of sporulation in budding yeast. *Science*, 282:699–705, 1998.

[CH03] C. Creighton and S. Hanash. Mining gene expression databases for association rules. *Bioinformatics*, 19:79–86, 2003.

[CHNW96] D.W. Cheung, J. Han, V.T. Ng, and Y. Wong. Maintenance of discovered association rules in large databases: An incremental technique. In *Proceedings of* 12^{th} *International Conference on Data Engineering*, pages 106–114, New Orleans, USA, 1996.

[CJM04] S. Chung, J. Jun, and D. McLeod. Mining gene expression datasets using density based clustering. Technical Report IMSC-04-002, USC/IMSC, University of Southern California, 2004.

[CSCR+06] P. Carmona-Saez, M. Chagoyen, A. Rodriguez, Oswaldo. Trelles, J. M. Carazo, and A. Pascual-Montano. Integrated analysis of

gene expression by association rules discovery *BMC Bioinformatics*, 7(54), 2006

[DC97] J Dopazo and JM Carazo Phylogenetic reconstruction using an unsupervised neural network that adopts the topology of a phylogenetic tree *Journal of Molecular Evolution*, 44 226–233, 1997

[DI97] J L DeRisi and P O Iyer, V R and Brown Exploring the metabolic and genetic control of gene expression on a genomic scale *Science*, 278 680–686, 1997

[DLR77] A Dempster, N Laird, and D Rubin Maximum likelihood from incomplete data via the EM algorithm *Journal of the Royal Statistical Society*, 39 (1) 1–38, 1977

[EKS+98] M Ester, H P Kriegel, J Sander, M Wimmer, and X Xu An incremental clustering for mining in a data warehousing environment In *Proceedings of the 24th VLDB Conference*, New York, USA, 1998

[EKSX96] M Ester, H P Kriegel, J Sander, and X Xu A density-based algorithm for discovering clusters in large spatial databases with noise In *Proceedings of International Conference on Knowledge Discovery in Databases and Data Mining (KDD-96)*, pages 226–231, Portland, Oregon, 1996

[ESBB98] M Eisen, P Spellman, P Brown, and D Botstein Cluster analysis and display of genome-wide expression patterns In *Proceedings of National Academy of Sciences*, volume 95, pages 14863–14868, 1998

[FAAM97] R Feldman, Y Aumann, A Amir, and H Mannila Efficient algorithms for discovering frequent sets in incremental databases In *Proceedings of ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 59–66, Tucson, AZ, 1997

[FPSV07a] P. Foggia, G. Percannella, C. Sansone, and M. Vento. Assessing the performance of a graph-based clustering algorithm. In *Proceedings of Sixth IAPR-TC-15 International Workshop on Graph-based representations in Pattern Recognition*, pages 215–227, Alicante, Spain, 2007.

[FPSV07b] P. Foggia, G. Percannella, C. Sansone, and M. Vento. A graph-based clustering method and its applications. In *Lecture Notes in Computer Science, Advances in Brain, Vision, and Artificial Intelligence*, volume 4729/2007, pages 277–287, Springer, Berlin, 2007.

[Gae02] M. Gaertler. Clustering with spectral methods. Master's thesis, Universität Konstanz, Germany, 2002.

[GR02] F. Gibbons and F. Roth. Judging the quality of gene expression based clustering methods using gene annotation. *Genome Research*, 12:1574–1581, 2002.

[GWBO+07] A. Gyenesei, U. Wagner, S. Barkow-Oesterreicher, E. Stolte, and R Schlapbach Mining co-regulated gene profiles for the detection of functional associations in gene expression data. *Bioinformatics*, 23(15):1927–1935, 2007.

[HCML05] D. Huang, W Chow, E. Ma, and J Li Efficient selection of discriminant genes from microarray gene expression data for cancer diagnosis. *IEEE Transactions on Circuits and Systems*, 52(9):1909–1918, 2005.

[HDRT04] F. V. D. Heijden, R. Duin, D. Ridder, and D. M. J. Tax. *Classification, Parameter Estimation and State Estimation: An Engineering Approach Using MATLAB.* John Wiley and Sons, 2004.

[HK04] J Han and M Kamber. *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers, San Fransisco, USA, 2004.

[HKY99] L J Heyer, S Kruglyak, and S Yooseph Exploring expression data identication and analysis of co-expressed genes *Genome Research*, 9(11) 11061115, 1999

[HPY00] E H Han, J Pei, and J Yin Mining frequent patterns without candidate generation In *Proceedings of the 2000 ACM-SIGMOD International Conference on Management of Data*, pages 1–12, Texas, USA, 2000

[HS78] E Horowitz and S Sahni *Fundamentals of Computer Algorithms* Computer Science Press, 1978

[HSL+99] E Hartuv, A Schmitt, J Lange, S Meier-Ewert, H Lehrach, and R Shamir An algorithm for clustering cDNAs for gene expression analysis using short oligonucleotide fingerprints In *Proceedings of 3rd International Symposium on Computational Molecular Biology (RECOMB 99)*, pages 188–197 ACM Press, 1999

[HVD01] J Herrero, A Valencia, and J Dopazo A hierarchical unsupervised growing neural network for clustering gene expression patterns *Bioinformatics*, 17 126136, 2001

[IER+99] V R Iyer, M B Eisen, D T Ross, G Schuler, T Moore, J Lee, J M Trent, L M Staudt, J J Hudson, M S Boguski, D Lashkari, D Shalon, D Botstein, and P O Brown The transcriptional program in the response of the human fibroblasts to serum *Science*, 283 83–87, 1999

[JP73] R A Jarvis and E A Patrick Clustering using a similarity measure based on shared nearest neighbors *IEEE Transactions on Computers*, 11, 1973

[JPZ03a] D Jiang, J Pei, and A Zhang DHC a density-based hierarchical clustering method for time series gene expression data In *Proceedings of BIBE2003 3rd IEEE International Symposium on Bioinformatics and Bioengineering*, page 393, Bethesda, Maryland, USA, 2003

[JPZ03b] D Jiang, J Pei, and A Zhang Interactive exploration of coherent patterns in time-series gene expression data In *Proceedings of Ninth ACM SIGKDD International Conference on KnowledgeDiscovery and Data Mining (SIGKDD03)*, pages 24–27, Washington, DC, USA, 2003

[JPZ03c] D Jiang, J Pei, and A Zhang Towards interactive exploration of gene expression patterns *SIGKDD Explorations*, 5(2) 79–90, 2003

[JPZ04] D Jiang, J Pei, and A Zhang GPX Interactive mining of gene expression data In *VLDB*, pages 1249–1252, 2004

[JTZ03] D Jiang, C Tang, and A Zhang Cluster analysis for gene expression data A survey available at www cse buffalo edu/ DBGROUP/bioinformatics/ papers/survey pdf , 2003

[Jus06] P Juszczak *Learning to recognise, a study on one-class classification and active learning* PhD thesis, Delft University of Technology, Netherlands, 2006 Isbn 978-90-9020684-4

[JW98] R A Johnson and D W Wichern *Applied Multivariate Statistical Analysis* Prentice Hall, NJ, USA, 1998

[KM99] K Krishna and M Murty Genetic k-means algorithm *IEEE Transactions on Systems, Man and Cybernetics - Part B Cybernetics*, 29 433–439, 1999

[Koh95] T Kohonen *Self-organizing maps* Springer-Verlag, Heidelberg, Germany, 1995

[Kra75] E F Krause *Taxicab geometry An adventure in non-Euclidean geometry* Dover, New York, 1975

[KVV00] R Kannan, S Vampala, and A Vetta On clustering Good, bad and spectral *In Foundations of Computer Science*, pages 367–378, 2000

[KW02] S A Krawetz and D D Womble *Introduction to Bioinformatics A Theoretical and Practical Approach* Humana Press, Totowa, NJ, USA, 2002

[LAA05] X Liu, Krishnan A , and Mondry A An entropy-based gene selection method for cancer classification using microarray data *BMC Bioinformatics*, 6, 2005

[LL91] G S Lennon and H Lehrach Hybridization and analysis of arrayed cDNA libraries *Trends in genetics*, 7(10) 314–317, 1991

[LLF+04a] Y Lu, S Lu, F Fotouhi, Y Deng, and S J Brown FGKA A fast genetic k-means algorithm In *Proc ACM Symposium on Applied Computing*, 2004

[LLF+04b] Y Lu, S Lu, F Fotouhi, Y Deng, and S J Brown Incremental genetic k-means algorithm and its application in gene expression data analysis *BMC Bioinformatics*, 5(172), 2004

[LWN+09] G Li, Z Wang, Q Ni, X Wang, B Qiang, and H Qing-juan Application of a new similarity measure in clustering gene expression data Downloaded from http //ieeexplore ieee org/stamp/stamp jsp?arnumber=05162382, 2009

[McQ67] J B McQueen Some methods for classification and analysis of multivariate observations In L M Le Cam and J Neyman, editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297 University of California Press, 1967

[MMB09] U Maulik, A Mukhopadhyay, and S Bandyopadhyay Combining pareto-optimal clusters using supervised learning for identifying co-expressed genes *BMC Bioinformatics*, 10(27), 2009

[Per01] W Perrizo Peano count tree technology Technical report, NDSU-CSOR-TR-01-1, North Dakota State University, Fargo, North Dakota, United States, 2001

[Rou87] P Rousseeuw Silhouettes a graphical aid to the interpretation and validation of cluster analysis *Journal of computational and applied mathematics*, 20 153–65, 1987

[RRAR06] R Ruiz, J C Riquelme, and J S Aguilar-Ruiz Incremental wrapper-based gene selection from microarray data for cancer classification *Pattern Recognition*, 39 23832392, 2006

[RWDF00] P Reymonda, H Webera, M Damonda, and EE Farmera Differential gene expression in response to mechanical wounding and insect feeding in arabidopsis *Plant Cell*, 12 707–720, 2000

[SAP06] R Syamala, T Abidin, and W Perrizo Clustering microarray data based on density and shared nearest neighbor measure In *Computers and Their Applications*, pages 360–365, 2006

[SAS99] SAS Institute Inc Cary, NC *SAS/STAT User Guide, Version 8 0*, 1999

[SMKS03] R Sharan, A Maron-Katz, and R Shamir Click and expander a system for clustering and visualizing gene expression data *Bioinformatics*, 19(14) 1787–1799, 2003

[SS00] R Sharan and R Shamir Click A clustering algorithm with applications to gene expression analysis In *Proceedings of 8th International Conference on Intelligent Systems for Molecular Biology*, pages 307–316 AAAI Press, 2000

[SSI+98] P T Spellman, M Q Sherlock, G andZhang, V R Iyer, K Anders, M B Eisen, P O Brown, D Botstein, and B Futcher Comprehensive identification of cell cycleregulated genes of the yeastsaccharomycescerevisiaebymicroarrayhybridization *MolecularBiology of theCell*, 9(12) 32733297, 1998

[Ste06] D Stekel *Microarray Bioinformatics* Cambridge University Press, Cambridge, UK , 2006

[SZCS03] G. Shu, B. Zeng, Y.P. Chen, and O.H. Smith. Performance assessment of kernel density clustering for gene expression prole data. *Comparative and Functional Genomics*, 4:287299, 2003.

[TA02] A. Tuzhilin and G. Adomavicius. Handling very large numbers of association rules in the analysis of microarray data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Data Mining and Knowledge Discovery*, pages 396–404, 2002.

[TH09] J.H. Travis and Y. Huang. Clustering of gene expression data based on shape similarity. *EURASIP Journal on Bioinformatics and Systems Biology,*, 2009(195712), 2009.

[THC+99] S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church. Systematic determination of genetic network architecture. *Nature Genet*, 22:281285, 1999.

[THHK02] S. Tomida, T. Hanai, H. Honda, and T. Kobayashi. Analysis of expression profile using fuzzy adaptive resonance theory. *Bioinformatics*, 18(8):1073–83, 2002.

[TS08] L. Tao and S.A. Sarabjot. HIREL: An incremental clustering algorithm for relational datasets. In *Proceedings of Eighth IEEE International Conference on Data Mining*, pages 887–892, 2008.

[TSK09] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining.* Pearson Education, New York, 2009.

[TSM+99] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E.S. Lander, and T.R. Golub. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation In *Proceedings of National Academy of Sciences*, volume 96(6), pages 2907–2912, USA, 1999.

[vD00] S.M. van Dongen. *Graph Clustering by Flow Simulation.* PhD thesis, University of Utrecht, Netherlands, 2000.

[WFM+98] X. Wen, S. Fuhrman, G.S. Michaels, D.B. Carr, S. Smith, J.L. Barker, and R. Somogyi. Large-scale temporal gene expression mapping of central nervous system development. *PNAS*, 95(1):334–339, 1998.

List of Publications

1. Rosy Das, D.K. Bhattacharyya and J.K. Kalita, *A Pattern Matching Approach for Clustering Gene Expression Data*, accepted for publication in International Journal of Data Mining, Modeling and Management, Vol 3, No. 2, 2011

2. Rosy Das, D.K. Bhattacharyya and J.K. Kalita, *Clustering Gene Expression Data using an Effective Dissimilarity Measure*, International Journal of Computational BioScience (Special Issue), Vol. 1 (1), pp. 55-68, 2010.

3. Rosy Das, D.K. Bhattacharyya and J.K. Kalita, *An Incremental Clustering of Gene Expression Data*, in the Proceedings of NABIC09, pp. 742 - 747, Coimbatore, India, 2009, doi: 10.1109/NABIC.2009.5393848.

4. Rosy Das, D.K. Bhattacharyya and J.K. Kalita, *Clustering Gene Expression Data using a Regulation Based Density Clustering*, International Journal of Recent Trends in Engineering, Vol. 2, No. 1-6, pp. 76-78 2009.

5. Rosy Das, J.K. Kalita and D.K. Bhattacharyya, *A New Approach for Clustering Gene Expression Time Series Data*, International Journal of Bioinformatics Research and Applications, Vol. 5, No. 3, pp. 310-328, 2009.

6. Rosy Das, Sauravjyoti Sarmah and D. K. Bhattacharyya, *A Pattern Matching Approach for Identifying Coherent Patterns over Gene Expression Data*, Journal of ASS, Vol.50 (1 & 2) 2009.

7. Rosy Das, D K Bhattacharyya and J.K. Kalita, *A Frequent Itemset-Nearest Neighbor Based Approach for Clustering Gene Expression Data*, in the Pro-

ceedings of Fifth Biotechnology and Bioinformatics Symposium (BIOT'08), pp. 73-78, Texas, 2008.

8. Rosy Das, D K Bhattacharyya and J.K. Kalita, *An Effective Dissimilarity Measure for Clustering Gene Expression Time Series Data*, in the Proceedings of Fourth Biotechnology and Bioinformatics Symposium (BIOT'07), pp. 36-41, Colorado, USA, October, 2007