2003

Adaptive Double Self-Organizing Map for Clustering Gene Expression Data

Dali Wang

ADAPTIVE DOUBLE SELF-ORGANIZING MAP FOR CLUSTERING GENE EXPRESSION DATA

By

Dali Wang

B.S. Shanghai Jiao Tong University, China, 1998

A THESIS

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

(in Electrical Engineering)

The Graduate School

The University of Maine

August, 2003

Advisory Committee:

Habtom Ressom, Assistant Professor of Electrical & Computer Engineering, Advisor

Mohamad T. Musavi, Professor of Electrical & Computer Engineering

Cristian Domnisoru, Research Associate Professor of Electrical & Computer Engineering

LIBRARY RIGHTS STATEMENT

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at The University of Maine, I agree that the Library shall make it freely available for inspection. I further agree that the Librarian may grant permission for "fair use" copying of this thesis for scholarly purposes. It is understood that any copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Signature: 

Date: 05/30/03

ADAPTIVE DOUBLE SELF-ORGANIZING MAP FOR CLUSTERING GENE EXPRESSION DATA

By Dali Wang

Thesis Advisor: Dr. Habtom Ressom

An Abstract of the Thesis Presented
in Partial Fulfillment of the Requirements for the
Degree of Master of Science
(in Electrical Engineering)
August, 2003

This thesis presents a novel clustering technique known as adaptive double self-organizing map (ADSOM) that addresses the issue of identifying the "correct" number of clusters. ADSOM has a flexible topology and performs clustering and cluster visualization simultaneously, thereby requiring no a *priori* knowledge about the number of clusters. ADSOM combines features of the popular self-organizing map with two-dimensional position vectors, which serve as a visualization tool to decide the number of clusters. It updates its free parameters during training and it allows convergence of its position vectors to a fairly consistent number of clusters provided that its initial number of nodes is greater than the expected number of clusters. A novel index is introduced based on hierarchical clustering of the final locations of position vectors. The index allows automated detection of the number of clusters, thereby reducing human error that could be incurred from counting clusters visually. The reliance of ADSOM in identifying the number of clusters is proven by applying it to publicly available gene expression data from multiple biological systems such as yeast, human, mouse, and bacteria.

ACKNOWLEDGEMENTS

I am grateful to my advisor Dr. Habtom Ressom, as well as Dr. Mohamad Musavi, for providing me with the opportunity to pursue my Master's degree at University of Maine. I would like to thank them for all their time, encouragement and guidance during my over two-year graduate study.

I would also like to thank Dr. Cristian Domnisoru for his kindly assistance in my graduate research work as well as in many other things at the University of Maine. I wish to thank graduate coordinator Dr. Donald Hummels, Dr. John Vetelino and Dr. David Kotecki for all of their time and assistance with the various courses I have taken.

Thanks also go to Ms. Padma Natarajan for her care and encouragement. I want to thank all the other faculty members in the department and other lab members in the Intelligent Systems Laboratory who have given me help during my graduate study at University of Maine. I would like to thank Dr. Su of the National Central University in Taiwan for his initial idea about double self-organizing map.

Finally, I would like to thank my family for their support and care, without which this thesis couldn't have been done. I also extend my thanks to all my friends for their encouragement and assistance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

Recent technological advances allow us to measure expression levels for thousands of genes simultaneously [1]. Efficient techniques are urgently needed to effectively analyze the generated gene expression data. The availability of reliable and accurate analysis tools will help the research community to identify the genetic make-up of the diseases and lead to suitable medical interventions.

Key and early processing steps in the analysis of gene expression data include clustering groups of genes that manifest similar expression patterns. Genes with similar expression profiles might be transcriptionally regulated through a same transduction pathway. Thus, the rationale for clustering gene expression data is to identify a new transduction pathway or novel genes, which may be co-regulated through the same known pathway.

A wide range of techniques have been applied for clustering gene expression data. Examples include hierarchical clustering [2], adaptive resonance theory (ART) [3], self-organizing map (SOM) [4], k-means [5], graph-theoretic approaches [6], [7], fuzzy ART [8], fuzzy c-means [9], fuzzy Kohonen [10], and growing cell structures network [10]. However, most of the above mentioned clustering algorithms are heuristically motivated, and the issues of determining the "correct" number of clusters and choosing a "good" clustering algorithm [2] are not yet rigorously solved. Clustering gene expression data

using hierarchical clustering and SOM has been very popular among the bioinformatics research community.

Hierarchical clustering organizes the expression profiles in a hierarchical tree structure, which allows detecting higher order relationships between clusters of profiles. Although hierarchical clustering has been proven valuable for describing gene expression [2], [11], it has several shortcomings. The hierarchical trees do not reflect the multiple ways in which expression patterns of genes can be similar. The deterministic nature of hierarchical clustering can cause the data points to be clustered on the basis of local decisions, with no opportunity to reevaluate. As the amount of data increases, this problem can be exacerbated. Mangiameli et al. [12] compared SOM with hierarchical clustering method and found that SOM is superior in both robustness and accuracy.

The design of SOM starts with defining a geometric configuration for the partitions in a one- or two-dimensional grid. Then, random weight vectors are assigned to each partition. During training, a gene expression profile is picked randomly. The weight vector closest to the expression profile is identified. The identified weight vector and its neighbors are adjusted to look similar to the expression profile. This process is repeated until the weight vectors converge to a prespecified degree. During operation, SOM maps gene expression profiles to the relevant partitions based on the weight vectors to which they are most similar. However, in circumstances where the expected number of partitions (clusters) available in gene expression data is unknown, the validation of SOM's clustering result becomes a critical issue. One may heuristically validate the clustering results to identify a reasonable number of clusters.

Many validation techniques have been implemented to evaluate clustering results. Yeung et al. [13] introduced figure of merit (FOM); Tibshirani et al. [14] applied "gap statistic"; Jain and Dubes [15] referred to using "Hubert and Jaccard index"; Hubert and Arabie [16] addressed the use of "adjusted rand index"; Lubovac et al. [17] proposed to use "entropy measure"; Musavi et al. [3] used the "change in internode distance per cluster". All of these algorithms have been proven valuable by some experiments. However, the use of heuristic evaluation technique makes the clustering process extremely time-consuming and complicated given the large volume and high-dimension of the gene expression data.

A number of methods have been proposed in the literature to accomplish data partitioning and cluster validation/visualization either simultaneously or independently. Fraley and Raftery [18], [19] developed model based clustering that provides functionality for displaying/visualizing cluster results. Ramoni et al. [20] introduced Bayesian method of model based clustering of gene expression dynamics. Kaski et al. [21], [22] introduced methods for detecting, visualizing and interpreting clusters generated by SOM. They improved the SOM-based method of U-matrix [23] for visualization of cluster information. Nikkilä et al. [24] and Kaski [25] have applied this improved method for the analysis and visualization of gene expression data. Herrero et al. [26], [27] have proposed a combination of self-organizing map and hierarchical method for clustering gene expression data.

Su and Chang [28] developed a new technique known as double self-organizing map (DSOM). In DSOM, each center in the network has an N-dimensional weight vector and a two-dimensional *position vector*. The position vectors are projection of the weight

vectors into a two-dimensional space and serve as a visualization tool for deciding how many clusters are needed, thus combining clustering and cluster visualization in one computational procedure. In other words, with the help of position vectors, DSOM adjusts its network structure during the learning phase so that neurons that respond to similar stimuli will not only have similar weight vectors but also move spatially nearer to each other.

Although DSOM addresses the problem of deciding number of clusters needed, the selection of its free parameters is vital for a proper projection of the position vectors in a two dimensional space. Some combinations of these parameters make all the position vectors converge too quickly into a small dense area. Some other combinations lead the updating process to "get stuck" after several epochs and result in wrong number of clusters. Thus, the regulation of these parameters remains a challenge.

In this thesis, an adaptive double self-organizing map (ADSOM) is proposed. ADSOM updates the free parameters involved in DSOM during the training process. This is achieved by carefully analyzing the mathematical relationships between the parameters and the updating processes. Unlike DSOM, ADSOM gives fairly consistent number of clusters provided that the initial number of nodes is greater than the expected number of clusters. In addition, a novel hierarchical tree-based index is introduced to help identify the number of clusters from the results obtained using ADSOM.

To demonstrate its effectiveness, ADSOM is applied to cluster gene expression data from multiple biological systems such as yeast, human, and mouse. The results show that ADSOM is a reliable technique for clustering gene expression data. ADSOM

addresses the issue of identifying unknown number of clusters while performing data partitioning simultaneously.

1.2 Purpose of the Research

The main objective of this thesis is to develop a tool that can effectively identify the number of clusters and accurately partition gene expression data.

A novel and reliable neural network-based clustering technique, adaptive double self-organizing map (ADSOM), is introduced in this thesis to accomplish both clustering and cluster validating simultaneously. The proposed technique is applied to identify the number of clusters and partition gene expression data from multiple biological systems, such as yeast, human, mouse, and bacteria.

Other clustering and validating techniques, such as fuzzy c-means, figure of merit, model-based clustering etc, are applied to analyze gene expression data as well. The comparisons are made in this thesis as well.

1.3 Thesis Organization

This thesis is comprised of eight chapters. Chapter 2 provides an overview of microarray technology, clustering gene expression data and its challenges. Chapter 3 reviews some important clustering methods, such as self-organizing map, adaptive resonance theory, fuzzy c-means, and model-based clustering. Chapter 4 introduces some validating techniques such as figure of merit, change in internode distance per cluster and Xie-Beni index. Chapter 5 introduces double self-organizing map and adaptive double self-organizing map (ADSOM). Chapter 6 described a novel hierarchical tree-based index that is implemented to validate number of clusters obtained

by ADSOM. Chapter 7 shows the experimental gene expression data and results. Chapter 8 offers concluding remarks and provides suggestions for future work.

CHAPTER 2

MICROARRAY TECHNOLOGY AND CLUSTERING

This chapter provides background information on microarray technology, gene expression data clustering and its challenges.

2.1 DNA Microarray Technology

DNA microarrays [1] attempt to analyze the expression of different genes in parallel on any scale up to the entire genome of an organism.

The construction of microarrays begins with the production of complimentary DNA (cDNA) segments that represent each gene. Each segment is the complement to the actual DNA sequence of a gene and differs from the corresponding mRNA sequence only in that thymine in cDNA replaces uracil in mRNA. Each spot on the microarray is created by inserting copies of a gene's cDNA sequence on a glass slide or other substrate by a high speed robotic process that physically binds the sequence to a small spot on the slide. A spot is created for each gene sequence to be used in the microarray. The substrate and the spots of DNA sequences are collectively known as the microarray. Each spot is referred to as a probe.

To measure gene expression for a cell population, mRNA is extracted from the cells and is reverse-transcribed into complimentary DNA (cDNA). This cDNA sequence is identical to the DNA sequence for the gene found in the nucleus and is thus complimentary to the cDNA probes on the microarray chip. The concentration of each sequence is multiplied proportionally through chemical reactions Chemical dyes (often

green and red in microarray experiments) are bound to the sequences to allow for subsequent analysis of concentration. A solution of this dyed cDNA is created and exposed to the microarray. On the microarray, the cDNA sequences bind, or hybridize, to the probes that contain their complimentary sequence. After a prescribed amount of time, the remaining cDNA solution is washed off the chip. What remains are the probes and the cDNA sequences that hybridized with them. The microarray is scanned with a laser set at the wavelength of the dye's color. The florescent intensity of each spot indicates approximately how many copies of the gene are bound to the spot, and thus, a relative perspective of the expression of that gene in the cell. The appearance of a scanned microarray can be found in Figure 2.1.

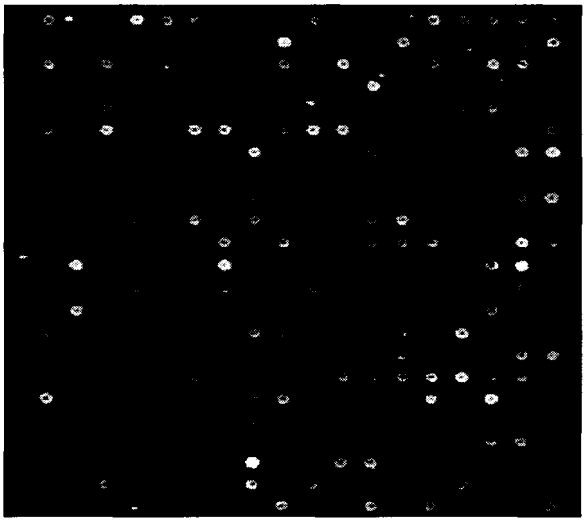


Figure 2.1: A scanned DNA microarray

Unfortunately, the florescence alone tells us very little when the gene expression from only one population is used; we cannot directly correlate the florescence of a probe to the copies of a gene on that probe. To alleviate the problem, we can add a second

population whose cDNA sequences were treated with a different dye. This second population can be used as a control population; in the case of time series data, the second (control) population is often the cell population at a fixed point of time while the first population is the same cell population at a later time. The two dyes should have colors of significantly different wavelengths to avoid "crosstalk", i.e., a situation where one dye affects the measured florescence of the other. The relative difference in florescence of the two dyes on a particular spot should tell us how much a gene's expression differs between the two populations. Expression levels can be reported as some form of difference between the two florescences, such as a ratio. Gene expression probes can be assembled from a series of these differential values at different points in time. The experiments of Spellman et al. [11] display gene expression time series as a listing of the ratios between the experimental and control expression levels for each time point.

Figure 2.2 presents an example of gene expression matrix. It shows that a gene expression data matrix is constructed with rows representing genes and with columns representing experimental conditions/samples. Usually, there are two common ways of analyzing the expression matrix: one is to compare rows (expression profiles of genes) in the gene expression data matrix; the other is to compare columns (expression profiles of samples) in the matrix.

The technology is young and still has some problems. First, the florescence signal is unlikely to exactly match the level of expression of each gene. The probe solution used is far from a free solution; the distribution of a certain cDNA sequence through the solution is not even. This problem may be partially alleviated by devoting several spots on the microarray to each gene and averaging the results, but it cannot guarantee the

elimination of the problem. cDNA probes with similar, but not identical, sequences to a particular spot on the microarray may still hybridize to the spot with mixed results, exaggerating the expression of one gene, possibly at the expense of another. Kerr et al. [29] named array effects, dye effects, populations and genes as source of variation that have a significant effect on the relative expression of a gene from these microarray experiments. This variation can be viewed in terms of "noise" in our signal of gene expression for each gene.

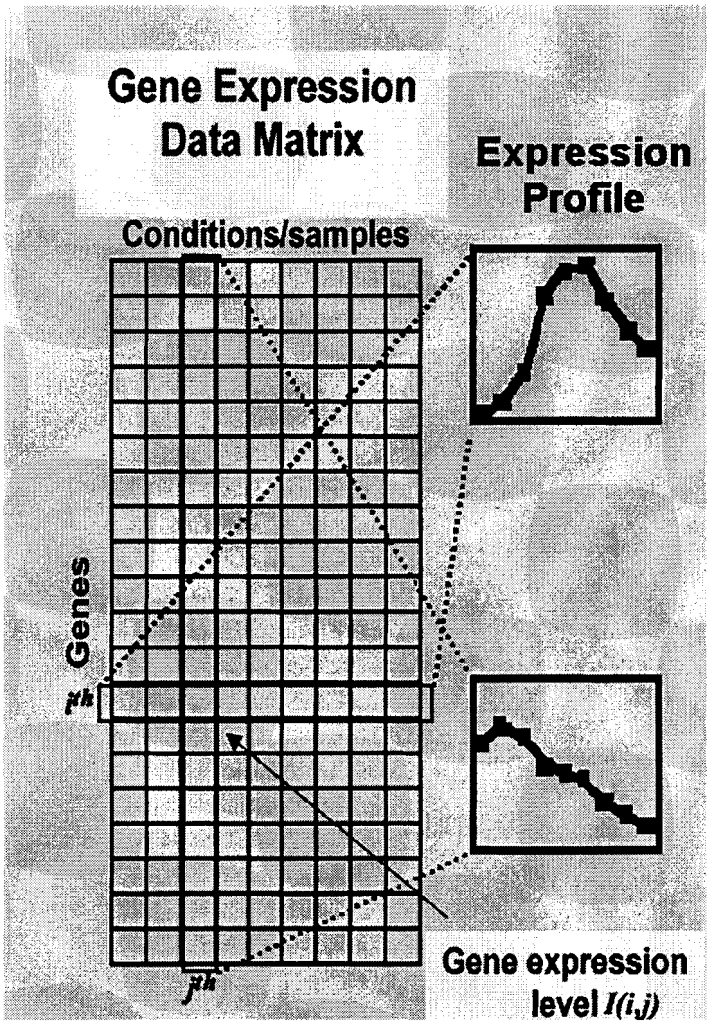


Figure 2.2: A sample of gene expression matrix

2.2 Clustering Gene Expression Data

It is difficult for researchers to interpret large amount of gene expression data without computational methods. Understandably, it needs a very careful analysis because biological signals may be hidden by experimental noise. Hence, the development of computational techniques for interpreting large amounts of gene expression data is a major challenge in functional genomics. According to Eisen et al. [2], this research area needs a holistic approach to the analysis of genome data that reflects the order in the whole set of observations, allowing biologists to develop an integrated understanding of the process being studied. Other researchers, like D'haeseleer et al. [30] for instance, uses computational methods that focus on identification of genes that are defined as significant for the intended purpose, instead of focusing on the whole data set. Hence, the methods that traditionally have been performed manually by biologists, like finding genes with significant change in expression, can be done in a more formal fashion by using different computational techniques.

In cluster analysis, one wishes to partition entities into groups based on given features of each entity, so that the groups are homogeneous and well separated. Each group is called a *cluster,* and the partition is called *clustering*. Clustering problems arise in numerous disciplines including biology, medicine, psychology, economics and others. Since there is a tight connection between a gene's function and its expression pattern [31], an assumption that is frequently made in many studies is that genes should be organized according to the similarities of their expression profiles [32]. Since the idea behind clustering methods is to group similar data points together [4], this approach has

been widely applied to gene expression analysis in terms of grouping together genes with similar expression patterns [31].

Analyzing multi-conditional gene expression patterns with clustering algorithms involves the following steps:

1. Determination of the gene expression data. The gene expression matrix (Figure 2.2) can be represented by a real-valued expression matrix I where I_{ij} is the measured expression level of gene i in experiment condition j. Expression levels should ideally be absolute, but often only relative levels are available. The i^{th} row of the matrix is a vector forming the expression pattern of gene i.
2. Calculation of a similarity matrix. In this matrix the entry S_{ij} represents the similarity of the expression patterns for genes i and j. Many possible similarity measures can be used here. A good choice of measure depends on the nature of the biological question and on the technology that was used to obtain the data. The similarity measure will be briefly discussed in section 2.3.3.
3. Clustering based on the gene expression matrix as well as on the similarity matrix. Genes that belong to the same cluster should have similar expression patterns, while different clusters should have distinct well-separated patterns.
4. Representations of the constructed solution. As hundreds or thousands of genes are involved, visualization tools are crucial for organizing, understanding and exploiting the results.

2.3 Challenges

2.3.1 Number of Clusters

Clustering is a very useful and important technique for analyzing gene expression data. Most clustering methods perform well when the number of clusters is given. However, identifying the number of clusters available in gene expression data is by itself a challenging task. Most clustering techniques require the number of clusters to be given prior to clustering. However, this information, especially in gene expression data, is usually unknown before clustering. Hence, various validation schemes are commonly used to choose the best number of clusters. In this thesis, a novel extension of the popular self organizing maps (SOM) known as adaptive double self-organizing map (ADSOM) is introduced to perform clustering and cluster visualization simultaneously, thereby requiring no *a priori* knowledge about the number of clusters.

2.3.2 Cluster Validation

Cluster validation is another challenge in gene expression data analysis. To identify the number of clusters, researchers commonly cluster the data by choosing different number of clusters heuristically and validate all the clustering results externally. This process is complicated and time-consuming. In addition, there are many different kinds of validation techniques such as figure of merit and Xie-Beni index. The selection of suitable validation technique for the chosen clustering method is very important for getting good results [13], [36].

This thesis introduces a new approach called tree-based index to help validate the number of clusters. This new approach is especially suitable to ADSOM.

2.3.3 Other Challenges

Choosing suitable similarity measure is also very important for analyzing the multi-condition gene expression In the similarity matrix *S*, the similarity between two gene expression levels in the original data is transferred to a single value, called pairwise similarity. After clustering the genes based on the similarity measures, the genes that belong to the same cluster should have similar expression pattern, while different clusters will have distinct or well-separated patterns. The two most popularly used similarity measures are Euclidean distance and correlation.

Euclidean distance is defined as:

$$d_{ij} = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \tag{2.1}$$

where d_{ij} is the distance between gene i and j, and x_{ik} and x_{jk} are the k^{th} expression values of the gene i and j, respectively. If Euclidean distance is chosen as similarity measure, the smaller the distance (d_{ij}) is, the more similar genes i and j are.

Correlation is used as an alternative approach similarity measure described by Eisen *et al* [2]. Correlation is defined as:

$$\rho_{ij} = \frac{\sum_{k=1}^{n} (x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j)}{\sqrt{[\sum_{k=1}^{n} (x_{ik} - \bar{x}_i)^2][\sum_{k=1}^{n} (x_{jk} - \bar{x}_j)^2]}} \tag{2.2}$$

where ρ_{ij} represents the correlation between gene i and j, x_{ik} and x_{jk} are the k^{th} expression values of the gene i and j respectively, and $\bar{x}_i$ and $\bar{x}_j$ are the mean expression values of gene i and j, respectively. As a similarity measure, higher correlation indicates that the corresponding genes are more similar.

Different similarity measures yield different results. For example, two genes, which have very high correlation, may have large Euclidean distance between each other. This problem comes out more frequently, if the data is unnormalized. Unfortunately, there are no general guidelines in the literature to determine which of these two candidates is better. Finding a method for comparing them is a challenging task.

Normalization is another challenge for gene clustering. It is a crucial step for preprocessing the data. Range normalization and standard normalization are the most commonly used normalization methods.

Gene expression data from microarray chips involve substantial noise. Commonly, researchers use two-fold or three-fold methods to filter noise.

Identifying outliers in the gene expression data is very important as well. A feature vector is an outlier if it is distant from all cluster centers. Clustering performance is strongly affected by the existence of outliers. Unless the outliers are identified and eliminated, they can influence the formation of clusters by competing with the rest of feature vector to attract the cluster centers.

This thesis mainly addresses the challenges described in sections 2.3.1 and 2.3.2. Careful selection of similarity measure, normalization scheme and filtering techniques are also made in the experiments conducted in this thesis.

CHAPTER 3

CLUSTERING METHODS

This chapter briefly describes a few popular clustering methods that have been applied in gene expression analysis. Some techniques that are explicitly used for cluster validation are described in Chapter 4.

3.1 Self-Organizing Map

The most typical notion of the self-organizing map (SOM) is to consider it as an artificial neural network model of the brain, especially of the experimentally found ordered "maps" in the cortex. There exists a lot of neurophysiological evidence to support the idea that the SOM captures some of the fundamental processing principles of the brain.

The design of SOM starts with defining a geometric configuration for the partitions in a one- or two-dimensional grid. Then, random weight (reference) vectors are assigned to each partition. During training, a gene expression profile (input vector) is picked randomly. The weight vector closest to the expression profile is identified. The identified weight vector and its neighbors are adjusted to look similar to the expression profile. This process is repeated until the weight vectors converge to a prespecified degree.

SOM algorithm is described as follows:

1. *Initialization*: Choose random values for the initial weight vectors $\boldsymbol{w}_j(0), j = 1,2,...N$, where N is the number of nodes.

2. *Sampling:* At each epoch k, choose an input vector $\boldsymbol{x}(k)$ from the input space with a certain probability.

3. *Similarity* Matching: Find the winning neuron $\boldsymbol{w}_w$ by using the minimum Euclidean distances criterion. $\boldsymbol{w} = \arg\min_j \left\| \boldsymbol{x}(k) - \boldsymbol{w}_j(k) \right\|, j = 1,2,...N$ (3.1)

4. *Updating*: Adjust the weight vector of the winning neuron w_w and its neighbors by using Equation (2).

$$\boldsymbol{w}_j(k+1) = \boldsymbol{w}_j(k) + \eta(k) h_{j,w}(k)[\boldsymbol{x}(k) - \boldsymbol{w}_j(k)] \quad (3.2)$$

Where $h_{j,w}(k)$ is a given neighborhood function.

5. *Continuation*: Repeat steps 2-4 until no noticeable change in the feature map is observed.

SOM has a number of features that make it particularly well suited to cluster gene expression patterns. However, there are some fundamental problems with this method, which are also present in most other clustering algorithms. One of those is that SOM is not suitable for visualization of high-dimensional data. It requires the aid of other techniques, for example Umatrix [23]. Another one is the issue of determining the number of clusters in the whole data. The more the number of clusters is, the tighter and more distinct clusters appear. But adding new nodes doesn't always supply the clustering with fundamentally new patterns and will make the interpretation worse.

In particular, in circumstances where knowledge about the number of clusters is unavailable like most gene expression data, use of SOM as a clustering tool will be time consuming. In which case, one will be forced to apply a trial and error approach to validate the clustering result. For example, one may start with a small number m, cluster the data into m groups and evaluate the result, increase the number m to $m+1$, cluster the data and evaluate the data again, and so on so forth. One needs to increase the number m by 1 continuously, and evaluate the result every time. This trial-and error process is inconvenient and very time-consuming.

3.2 Adaptive Resonance Theory

Adaptive resonance theory (ART) allows to dynamically add nodes as needed by the data. It has one simple parameter "vigilance" to vary and its convergence is guaranteed within a few (often one) epochs [34]. Therefore, ART can cluster gene expression data effectively. Previous applications of ART on gene clustering have shown that it is robust to noise and able to create fine distinctions even with the same number of clusters when compared to other clustering algorithms [8].

However, to use the ART algorithm effectively, some details of its implementation must be properly understood. Unlike most clustering algorithms that use Euclidean distance (L2), ART uses the City Block distance (L1). The L1 distance is computationally quicker than the L2 distance. However, this is partly offset because ART uses a process called "complement coding" that represents each cluster (node) in twice its original input dimensions. For example, a 13-dimensional input data is clustered into a 26 dimensional node in ART architecture. Therefore, unless otherwise

noted, the distance between two nodes (internode distance) will be measured in the L1 distance between complement-coded nodes. This is because L2 distance does not directly correspond to any part of the ART algorithm. The ART parameter "vigilance" dictates whether two genes should be grouped into the same cluster or not; higher vigilances lead to more nodes and lower vigilances lead to fewer nodes. Note that in SOM, one needs to input the number of clusters while in ART the number of clusters is controlled by the vigilance factor. The only required adjusted parameter is vigilance that ranges from 0 to 1. Note that the use of ART involves a heuristic selection of vigilance or some external validation techniques to arrive at a reasonable number of clusters.

3.3 Fuzzy C-Means

Fuzzy c-means (FCM) clustering facilitates the identification of overlapping groups of objects by allowing the objects to belong to more than one group. The essential difference between FCM and hard c-means is the partitioning of genes into each group. Instead of hard partitioning, where genes belong to only a single cluster, FCM clustering considers each gene to be a member of every cluster, with a variable degree of membership. Each gene has a total membership of 1.0 that is apportioned to clusters on the basis of the similarity between the gene's expression pattern and that of each cluster centroid. Genes whose expression patterns are very similar to a given centroid will be assigned a high membership in that cluster, whereas genes that bear little similarity to the centroid will have a low membership. Hence, FCM computes the fuzzy partition matrix U whose ik^{th} element $u_{ik} \in [0,1]$ expresses the membership degree of the sample $\boldsymbol{x}(k)$ to cluster i, $i=1..c$.

3.4 Model-Based Clustering

Raftery et al. [18] [19] introduced a clustering algorithm based on probability models which offers an alternative to heuristic-based algorithms. Particularly, the model-based approach assumes that the data is generated by a finite mixture of underlying probability distributions such as multivariate normal distributions. With the underlying probability model, the problems of determining the number of clusters and of choosing an appropriate clustering method become statistical model choice problems.

Suppose the data x consists of independent multivariate components $x_1, x_2, \ldots, x_n$. In the Gaussian mixture model, each component k is modeled by the multivariate normal distribution with parameters mean vector μ_k and covariance matrix Σ_k:

$$f_k(x_i \mid \mu_k, \Sigma_k) = \frac{\exp\left[-\frac{1}{2}(x_i - u_k)^T \Sigma_k^{-1}(x_i - \mu_k)\right]}{\sqrt{\|2\pi\Sigma_k\|}}$$

The covariance matrix Σ_k determines geometric features, such as shape, volume and orientation. Banfield et al. [42] proposed a general framework to exploit the presentation of the covariance matrix in terms of its eigenvalue decomposition

$$\Sigma_k = \lambda_k Q_k P_k Q_k^T$$

where Q_k is the orthogonal matrix of eigenvectors which dertermines the orientation of the component; P_k is a diagonal matrix whose elements are proportional to the eigenvalues of Σ_k and P_k determines the shape of the component; λ_k is a scalar which determines the volume of the component.

The most important models are determined by four different types of Σ_k. These four models are equal volume spherical model (EI), unequal volume spherical model (VI), ecliptical model with equal volume, shape and orientation (EEE) and unconstrained model (VVV). EI model is parameterized by $\Sigma_k = \lambda I$, I is the identity matrix. In VI, $\Sigma_k = \lambda_k I$. In EEE, $\Sigma_k = \lambda QPQ^T$. VVV model allows all of λ_k, Q_k, P_k to vary between components.

Instead of using Bayes factor that has the main difficulty in evaluating the integrated likelihood, Yeung et al. [35] suggested using an approximation called the Bayesian Information Criterion (BIC) to compare models with different numbers of clusters and different covariance matrix parameterization. A large BIC score indicates strong evidence for the corresponding model.

CHAPTER 4

CLUSTER VALIDATION METHODS

The clusters obtained by different clustering algorithms can be remarkably different. Without validation procedures, results of clustering algorithms may easily be misinterpreted. This chapter describes some cluster validation methods which will be used later in Chapter 7. These methods include figure of merit, change in internode distance per cluster and Xie-Beni index.

4.1 Figure of Merit

Figure of merit (FOM) is an estimate of the predictive power of a clustering algorithm. It is a systematic and quantitative framework to assess the results of clustering algorithms. Yeung et al [13], introduced FOM to validate clustering performances. They compute FOM for different clustering algorithms so as to solve the problem for choosing an appropriate clustering algorithm for a specific data set.

FOM is described here as defined by Yeung *et al* [13]. A typical gene expression data set contains measurements of expression levels of n genes under m conditions. Suppose a clustering algorithm is applied to the data from condition 1,2,3,..,(e-1), (e+1),...,m and condition e is used to estimate the predictive power of the algorithm. Suppose there are k clusters, c_1, c_2...,c_k. Let $R(g,e)$ be the expression level of gene g under condition e in the raw data matrix. Let $Uc_i(e)$ be the average expression level in condition e of genes in clusters c_i. So, the FOM under the condition e is defined as:

$$\mathrm{FOM}(e,k) = \sqrt{\frac{1}{n}\sum_{i=1}^{k}\sum_{x\in c_i}\left[R(x,e) - Uc_i(e)\right]^2} \tag{4.1}$$

And then, the aggregate figure of merit of all conditions is defined:

$$\mathrm{FOM}(k) = \sum_{e=1}^{m} FOM(e,k) \tag{4.2}$$

After computing FOM for different number of clusters and plotting FOM versus number of clusters, we expect FOM to decrease as the number of clusters increases saturating eventually. If a curve enters its saturation region and the corresponding number of cluster at that point is N, we say N nodes are sufficient to cluster the data.

4.2 Change in Internode Distance per Cluster

Musavi et al. [3] applied the method change in "internode distance per cluster" to identify the number of clusters obtained using ART. To investigate how a "good" number of clusters can be selected for a gene expression data, let average internode distance be $D(\theta)$, where θ is the algorithm's parameters vigilance. Furthermore, let $N(\theta)$ be the number of clusters formed by the algorithm for a given θ. Note that $D(\theta)$ is necessarily a function of $N(\theta)$, and can be written as $D(\theta,N(\theta))$. Plotting $D(\theta)$ as a function of $N(\theta)$, one normally observes the saturation point where adding further nodes does not decrease internode distance significantly. For example, consider Figure 4.1, which shows the internode distance versus the number of clusters. As stated before, the internode distance approaches a saturation level as the number of clusters increases. Using this curve for finding an appropriate number of clusters may prove to be hard because of the difficulty in identifying the point of saturation.

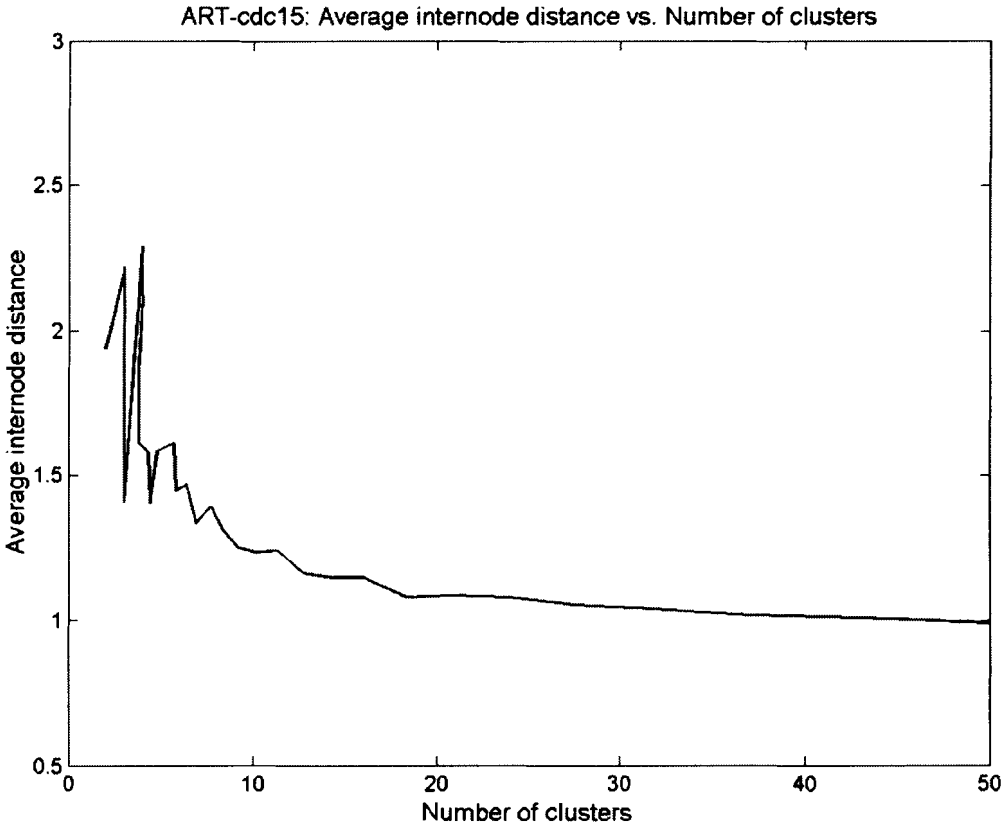


Figure 4.1: An example of Internode distance vs. the number of nodes.

The internode distance per node is measured with $D(\theta)/N(\theta)$. The effect of decreasing internode distance per node can be measured by plotting $D'(\theta)/N(\theta)$ versus $N(\theta)$, where $D'(\theta)$ is the first partial derivative of $D(\theta)$ with respect to $N(\theta)$. Now, it can be stated that when $D'(\theta)/N(\theta)$ approaches 0, adding more nodes does not significantly decrease internode distance. After this point, overclustering occurs.

4.3 Xie-Beni Index

This method is particularly suitable to validate the clustering results formed by fuzzy c-means. We assessed the goodness of each resulting partition using the Xie-Beni index [36], which computes the ratio of compactness and separation of clusters as follows:

$$\chi(U,V,\boldsymbol{x}) = \frac{\Sigma_{i=1}^{c}\Sigma_{k=1}^{N} u_{ik}^{m}\left\|\boldsymbol{x}(k)-\underline{V}_i\right\|^2}{N.\left(\min_{i\neq j}\left\|\underline{V}_i-\underline{V}_j\right\|^2\right)}$$

where N is the number of training samples, $\boldsymbol{x}(k)$ is the kth input vector (sample), and $\underline{V}_i$ is the ith cluster prototype. After plotting the Xie-Beni index versus c and choose, as optimal number of clusters, the value of c corresponds to the first distinctive local minimum.

CHAPTER 5

ADAPTIVE DOUBLE SELF-ORGANIZING MAP (ADSOM)

5.1 Double Self-Organizing Map

Double self-organizing map (DSOM), introduced by Su and Chang [28], adjusts its network structure during the learning phase so that neurons which respond to similar stimuli will not only have similar weight vectors but also move spatially nearer to each other. This is accomplished by combining features of self-organizing maps (SOM) with position vectors, which serve as a visualization tool to decide how many clusters are needed.

In DSOM, as described in [28], each node j has an N-dimensional synaptic weight vector $\boldsymbol{w}_j$. In addition to the weight vector, another two-dimensional position vector $\boldsymbol{p}_j$ is also assigned to each neuron j. The vector $\boldsymbol{p}_j$ determines the position of neuron j in the network structure. During the self-organizing process, not only the weight vector $\boldsymbol{w}_j$'s but also the position vector $\boldsymbol{p}_j$'s are updated.

All the updating formulae for both vectors at epoch k are given in Equations (5.1)-(5.6):

$$\boldsymbol{w}_j(k+1) = \boldsymbol{w}_j(k) + \eta_1(k)\Lambda_{j,w}(k)\left[\boldsymbol{x}(k) - \boldsymbol{w}_j(k)\right], \quad j = 1,2,...N \tag{5.1}$$

$$\boldsymbol{p}_j(k+1) = \boldsymbol{p}_j(k) + \eta_{2,j}(k)h_{j,w}(k)\left[\boldsymbol{p}_w(k) - \boldsymbol{p}_j(k)\right], \quad j = 1,2,...N \tag{5.2}$$

where

$$\eta_1(k) = \eta_w \frac{1}{k+1}, \tag{5.3}$$

$$\Lambda_{j,w}(k) = \exp\left[- s_w \left(1 + \frac{k}{k_{\max}} \right) \| \boldsymbol{p}_j(k) - \boldsymbol{p}_w(k) \|^2 \right], \tag{5.4}$$

$$\eta_{2,j}(k) = \frac{\eta_p}{1+k} \exp\left[- s_p \left(1 + \frac{k}{k_{\max}} \right) \| \boldsymbol{p}_j(k) - \boldsymbol{p}_w(k) \| \right], \text{ and} \tag{5.5}$$

$$h_{j,w}(k) = \exp\left\{ - s_x \left(1 + \frac{k}{k_{\max}} \right) \left[\| \boldsymbol{w}_j(k) - \boldsymbol{x}(k) \| - \| \boldsymbol{w}_w(k) - \boldsymbol{x}(k) \| \right]^2 \right\} \tag{5.6}$$

Here $\|\bullet\|$ denotes the Euclidean distance; $\boldsymbol{p}_w(k)$ and $\boldsymbol{w}_w(k)$ represent the position and weight vectors of the winning neuron when the k^{th} input vector $\boldsymbol{x}(k)$ is applied; η_w and η_p are initial learning rates, s_w is a scalar parameter which regulate how fast the function $\Lambda_{j,w}(k)$ decreases, and k_{max} is the maximum number of epochs. s_p and s_x are two predetermined scalar parameters which regulate the speed of the movement of the position vectors.

For an instance, there are m (actually we don't know the appropriate number of clusters) classes in a data set. Before clustering, one initializes n nodes and their corresponding position vectors. Here n must be no less than m. Based on formulae (5.1-5.6), we can find that the closer the nodes (weights) are, the closer the corresponding position vectors will be. Theoretically, the n position vectors should move into the m groups after clustering. Because the position vectors are two-dimensional, we can visualize the number of groups of the position vectors by plotting them. As a result, we can determine the number of clusters available in the underlying data set.

DSOM constructs a relationship between weight vector and position vector by using a few non-linear exponential functions. This method will not ruin the inner relations among those weight vectors and will not lose any possibly important information preserved in the high dimensional weight vectors.

5.2 Adaptive Self-Organizing Map

Although DSOM addresses to the problem of deciding number of clusters, the selection of its free parameters, which is important to a proper projection of the position vectors in a two-dimensional space, remains a challenge. Some combinations of these parameters make all the position vectors converge too quickly into a small dense area; some other combinations lead the updating process to "get stuck" after several epochs and result in wrong number of clusters. Thus, the regulation of these parameters is a very important task. In this thesis, a novel adaptive self-organizing map (ADSOM) was developed by implementing a systematic method of updating the free parameters involved in DSOM during the training process with the aim of guaranteeing convergence. Looking at Equations (5.2), (5.5) and (5.6), one see that there are several parameters such as s_x, s_p, and number of epochs, which affect the movement of position vectors. In Equation (5.4), $\boldsymbol{p}_w(k)$ is the position vector corresponding to a winner weight at epoch k; $\boldsymbol{p}_j(k)$ and $\boldsymbol{p}_j(k+1)$ are the old and new j^{th} position vectors, respectively. $\eta_{2,j}(k)$ and $h_{j,w}(k)$ are both positive scalars and are always less than 1.

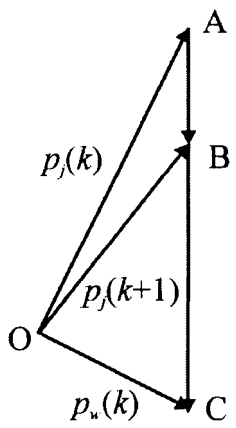


Figure 5.1: Movement of position vector.

In Figure 5.1, OA, OB and OC represent vectors $\boldsymbol{p}_j(k)$, $\boldsymbol{p}_j(k+1)$, and $\boldsymbol{p}_w(k)$ respectively. According to Equation (5.2), one can easily prove that points A, B, and C are located on a straight line. Let's define a positive scalar $\Phi_j(k)$ as follows:

$$\Phi_j(k) = \eta_{2,j}(k) h_w(k)$$

It is obvious that the ratio between the length of vector $\overrightarrow{AB}$ and $\overrightarrow{AC}$, which actually represents the vectors $\boldsymbol{p}_j(k+1) - \boldsymbol{p}_j(k)$ and $\boldsymbol{p}_w(k) - \boldsymbol{p}_j(k)$ respectively, is $\Phi_j(k)$ (from Equation 5.2). A conclusion can be drawn that the larger $\Phi_j(k)$ is, the faster the j^{th} position vector $\boldsymbol{p}_j$ moves toward $\boldsymbol{p}_w$.

Assuming there are two weights $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$, and their corresponding position vectors are $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$. At epoch k and for a given input pattern x(k), if $\boldsymbol{w}_1(k)$ is closer to $\boldsymbol{x}(k)$ than $\boldsymbol{w}_2(k)$, which means $||\boldsymbol{w}_1(k)$-$\boldsymbol{x}(k)||$ is less than or equal to $||\boldsymbol{w}_2(k)$-$\boldsymbol{x}(k)||$, one requires that $\boldsymbol{p}_1(k)$ moves toward $\boldsymbol{p}_w(k)$ faster than $\boldsymbol{p}_2(k)$, which means $\Phi_1(k) \geq \Phi_2(k)$. In another words, Equation (5.7) as follows should be true to true to keep position vectors from "getting stuck" and avoid unexpected results.

$$\left[\Phi_j(k)-\Phi_i(k)\right]\left[\left\|\boldsymbol{w}_j(k)-\boldsymbol{x}(k)\right\|-\left\|\boldsymbol{w}_i(k)-\boldsymbol{x}(k)\right\|\right]\le 0 \ \ for \ \ \forall i,j(i\neq j) \tag{5.7}$$

In order to study Equation (5.7) more flexibility, instead of using constant values for s_p and s_x, we suggest adopting them at every epoch. In another word, $s_p(k)$ *and* $s_x(k)$ are used. Moreover, we replace the power to $\left\|\boldsymbol{w}_i(k)-\boldsymbol{x}(k)\right\|-\left\|\boldsymbol{w}_w(k)-\boldsymbol{x}(k)\right\|$ in Equation (5.6) from *2* to a variable *n* (n>0) and replace the power to $\left\|\boldsymbol{p}_i(k)-\boldsymbol{p}_w(k)\right\|$ in Equation (5.5) from 1 to a variable *m* (m>0).

Assuming $A_i(k)=\left\|\boldsymbol{w}_i(k)-\boldsymbol{x}(k)\right\|-\left\|\boldsymbol{w}_w(k)-\boldsymbol{x}(k)\right\|$, Equation (5.7) will be equivalent to (5.8).

$$\left[\Phi_j(k)-\Phi_i(k)\right]\left[A_j(k)-A_i(k)\right]\le 0,\forall i,j(i\neq j) \tag{5.8}$$

Defining $B_i(k)=\left\|\boldsymbol{p}_i(k)-\boldsymbol{p}_w(k)\right\|$, $s_{x'}(k)=s_x(k)\left(1+\frac{k}{k_{\max}}\right)$, and $s_{p'}(k)=s_p(k)\left(1+\frac{k}{k_{\max}}\right)$,

$$\Phi_i(k)=\frac{\eta p}{1+k}\exp\left[-s_{p'}(k)B_i^m(k)-s_{x'}(k)A_i^n(k)\right]$$

If $A_j(k)=A_i(k)$, then Equation (5.8) is always valid regardless of the relationship between $\Phi_i(k)$ and $\Phi_j(k)$. So, in the following, we just need to analyze the cases when $A_j(k)\neq A_i(k)$.

Case 1:

If $A_j(k)>A_i(k)$, then $\Phi_j(k)\le\Phi_i(k)$

$$\Leftrightarrow \exp\left[-s_{p'}(k)B_j^m(k)-s_{x'}(k)A_j^n(k)\right] \le \exp\left[-s_{p'}(k)B_i^m(k)-s_{x'}(k)A_i^n(k)\right]$$

$$\Leftrightarrow s_{p'}(k)B_j^m(k)+s_{x'}(k)A_j^n(k) \ge s_{p'}(k)B_i^m(k)+s_{x'}(k)A_i^n(k)$$

$$\Leftrightarrow s_p(k)B_j^m(k)+s_x(k)A_j^n(k) \ge s_p(k)B_i^m(k)+s_x(k)A_i^n(k)$$

$$\Leftrightarrow s_p(k)\left[B_i^m(k)-B_j^m(k)\right] \le s_x(k)\left[A_j^n(k)-A_i^n(k)\right] \tag{5.9}$$

$\because A_j(k) > A_i(k) > 0, \therefore A_j^n(k) - A_i^n(k) \ge 0$; and since $s_x(k) > 0, s_p(k) > 0$, Equation (5.9) is equal to

$$\frac{s_x(k)}{s_p(k)} > \frac{B_i^m(k)-B_j^m(k)}{A_j^n(k)-A_i^n(k)} \forall i, j (i \ne j) \tag{5.10}$$

Case 2:

If $A_j(k) < A_i(k)$, then $\Phi_j(k) > \Phi_i(k)$

Following the same steps used in case 1, we can prove that

$$\frac{s_x(k)}{s_p(k)} \ge \frac{B_i^m(k)-B_j^m(k)}{A_j^n(k)-A_i^n(k)} \forall i, j (i \ne j) \tag{5.11}$$

From Equations (5.10) and (5.11), we can draw a conclusion that Equation (5.8) holds, if

$$\frac{s_x(k)}{s_p(k)} \ge \frac{B_i^m(k)-B_j^m(k)}{A_j^n(k)-A_i^n(k)} \forall i, j (i \ne j) \tag{5.12}$$

Equation (5.12) specifies one end of the boundary for the ratio $\frac{s_x(k)}{s_p(k)}$. However, the other end of the boundary is not specified. It is clear that closing both sides of the

boundaries will be helpful to provide a more complete range of this ratio, thus further improve the accuracy and effectiveness of ADSOM. This task will be accomplished in the future. In this thesis, we use a particular choice of this ratio as follows:

Defining $t_u(k) = \max_v \left\{ \frac{B_u^m(k) - B_v^m(k)}{A_v^n(k) - A_u^n(k)} \right\}, \forall u, v, (v \neq u)$, we specify the following relationship that allows Equation (5.12) to be satisfied and which we found to be effective in our experiments.

$$\frac{s_x(k)}{s_p(k)} = \begin{cases} \text{unchanged}, & \text{if } \max_u \{t_u(k)\} \leq 0 \\ 1.2 \max_u \{t_u(k)\}, & \text{if } \max_u \{t_u(k)\} > 0 \end{cases} \tag{5.13}$$

Beside the problem mentioned above, defining a criterion for stopping the updating process is also a principal issue. On one hand, over-training will unexpectedly worsen the results. On the other hand, under-training will result in uncertain outcome. So, instead of directly finding out a way to intelligently select the number of epochs, we define a new parameter that would serve as a stopping criterion.

We define the maximum change of position vectors (α) at epoch k as $\alpha(k) = \max_l \{ |\boldsymbol{p}_l(k) - \boldsymbol{p}_l(k-1)| \}$, and we define a threshold (β) as 0.1% of largest distance among all the original position vectors. If the α remains less than β, the training process will be ceased.

Finally, we demonstrate a proper mathematical relationship between α and the parameter $s_p(k)$ so that we can adaptively adjust $s_p(k)$ at each epoch k. Once we know $s_p(k)$ we can use Equation (5.13) to adjust the parameter $s_x(k)$

If α becomes larger, which means the position vectors move faster, one needs to reduce the speed of movement of position vectors. From Equation (5.4), the parameter s_p needs to be increased. Similarly, if α becomes smaller, the parameter s_p needs to be decreased. So, at epoch k, our objective is to figure out a mathematical continuous function $f(y)$ to represent the relationship between the α and s_p. It can be shown that one can update s_p by using the following function:

$$s_p(k+1)=\begin{cases} s_p(k)f\left(\dfrac{\alpha(k)}{\alpha(k-1)}-1\right), & \text{if} \quad k\geq 2 \\ s_p(k)=\text{const}>0 & \text{if} \quad k=1 \end{cases} \tag{5.14}$$

According to the analysis in the previous paragraph, one can easily infer that $f(y)$ is an increasing function, whose range should be between 0 and a constant positive value C. In addition, when $\dfrac{\alpha(k)}{\alpha(k-1)}=1$, one requires $f\left(\dfrac{\alpha(k)}{\alpha(k-1)}-1\right)=1$; when $\dfrac{\alpha(k)}{\alpha(k-1)}<1$, $0<f\left(\dfrac{\alpha(k)}{\alpha(k-1)}-1\right)<1$; when $\dfrac{\alpha(k)}{\alpha(k-1)}>1$, $1<f\left(\dfrac{\alpha(k)}{\alpha(k-1)}-1\right)<C$. So we suggest using any function $f(y)$, which satisfies the following requirements:

(1) $f(y)\in(0,C)$ for $y\in(-1,+\infty)$, where C is a constant positive number,.

(2) $f(y)$ is a continuous, increasing, non-linear function.

(3) $f(0)=1$

In fact, there are many functions, which satisfy all the property mentioned above. These function include $f(y)=\dfrac{e}{e-1}(1-e^{-(y+1)})$ and $f(y)=e^{\left(\frac{y}{y+1}\right)}$. The former one was used in the experiments shown in chapter 7. It is possible that some other functions may

be more suitable to present the relationship between α and s_p. It is also possible that the functions can be discrete ones.

In summary, ADSOM's algorithm repeats the following equations to adaptively update weights and position vectors until α is steadily less than β, which is a pre-defined threshold.

$$\boldsymbol{w}_j(k+1) = \boldsymbol{w}_j(k) + \eta_1(k)\Lambda_{j,w}(k)\left[\boldsymbol{x}(k) - \boldsymbol{w}_j(k)\right], j = 1,2,...N$$

$$\boldsymbol{p}_j(k+1) = \boldsymbol{p}_j(k) + \eta_{2,j}(k)h_{j,w}(k)\left[\boldsymbol{p}_w(k) - \boldsymbol{p}_j(k)\right], j = 1,2...N$$

$$\eta_1(k) = \eta_w \frac{1}{k+1}$$

$$\Lambda_{j,w}(k) = \exp\left[-s_w\left(1+\frac{k}{k_{\max}}\right)\| \boldsymbol{p}_j(k) - \boldsymbol{p}_w(k) \|^2\right]$$

$$\eta_{2,j}(k) = \frac{\eta_p}{1+k}\exp\left[-s_p(k)\left(1+\frac{k}{k_{\max}}\right)\| \boldsymbol{p}_j(k) - \boldsymbol{p}_w(k) \|^m\right]$$

$$h_{j,w}(k) = \exp\left\{-s_x(k)\left(1+\frac{k}{k_{\max}}\right)\left[\| w_j(k) - x(k) \| - \| w_w(k) - x(k) \|\right]^n\right\}$$

$$s_p(k+1) = \begin{cases} s_p(k) f\left(\dfrac{\alpha(k)}{\alpha(k-1)} - 1\right), & \text{if} \quad k \geq 2 \\ s_p(k) = \text{const} > 0 & \text{if} \quad k = 1 \end{cases}$$

$$\frac{s_x(k)}{s_p(k)} = \begin{cases} \text{unchanged}, & \text{if} \max\limits_u \{t_u(k)\} \leq 0 \\ 1.2 \max\limits_u \{t_u(k)\}, & \text{if} \max\limits_u \{t_u(k)\} > 0 \end{cases}$$

where

$$t_u(k) = \max_v \left\{ \frac{B_u^m(k) - B_v^m(k)}{A_v^n(k) - A_u^n(k)} \right\}, \forall u, v, (v \neq u)$$

$$\alpha(k) = \max_l \left\{ \left\| \boldsymbol{p}_l(k) - \boldsymbol{p}_l(k-1) \right\| \right\}, \forall l$$

η_w and η_p are the initial learning rates which are between 0 and 1; s_w is a scalar parameter which regulates how fast the function $\Lambda_w(k,j)$ changes and it's between 0 and 1; k_{max} is the maximum number of epochs; $s_p(k)$ and $s_x(k)$ are the values of scalar parameters at epoch k that regulate the speed of the movement of the position vectors whose initial values are 1; m and n are positive numbers (for example, one may choose n=3 and m=0.3). However, effective choice of m and n needs to be further investigated.

5.3 Initialization Scheme

Proper initialization of the weight vectors of ADSOM helps the parameter updating process converge quickly. The weight initialization scheme provided by Su et al. [33] is used for initialization in ADSOM.

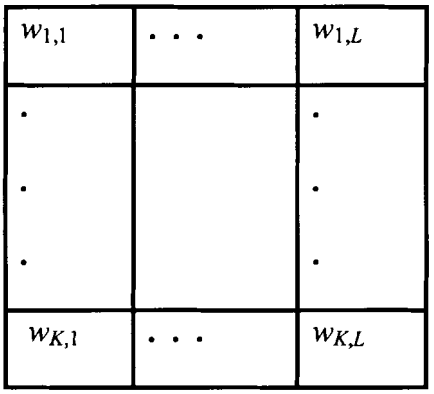


Figure 5.2: The arrangement of an K x L weight array.

Assuming the size of weights array to be $K \times L$ (Figure 5.2), the following steps are used for initialization:

1. Select a pair of input patterns whose distance is the largest one among the whole input data set, and then initialize the weight nodes on the lower left corner and the upper right corner (i.e. $w_{K,1}$ and $w_{1,L}$) as these two input patterns, respectively. From the remaining input data, use the pattern that is the farthest to the two chosen patterns to initialize the weight vector on the upper left corner (i.e. $w_{1,1}$). Set the weight on the lower right corner (i.e. $w_{K,L}$) as the input pattern which is the farthest to the previously selected three patterns.

2. Initialize the weights of the neurons on the four edges by uniformly partitioning each edge into L-1 or K-1 segments.

3. Initialize the remaining weights from left to right, and from top to bottom as follows:

for i from *2* to K-1

for j from *2* to L-1

$$w_{i.j} = \frac{w_{i,L} - w_{i,1}}{L-1}(j-1) + w_{i,1}$$

end

end

As described in [28], the initial map constructed by directly partitioning the input space into $M \times N$ hypercubes and then using the coordinates of the centers of the hypercubes to initialize the weights of the network will tend to undersample high probability regions and oversample low probability ones. As a result, this direct initialization will lead to more iteration to refine the map.

CHAPTER 6

HIERARCHICAL TREE-BASED METHOD FOR VALIDATION

This chapter introduces a novel validation technique that is especially suitable to ADSOM. This method is primary derived from the method of hierarchical tree. It helps to indicate the optimal number of clusters through a tree-based index.

Once the final locations of the position vectors are obtained, one can visually determine the number of clusters. To minimize possible human error resulting from counting the number of clusters visually, a novel technique that provides an index for each cluster is developed. The index is calculated based on the outcome of clustering the position vectors themselves using hierarchical method. Consider a 2-by-N matrix $\boldsymbol{T}$ that contains the final locations of all two-dimensional position vectors; where N denotes the total number of position vectors. We calculate the Euclidean distances between each pair of position vectors in T. Based on these distances and subsequent distances between centers of merged clusters, we create a hierarchical cluster tree that grows vertically upwards. During clustering, we store the distance between two clusters that are merged together through a branch by applying the single linkage method. Hence, the height of each branch provides information about compactness and separation between the clusters connected. It will be large if the compactness is small or the separation is large. Consider a horizontal line that cuts c branches of the hierarchical cluster tree perpendicularly, the horizontal line picks c clusters that are embraced by the branches it

cuts. An index corresponding to this horizontal line is designated as $\gamma(c)$ and is defined as the sum of the heights of each branch it cuts.

$$\gamma(c) = \sum_{i=1}^{c} z_c(i) \qquad c = 2,..N$$

where $z_c(i)$ is the height contributed by the ith branch for the case when a horizontal branch cuts c branches of the tree. The index, referred to as tree-based index, is calculated for $c = 2,..,N$. Note that $\gamma(1)=0$, since there will be no horizontal line that would cut the tree only once. The value of c at which the index reaches its highest peak is considered as an indicator of the maximum separation between clusters in a two-dimensional space. Hence, it is used as a means to detect the number of clusters.

Figure 6.1 shows an example where 18 position vectors (top) are clustered using a hierarchical clustering method (bottom). The three horizontal dashed lines in the figure (bottom) show the branches that they cut for c=2, 3, and 4, respectively. For example, the third dashed line from the top cuts four branches of the tree with the corresponding lengths designated as $Z_4(1)$-$Z_4(4)$.

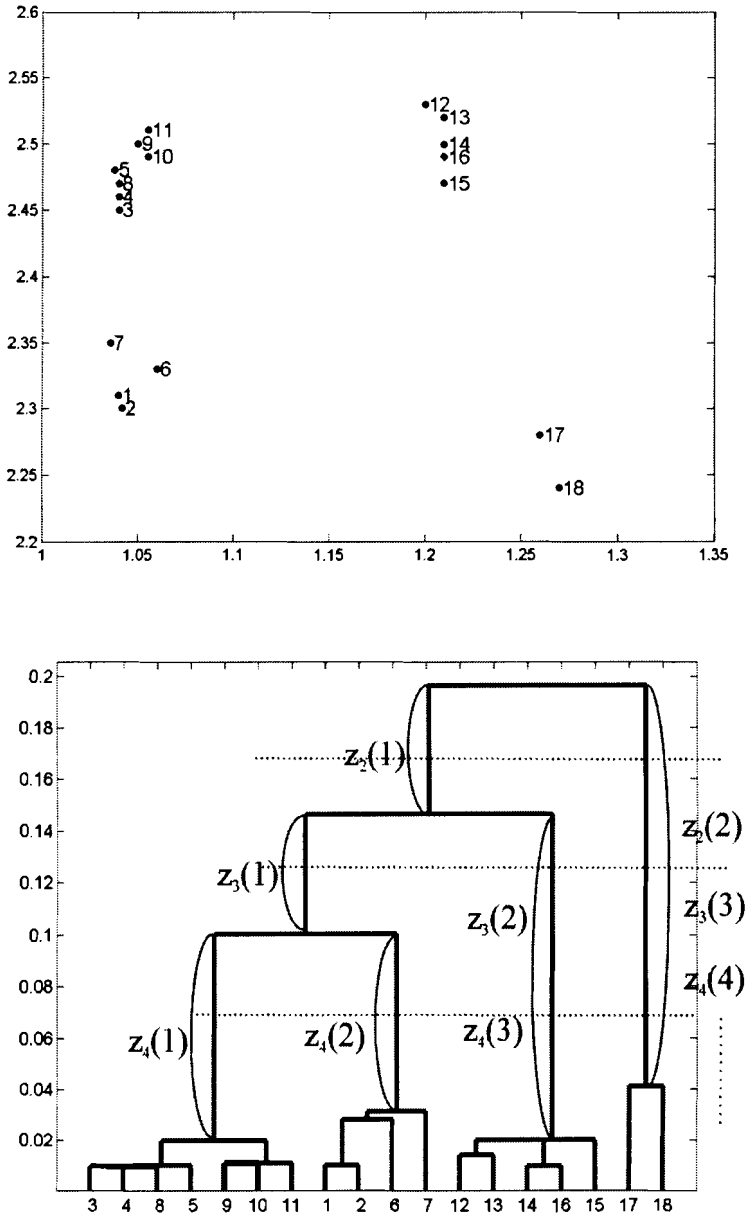


Figure 6.1: An example of hierarchical tree method: position vectors (top) and corresponding hierarchical tree (bottom).

CHAPTER 7

EXPERIMENTAL SCHEMES AND RESULTS

To test ADSOM's performance, experimental tests were conducted using artificial data and gene expression data from multiple biological systems. Other clustering and validating methods such as figure of merit, model-based clustering, and fuzzy c-means were used for comparison.

Table 7.1 shows the parameters of ADSOM used in some of the experiments conducted in this thesis.

7.1 Artificial Data

An artificial data set, which has six groups, was created. The centers of the six groups are chosen as shown in Figure 7.1. Each center is a vector with 17 elements, whose range is between 0 and 1. For every group, 100 random vectors are generated with a variance v=0.25. Hence, each group has 100 sample vectors. The reason for choosing such kind of centers is that the center of the real gene data is similar to one of these centers, or similar to the translation, rotation or combination of these centers.

Data (# of initial nodes)	m	n	η_w	η_p	$s_p(0)$	$s_x(0)$
Artificial data (9)	1	2	0.9	0.9	1	1
Yeast cell cycle data (9)	1	4	0.9	0.9	1	1
Yeast cell cycle data (12)	4	5	0.9	0.9	1	1
Yeast cell cycle data (15)	4	3	0.9	0.9	1	1
Yeast cell cycle data (20)	1	4	0.9	0.9	1	1
Yeast sporulation data (9)	1	2	0.9	0.9	1	1
Yeast sporulation data (20)	1	4	0.9	0.9	1	1

Table 7.1: Parameters of ADSOM used in some of the experiments conducted in this thesis.

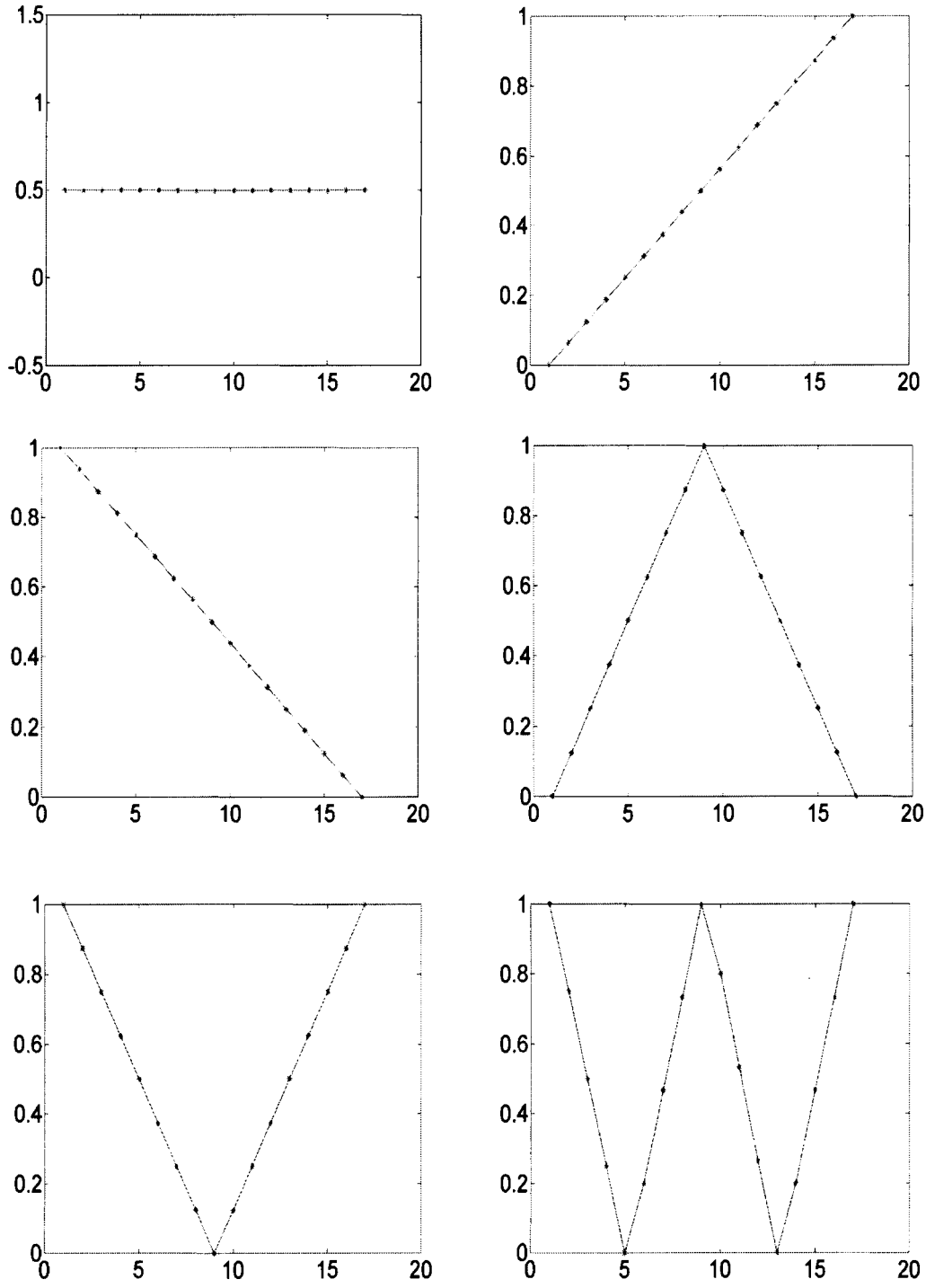
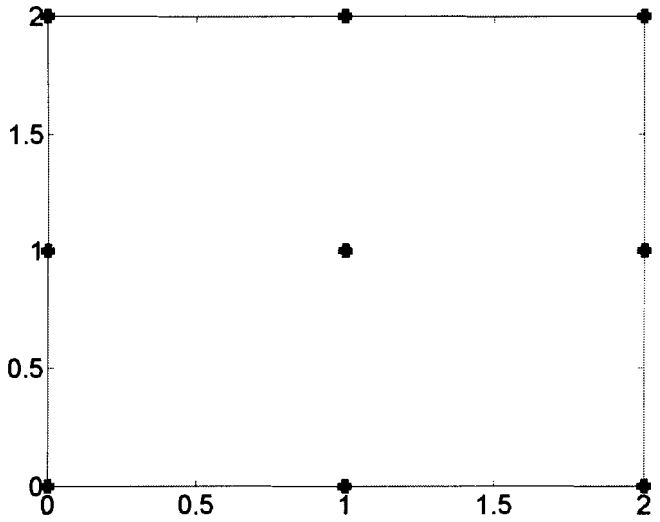
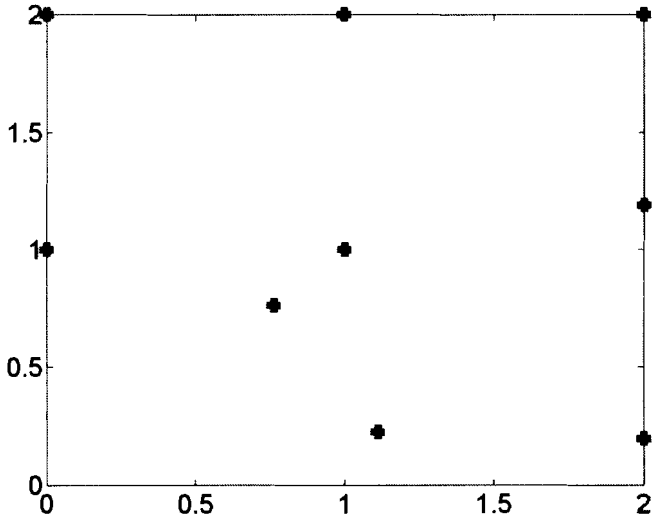


Figure 7.1: Centers of the artificial data set. (Each center has 17 data points)

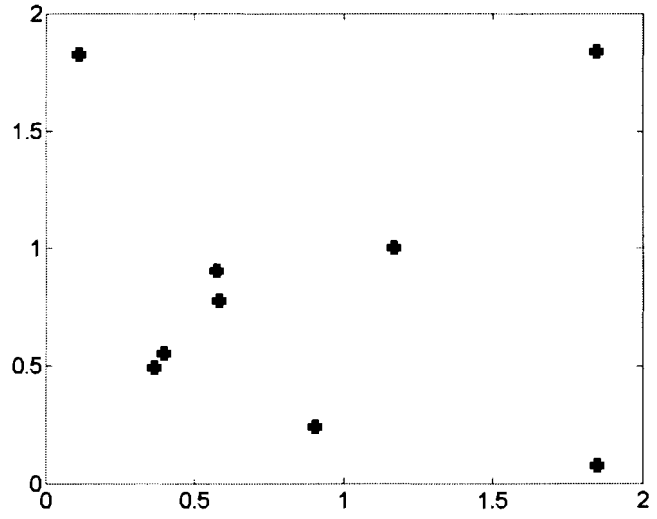


(a) Original position vectors

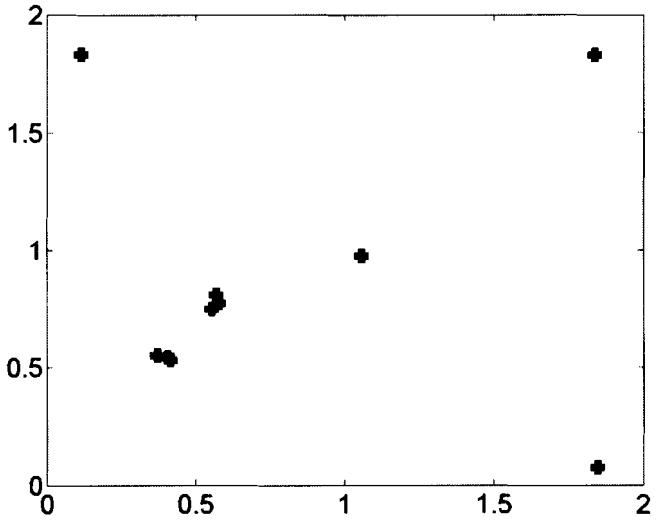


(b) Position vectors after 200 epochs

Figure 7.2: Final position vectors after using ADSOM for the artificial data set.



(c) Position vectors after 500 epochs



(d) Position vectors after 1000 epochs

Figure 7.2: Final position vectors after using ADSOM for the artificial data set (continued).

As shown in Figure 7.2 (a), nine original position vectors are placed onto a 3×3 grid. Then the ADSOM is run for 1000 epochs. The results are shown in Figure 7.2 (b)-Figure 7.2 (d). In Figure 7.2 (d), the nine position vectors fall into the six groups. This shows us that there are six clusters in this artificial data set. In Figure 7.3, the tree based index peaks when number of cluster reaches 6, which indicates that this artificial data have 6 clusters.

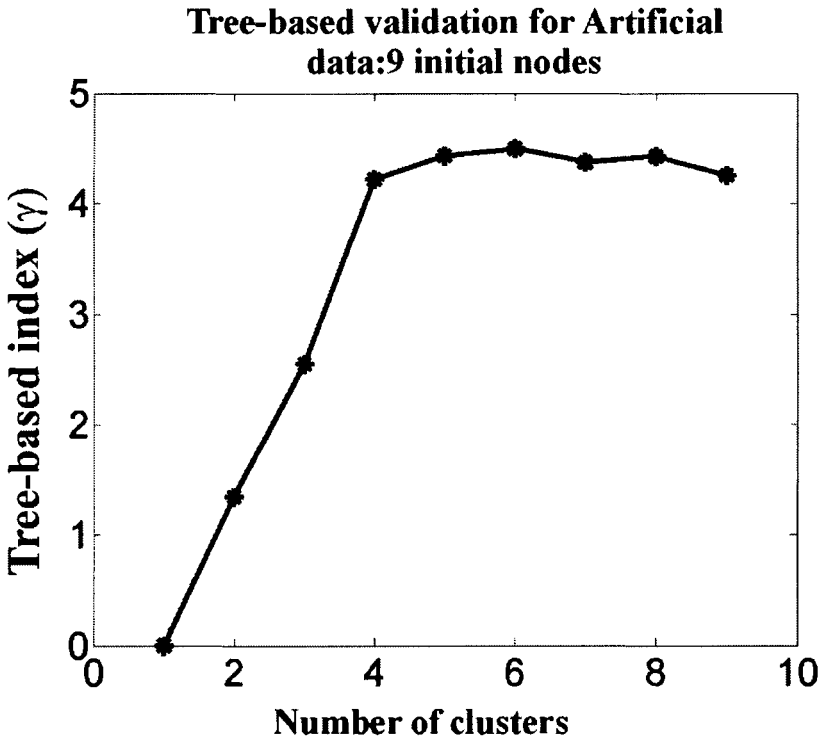


Figure 7.3: Tree based index for artificial data

7.2 Yeast Cell Cycle Data with the 5 Phase Criterion

A subset of the gene expression data provided by Cho et al [37] was extracted. The data show the fluctuation of expression levels of approximately 6000 genes over two cell cycles (17 time points) corresponding to mitotic cell cycle of *Saccharomyces cerevisiae* from affymetrix chip. As in Yeung et al. [13] [35] a subset of 384 genes whose expression levels peak at different time points corresponding to the five phases of cell cycle was used for our experiments. These data were obtained from supplementary web site provided by Yeung et al. at http://staff.washington.edu/kayee/cluster/raw_cellcycle_384_17.txt.

Prior to clustering, the data were normalized to have zero mean and unit variance. ADSOM was applied to cluster this dataset with different number of initial weights such as 9, 12, 15 and 20. As depicted in Figure 7.4, the final locations of position vectors converge into fewer groups in a two-dimensional space regardless of the initial number of nodes. We applied the tree-based validation index to estimate the optimal number of clusters from the final locations of the position vectors. As described before, the number of clusters at which the tree-based index reaches its highest peak can be used as an indicator of optimal number of clusters. The results in Figure 7.4 indicated that the optimal number of clusters is 5 regardless of different number of initial nodes. This result agrees with that of Yeung et al. [35] and was as expected from a dataset corresponding to 5 phases of the yeast cell cycle.

Table 7.2 shows the number of genes in each cluster as well as the number of common genes found when clustering the yeast cell cycle data with different number of initial nodes (N), i.e., N= 9, 12, 15 and 20. The percentage of common genes between

clusters formed by N=9 & 12 was 88%, N=9,12, & 15 was 79.7%; and N=9,12, 15, & 20 was 70.8%.

To compare the performance of ADSOM with the model-based clustering method, we run MCLUST for several models. The models are characterized by their covariance matrices such as equal volume spherical (EI), unequal volume and spherical (VI), ellipsoidal with equal volume, shape and orientation (EEE), ellipsoidal, varying volume, shape, and orientation (VVV), diagonal, varying volume, varying shape (VVI), etc. The Bayesian information criterion (BIC) introduced by Yeung et al. [35] was used to compare models with different numbers of clusters and different covariance matrix parameterization. A large BIC score indicates strong preference for the corresponding model. Figure 7.5 shows the BIC scores obtained by clustering the yeast cell cycle data for various cluster numbers ranging from 2 to 32 using five different models. As can be seen from the figure, the model-based approach favors the EEE model, which reached its maximum BIC score at 6 clusters.

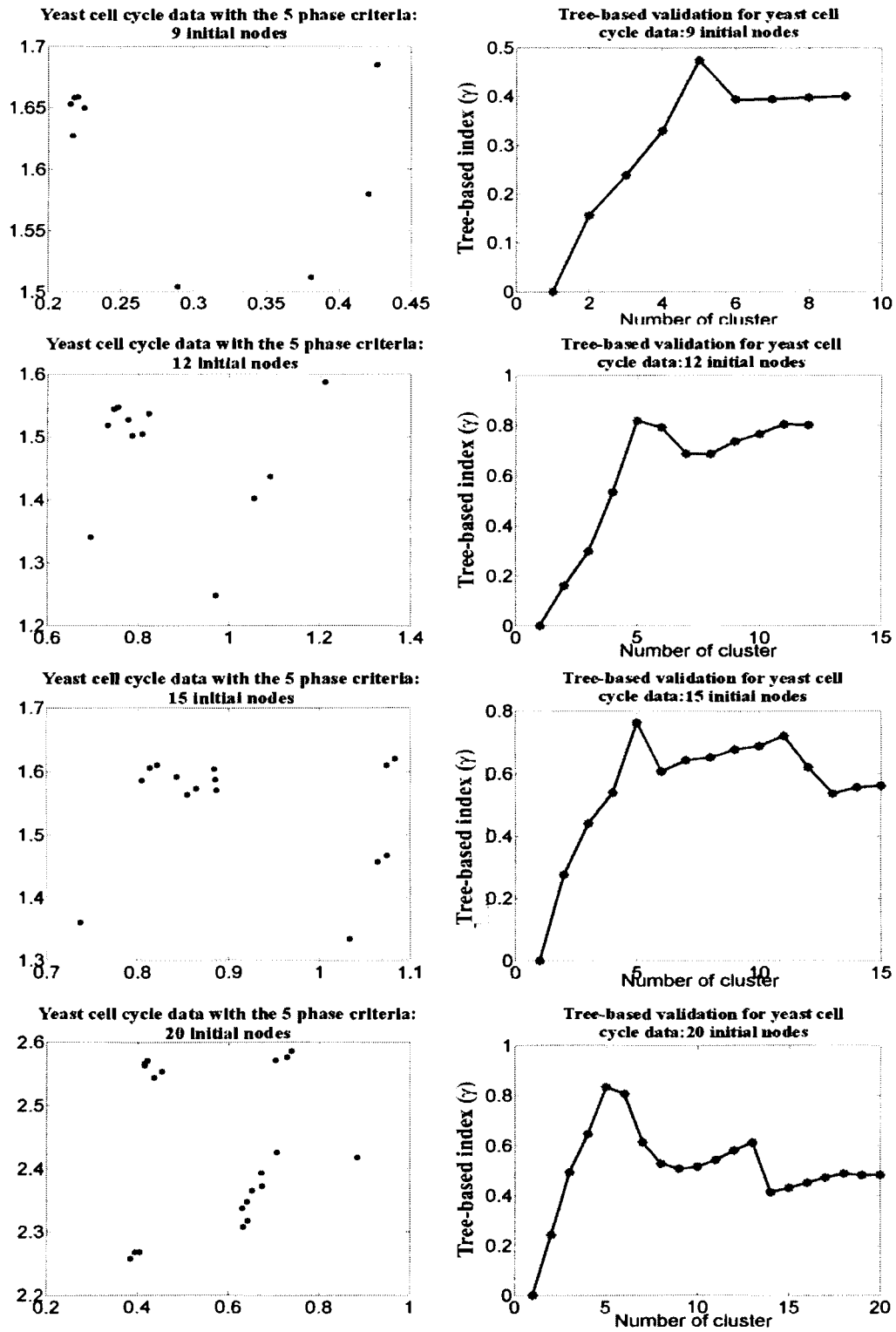


Figure 7.4: Final position vectors (left) after using ADSOM to cluster yeast cell cycle data with initial nodes 9, 12, 15, and 20 and the corresponding tree-based validation results (right).

Cluster #	I	II	III	IV	V
# of genes for *N*=9	43	89	77	107	68
# of genes for *N*=12	38	100	71	100	75
# of genes for *N*=15	40	89	73	116	66
# of genes for *N*=20	35	93	70	110	76
# of common genes for *N*=9 & 12	29	81	71	92	65
# of common genes for *N*=9, 12, &15	28	66	73	83	61
# of common genes for *N*=9, 12, 15, &20	24	55	70	76	53

Table 7.2: Number of genes as well as number of common genes for yeast cell cycle data using ADSOM with 9, 12, 15, and 20 initial nodes; *N*, number of initial nodes.

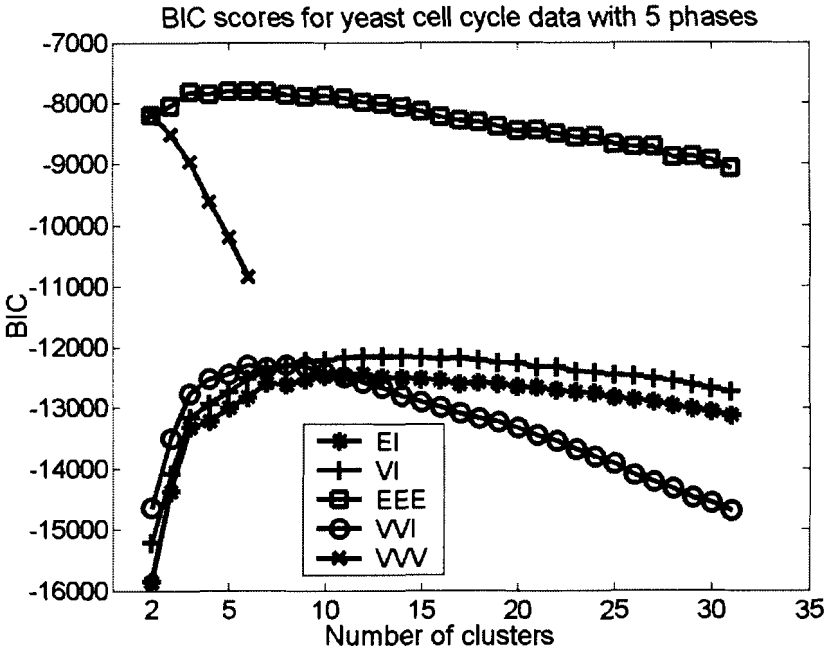


Figure 7.5: BIC scores for yeast cell cycle data with 5 phases.

7.3 Yeast Sporulation Data

Gene expression data in the budding yeast *Saccharomyces cerevisiae* from spotted c-DNA microarrays studied during the diauxic shift, the mitotic cell division cycle, sporulation, and temperature and reducing shocks were considered. In particular, the data corresponding to sporulation were used for this experiment. From the yeast sporulation data, 218x11 gene expression data that belong to four classes were extracted. Variance normalization was applied to the data. The four classes are ribosomal proteins, respiration, mitochodrial organization, and tricarboxylic-acid pathway. This functional classification was made using information from Munich Information Center for Protein Sequences (MIPS) [39].

ADSOM clustered this yeast data into its four distinct classes without having the knowledge of the actual number of clusters. The beauty of ADSOM algorithm is that it would converge to the actual number of clusters without having to go through tedious cluster validation techniques. To demonstrate this, two experiments were conducted with this dataset; once ADSOM was assigned 9 initial nodes and once 20 nodes. ADSOM's position vectors that were assigned to these initial nodes converged to four final clusters regardless of the initial number of nodes chosen. Figure 7.6 shows the location of final position vectors for 9 initial nodes (left), for 20 initial nodes (middle) and the results obtained after applying the tree-based validation index (right). Note that these final four clusters that represent the actual and true number of clusters in the data are shown by enclosed boundaries. It also shows which of the initial 9 or 20 nodes had to be grouped together in order to make the final four classes. As can be seen from the Figure 7.6

(right), the optimal number of clusters is 4 regardless of different number of initial nodes (9 and 20).

It was observed that ADSOM resulted in partitioning the data with 70% accuracy. This accuracy was calculated based on the biological classification made by MIPS. In gene expression data, one doesn't normally have a *priori* knowledge about the true number of classes, and what distinguishes ADSOM from the currently available techniques is that one doesn't need to have this knowledge to be able to find the true number of classes accurately.

To evaluate the consistency of the clustering, the number of common genes in two clusters for different number of initial nodes was investigated. Table 7.3 illustrates the number of genes obtained in the four clusters formed by ADSOM and the number of common genes between the cluster pair formed with 9 and 20 initial number of nodes, respectively. The robustness and consistency of ADSOM in clustering is evident from the large number of common genes (90.4%) in the clusters as shown below.

In a separate experiment, the performance of ADSOM for different cluster sizes was investigated. To accomplish this, yeast sporulation data from 2, 3, and 4 MIPS classes were extracted and clustered independently. As shown in figure 7.7, the number of clusters was correctly identified by ADSOM using 9 initial nodes.

Figure 7.8 shows the BIC scores obtained using MCLUST for four different models at clusters ranging from 1 to 20. As can be seen from the figure, the model-based approach favors the EEE model whose first maximum BIC score is reached at 10 and the second maximum BIC score is reached at 5 clusters. Note that the expected number of clusters for the yeast sporulation data is 4.

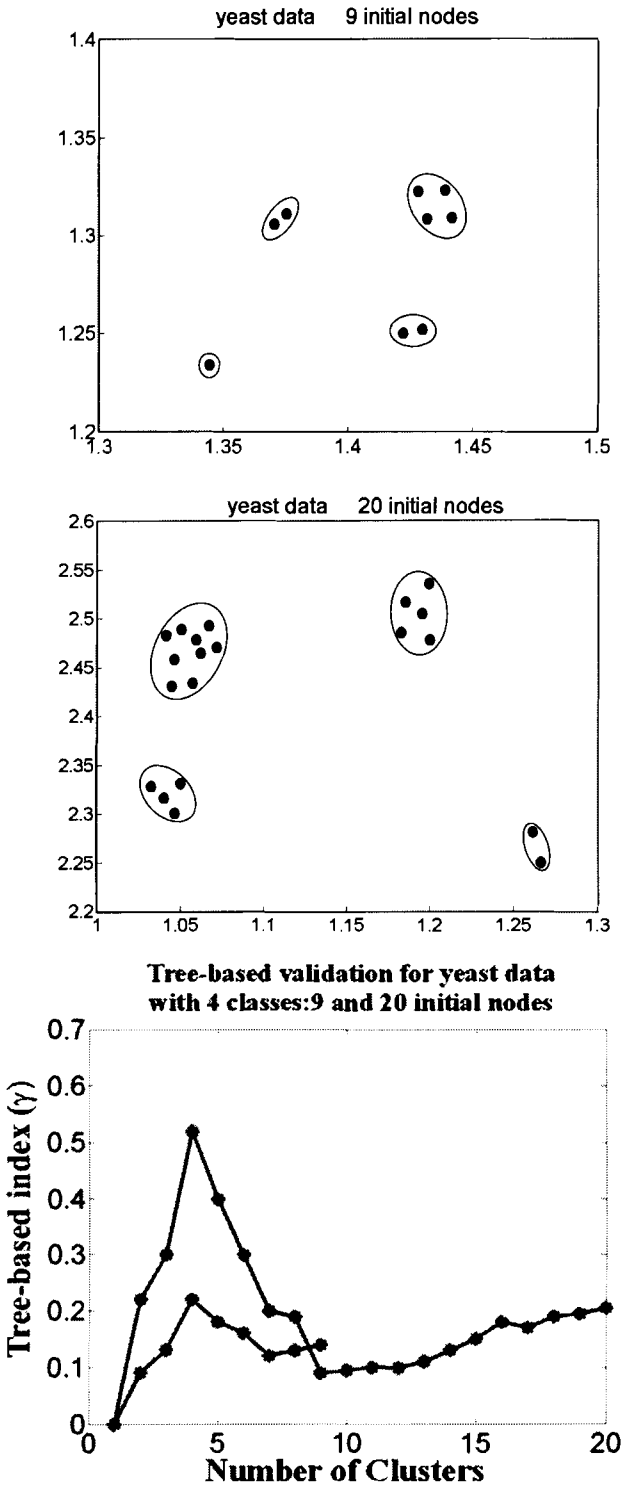


Figure 7.6: Final position vectors with initial nodes 9 (left), 20 (middle) and the corresponding tree-based validation results (right) using ADSOM for yeast sporulation data with 4 classes.

	Cluster I	Cluster II	Cluster III	Cluster IV
# of genes for N=9	60	28	19	111
# of genes for N=20	55	25	21	117
# of common genes for N=9 & 20	46	24	16	111

Table 7.3: Number of genes as well as number of common genes for yeast sporulation data using ADSOM with 9 and 20 initial nodes; N, initial number of nodes.

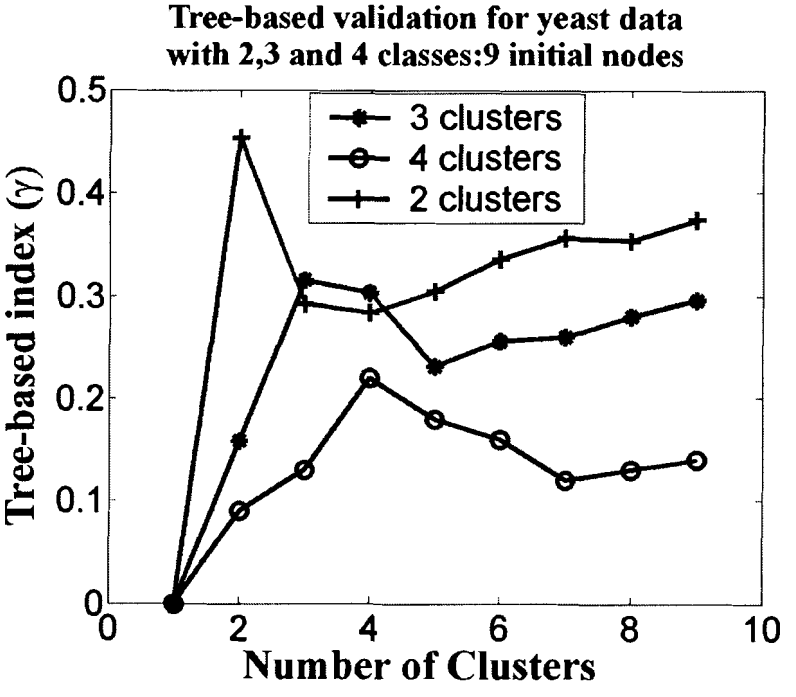


Figure 7.7: Results of tree-based validation for yeast sporulation data.

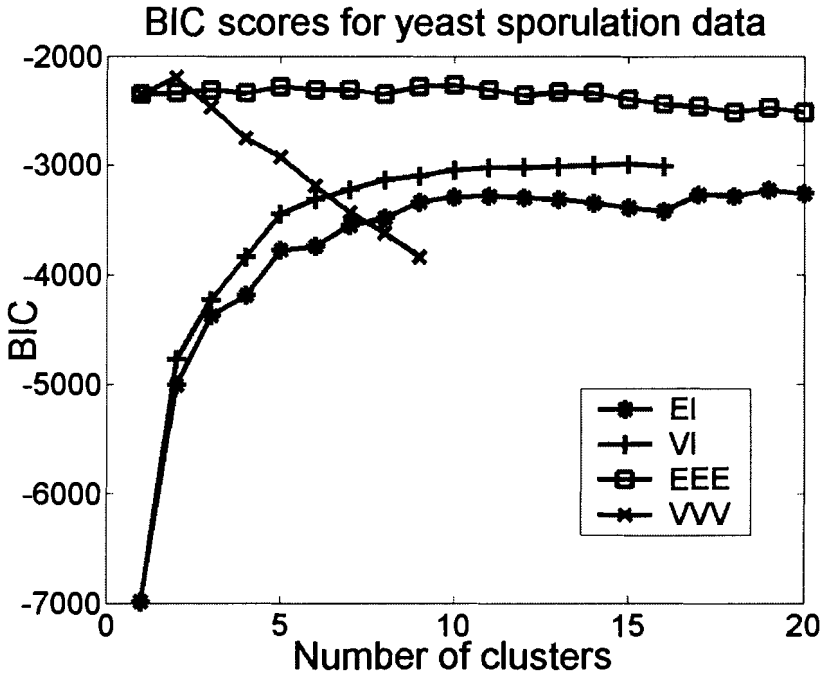


Figure 7.8: BIC scores for yeast sporulation data.

SOM was applied to group the yeast data in various numbers of clusters. The clusters are evaluated using FOM. Figure 7.9 reveals that that the optimal number of clusters is six.

Figure 7.10 depicts the Xie-Beni index calculated for various numbers of clusters after clustering the yeast data using fuzzy c-means. As shown in the figure, the index has its first minimum at four, confirming the existence of four clusters in the yeast data.

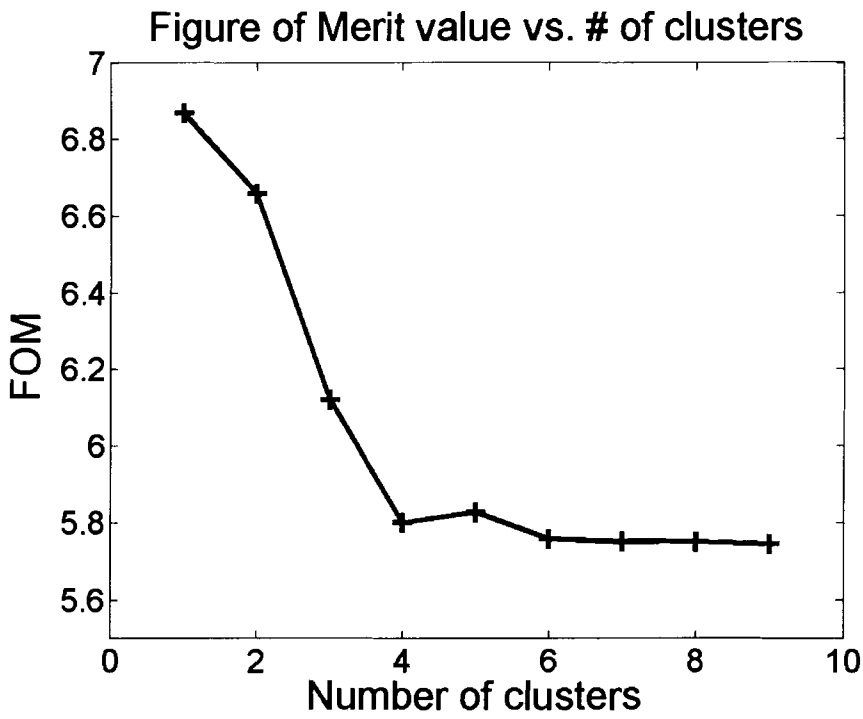


Figure 7.9: Clustering results using SOM for yeast data.

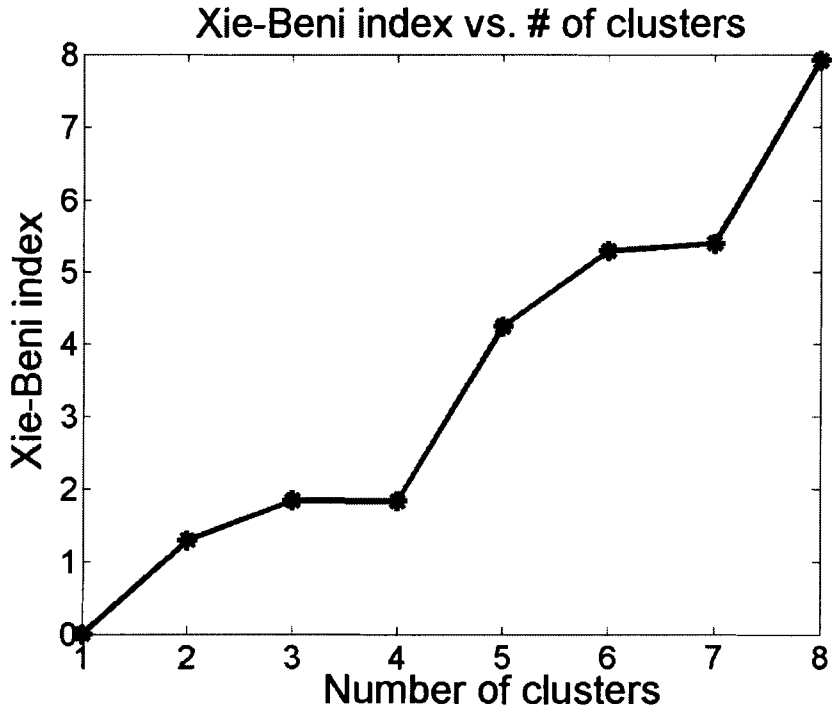


Figure 7.10: Clustering results using FCM for yeast data.

7.4 Yeast Cdc15 and Elu

The spotted c-DNA microarray data used for this experiment were those corresponding to cdc-15 and elu. Cdc15 data consisted of 15 time points following arrest of cdc15 temperature sensitive mutant and elu data contained 14 time points following elutriation. After removing genes with missing points, there were 1882 genes in cdc-15 data and 2020 genes in elu data. Before applying ADSOM, variance normalization was applied to the data. ADSOM was used to cluster each data set with different number of initial weights such as 25, 30, 36 and 42. The tree-based indices corresponding to the final locations of the position vectors were used to identify the number of clusters in both data sets. ADSOM gave fairly consistent number of clusters in the unknown yeast data set, thus demonstrating its effectiveness and reliability. As it can be seen from Table 7.4, the result of this experiment shows that there are 16 or 17 clusters available in both data sets.

Figure 7.11 and 7.12 shows the BIC scores obtained using MCLUST for five different models in clustering the yeast elu (Figure 7.11) and yeast cdc15 data (Figure 7.12). As can be seen from the figure, the maximum BIC score is reached at 21 clusters for the EEE model when clustering the yeast elu data. For the yeast cdc15 data, the EEE model reached its first, second, and third maximum BIC scores at 27, 21 and 19 clusters, respectively.

Data set	Number of data	Number of clusters obtained using ADSOM				
		Initial nodes =25	Initial nodes =30	Initial nodes =36	Initial nodes =42	Average
Yeast cdc15	1882x15	16	16	17	17	16.5
Yeast elu	2020x14	16	16	16	17	16.25

Table 7.4: Number of clusters obtained using ADSOM.

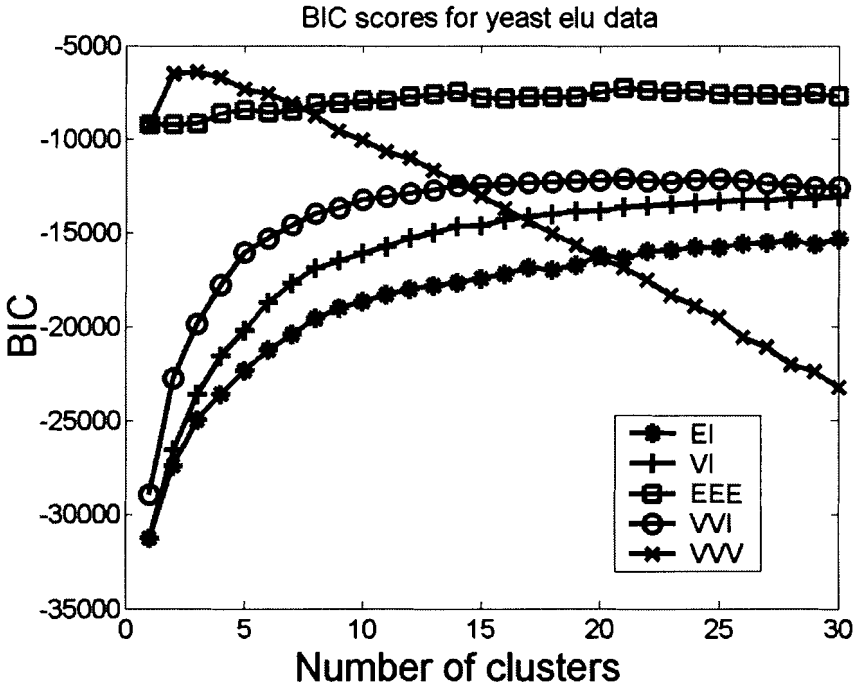


Figure 7.11: B7IC scores for yeast elu data

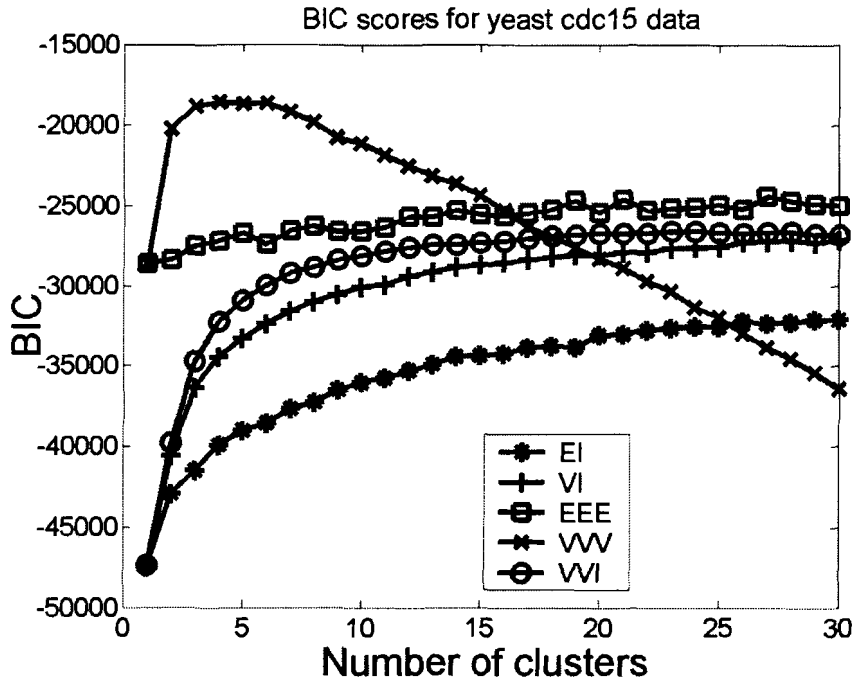


Figure 7.12: BIC scores for yeast cdc15 data.

In addition, ART and SOM were applied to cluster the yeast *cdc15*. It is observed that similar results were found using these two techniques heuristically. However, ART and SOM require the aid of an external validating methods. In order to find the accurate number of clusters from gene expression data, one usually studies internode distance as a function of the number of nodes. When the average internode distance does not decrease appreciably as more nodes are added, "overclustering" occurs. In other words, beyond this "saturation point," adding further nodes is not fruitful in differentiating the genes. For SOM, however, rather than measuring internode distance, each node's average gene is compared (in the Euclidean sense) to other nodes' average gene. Unfortunately, this measurement of "inter-average gene" distance does not probe the weights of the SOM. In fact, certain initial weights can create "kinks" that do not accurately cover the input space. These "kinks" can be better detected with internode distance than inter-average

gene distance. Biologically, internode distance can be thought of as the distance between the centers of two gene clusters.

With ART, a similar saturation point can be demonstrated by varying vigilance. For example, Figure 7.13 illustrates the number of nodes and their internode distance created by ART versus vigilance for the *cdc15* dataset. Thus both SOM's and ART can create different number of nodes for a given dataset. It is noted that as vigilance increases, the number of clusters increases while the internode distance decreases.

Many experiments using ART and SOM algorithms were conducted with *cdc15*. For ART, the vigilance parameter was varied from 0 to 0.7 in steps of 0.025. Vigilance of above 0.7 leads to more than 60 clusters. Results are the average of 5 runs. The other operational parameters of ART such as alpha and beta were fixed at 0.001 and 1 ("fast learning"); respectively; the Weber rule was used. $D'(\theta)$ was approximated from $D(\theta)$ to the first order. The simulation for each dataset took approximately between 7-10 minutes using a PC with Pentium III processor. From the 1% change criteria, as shown in Figure 7.16, a good estimate for the number of clusters in each dataset and the corresponding vigilance were found to be 16 and 0.525, respectively.

SOM was changed from 3 to 40 with 3 runs at each number of clusters to average the outcome. Conducting the experiment for each dataset approximately took about 10 hours on the same PC as compared to minutes for ART. It is noted that the larger the number of genes in a dataset is the more time consuming the processing is. Figure 7.14 shows the results of SOM for *cdc15* dataset. Again using the 1% criteria for finding the saturation level, the number of clusters is found to be 17, as was also the case for ART algorithm.

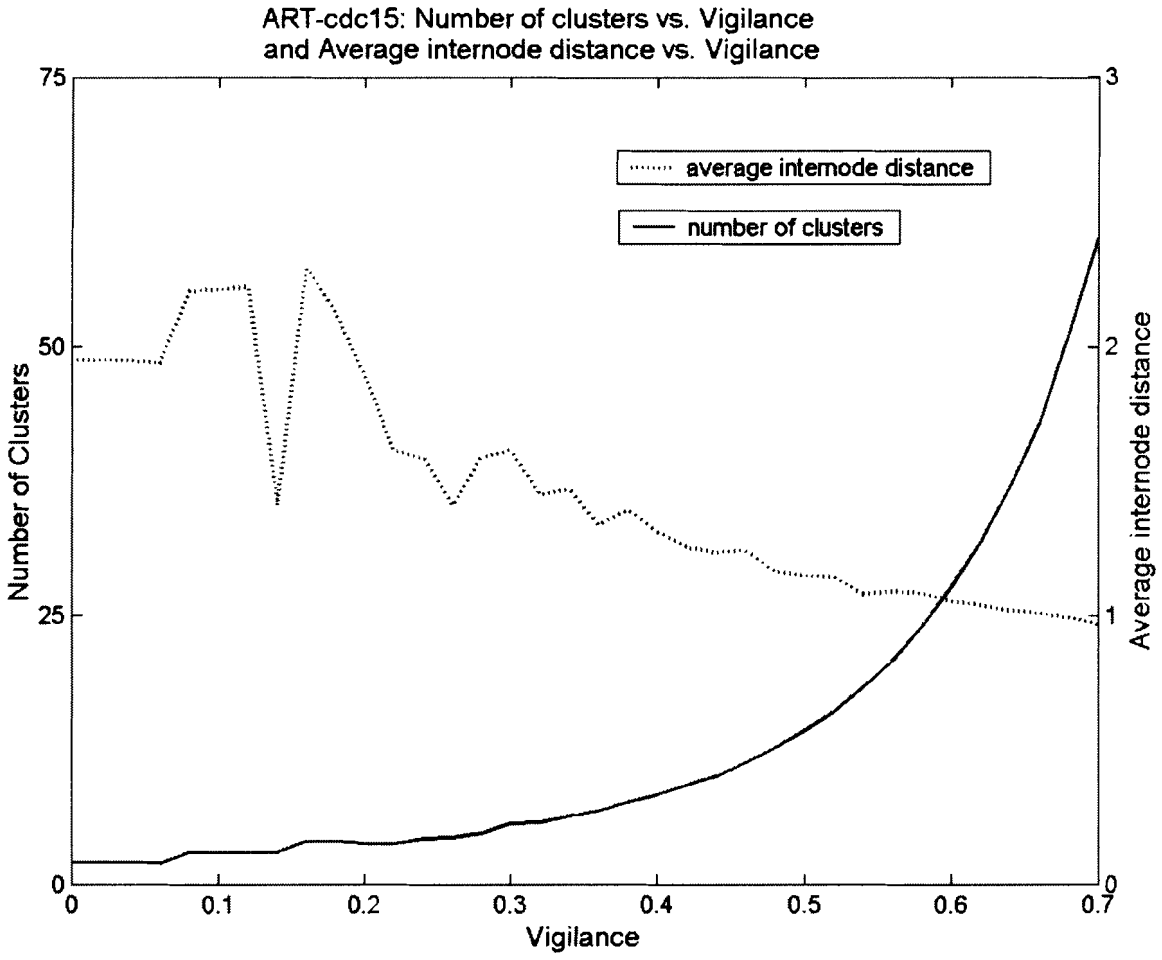


Figure 7.13: Number of clusters and average internode distance vs. vigilance for cdc15.

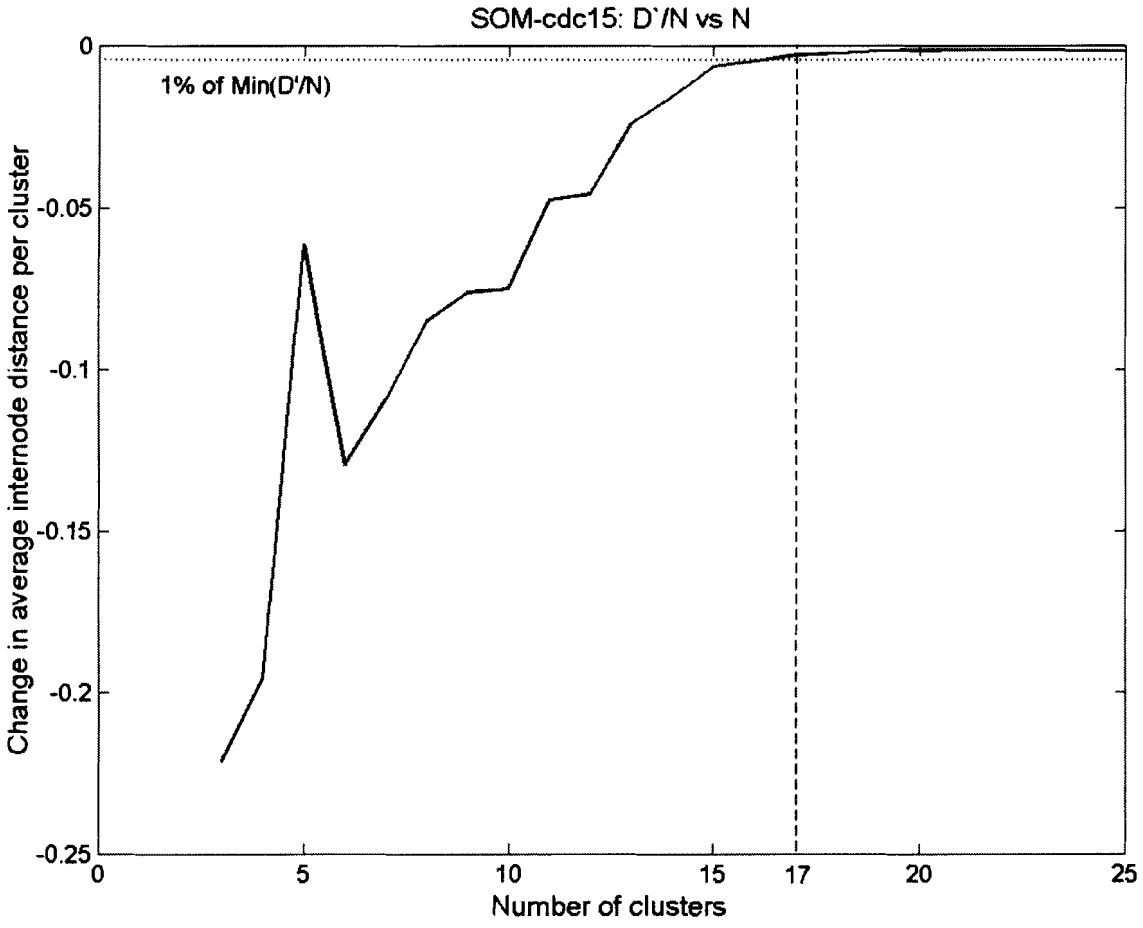


Figure 7.14: Change in internode distance per cluster (D'/N) vs. number of clusters (N).

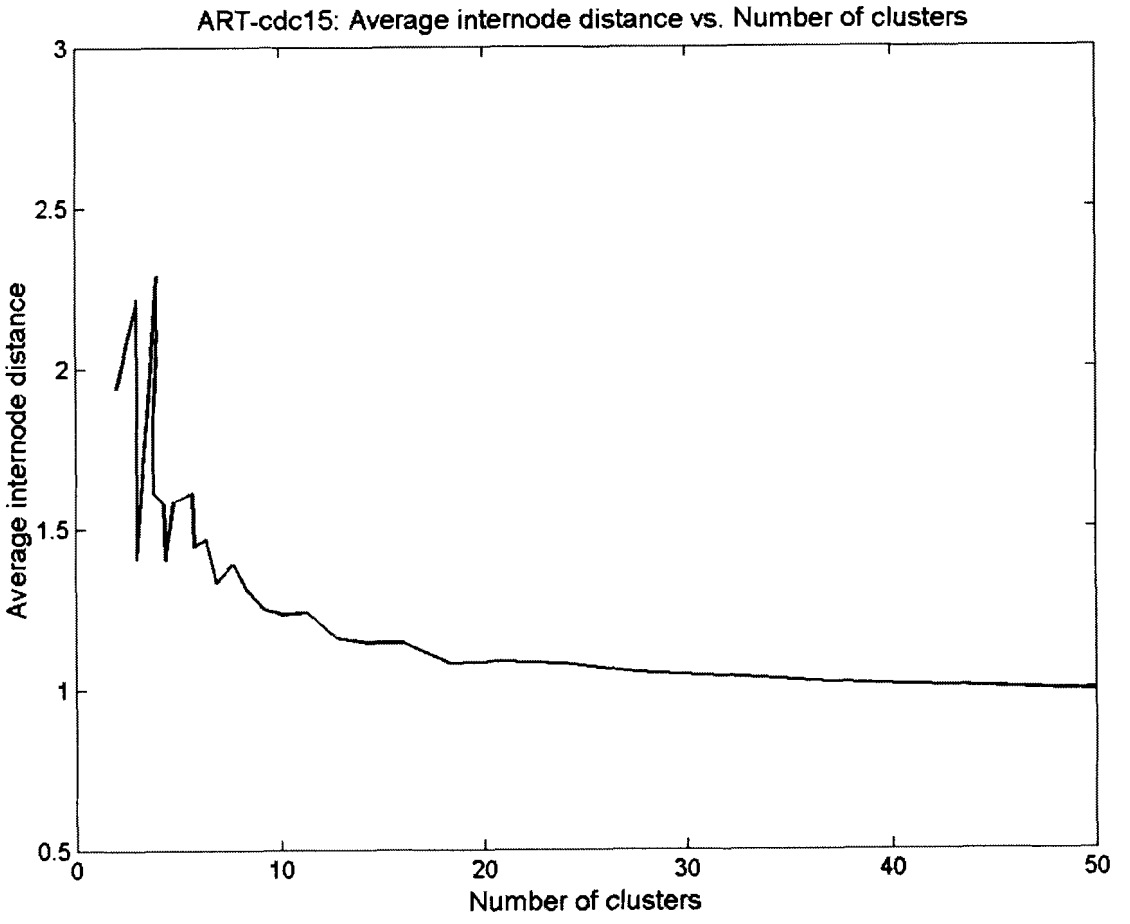


Figure 7.15: Internode distance vs. the number of nodes for cdc15.

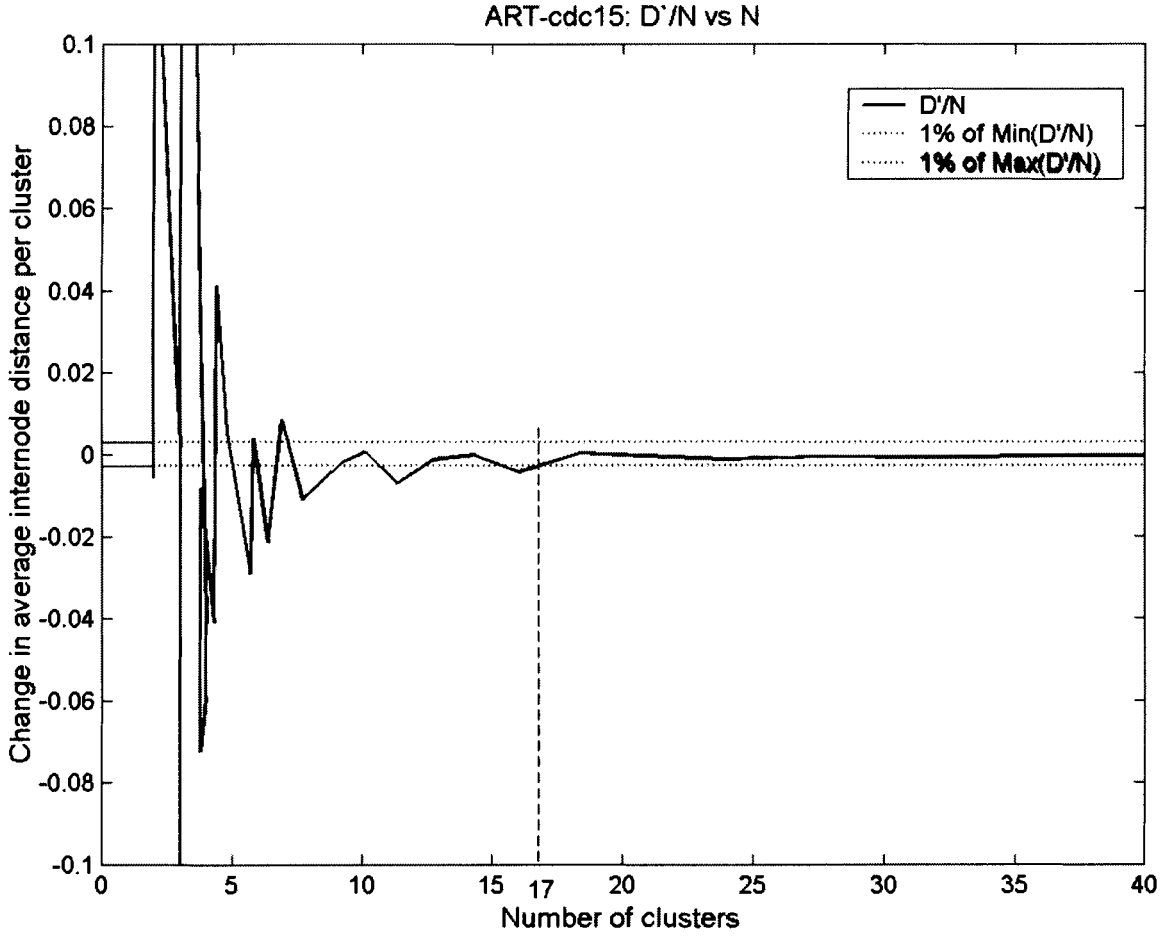
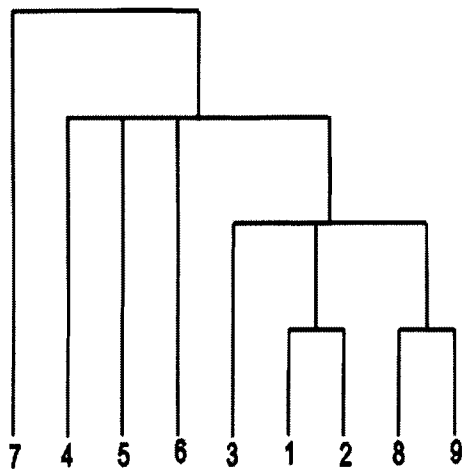


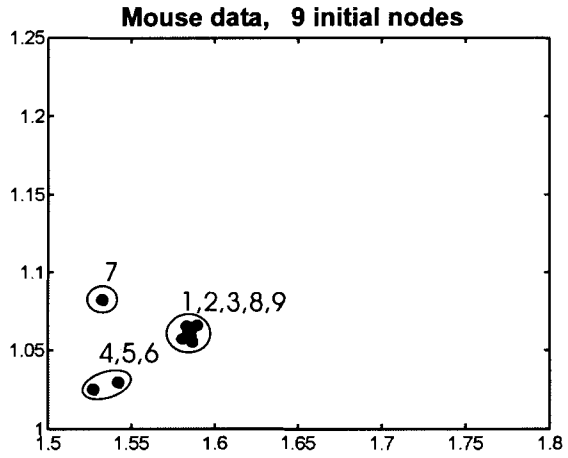
Figure 7.16: Change in average internode distance (D') per cluster (N) vs. number of clusters (N).

7.5 UNC 9 Mouse Tumor Data

Mouse data that contained 9 tumor samples from different mouse strains were extracted for The Jackson Laboratory (http://www.jax.org/staff/churchill/labsite/datasets/expression/tumors/index.html). The 9 independent tumor samples were assayed with 18 cDNA microarrays using dye-swap reference design. There were 15488 rows of genes and 36 columns representing the samples. The data after filtering (3-fold) and range normalization (i.e., between 0 and 1) had 12866 genes. We used the proposed ADSOM to cluster these tumor samples. As shown in Figure 7.17 (middle), ADSOM identified three distinct clusters. The first cluster contained samples 1, 2, 3, 8, and 9; second cluster contained samples 4, 5, and 6; third cluster contained sample 7. As shown in Figure 7.17 (right), the tree-based index cluster validation result also identifies three clusters. Interpreting the results, it was observed that the first cluster contained all mammary tumor tissue samples; second cluster contained all normal mammary tissue samples; and the third cluster contained Waptag liver control samples. An interesting observation made is that tumors tend to cluster together than different tissues of the same strain.

ADSOM's results were compared with hierarchical clustering in partitioning these tumor samples. MA-ANOVA [40] was used to perform hierarchical clustering. The results obtained using the hierarchical method is shown in Figure 7.17 (left). Three clusters were identified which were similar to those obtained using ADSOM. However, interpretation of the number of clusters is user dependent. This can be a problem when there are a large number of clusters.





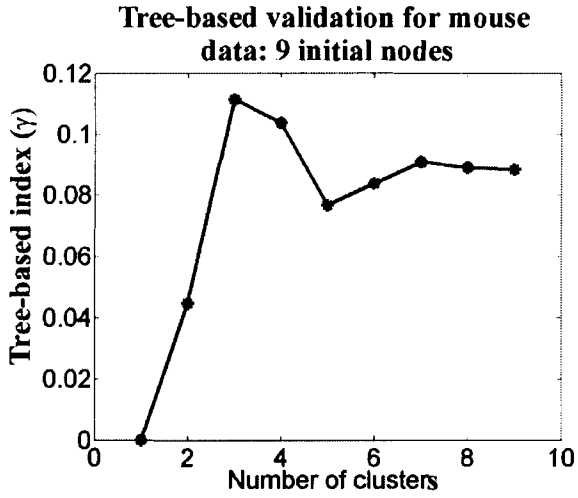


Figure 7.17: Results obtained using hierarchical clustering (left) and ADSOM visual (middle) ADSOM tree-based validation (right) for UNC 9 tumor data.

7.6 Human Fibroblast Data

A subset of the gene expression data provided by Iyer et al. [41] was used for this experiment. The full data shows the response of human fibroblast to serum, using cDNA microarrays representing about 8600 distinct human genes to observe the temporal program of transcription that underlies this response. DNA microarray hybridization was used to measure the temporal changes in mRNA levels of 8613 human genes at 12 times, ranging from 15 minutes to 24 hours after serum simulation. The subset of 517 genes that were chosen on the basis of substantial change in the expression by Iyer et al. was considered. The data containing normalized R/G ratios was obtained from http://genome-www.stanford.edu/serum/data.html.

ADSOM was applied to cluster this data set with different initial nodes such as 16, 20, 25, and 30. Repeated trials resulted in either 10 or 11 final clusters. The location of the final position vectors and the results of tree-based index for 16 and 20 initial nodes are shown in Figure 7.18, respectively. Figure 7.19 depicts the BIC scores for five different models using MCLUST for the human fibroblast data. As can be seen from the figure, model-based clustering favors the EEE model, whose maximum BIC score is reached at 11 clusters. Note that ADSOM also detected 11 clusters in most cases. Similar number of clusters was obtained using hierarchical clustering by Iyer et al. Interpretation of the correct number of clusters using hierarchical clustering is, however, user dependent.

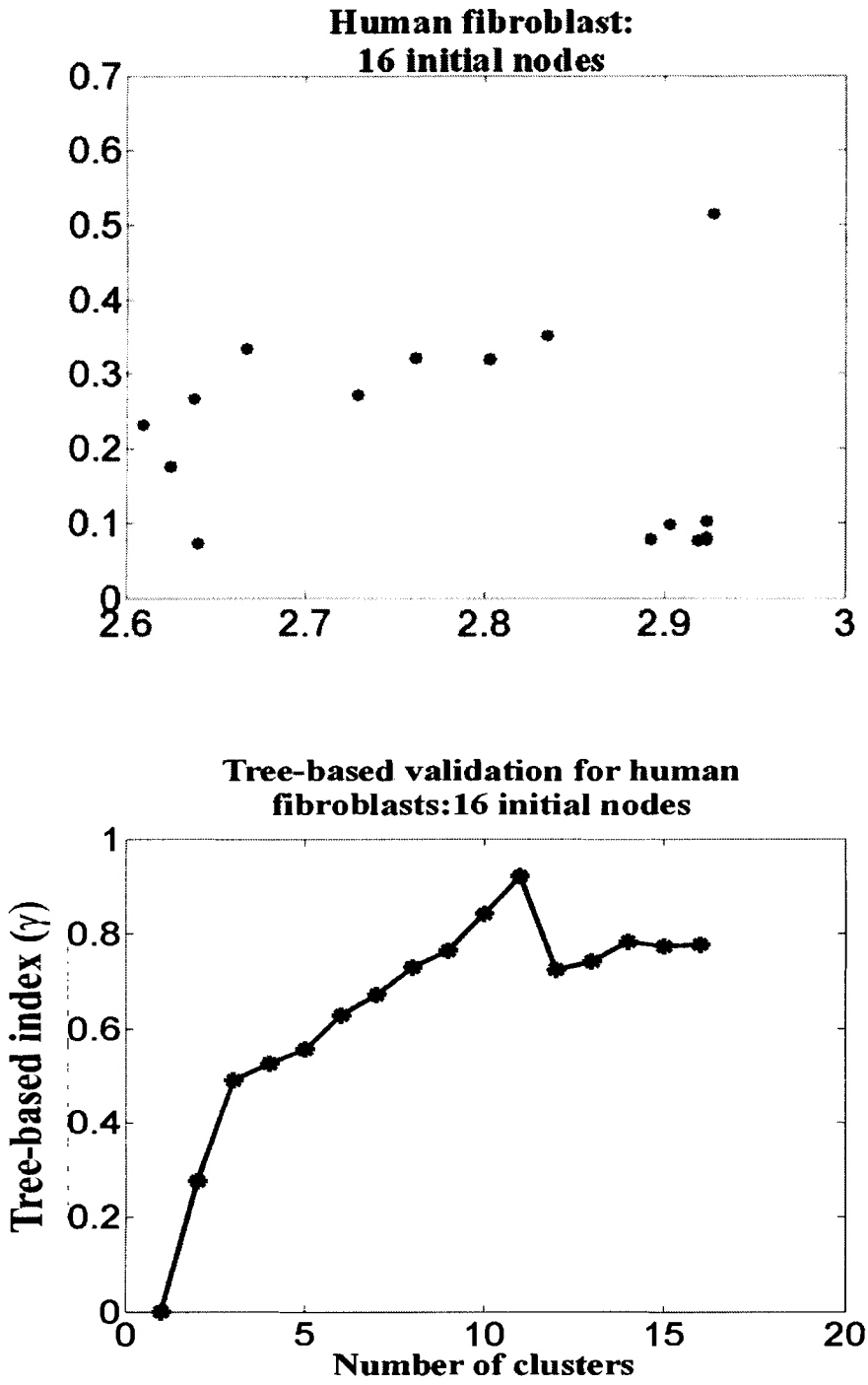


Figure 7.18: Final position vectors after using ADSOM to cluster human fibroblasts data with 16 initial nodes and 20 initial nodes and corresponding tree-based validation results.

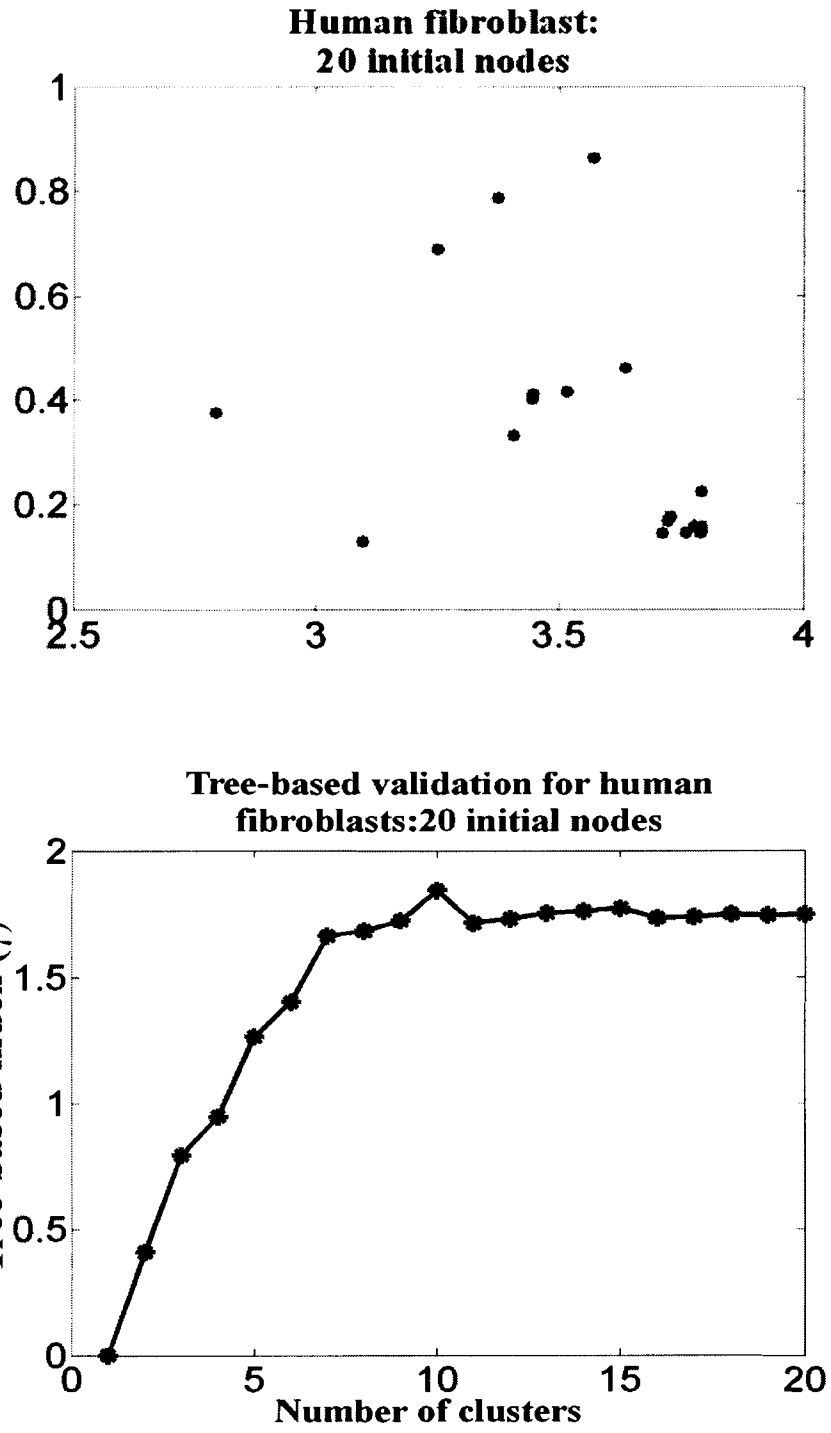


Figure 7.18: Continued.

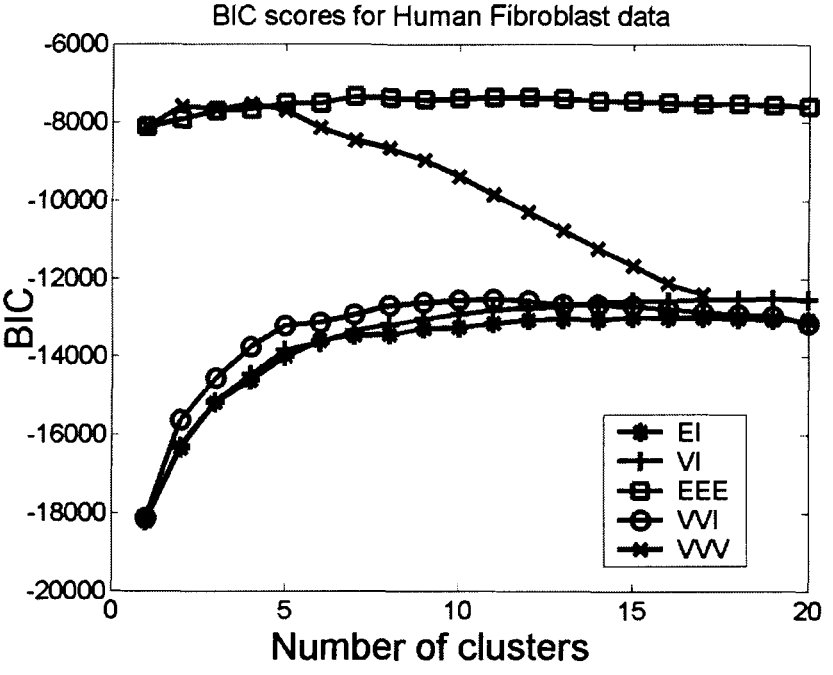


Figure 7.19: BIC scores for the human fibroblast data.

7.7 Escherichia Coli Data

E-coli data published by Tao et al. [22] were used in this experiment. The data originated from E-coli MG1655 cultures grown under different conditions on

(i) minimal medium containing 0.2% glucose

(ii) rich medium with luria broth containing 0.2% glucose

(iii) gluconate medium

The data were determined by the Pnaorama E-coli Gene Arrays using hybridization of mRNA isolated from E-coli cells grown under different conditions with the ORF specific DAN fragments immobilized on the array followed by radioactivity detection and image analysis. 67% of 4900 genes in 21 functional groups are selected. After removing the data with missing points, 2768 genes are used for clustering. The expected "correct" number of clusters was 21.

ADSOM was used to cluster this data set with different number of initial weights such as 25, 30 and 36. As depicted before, the final locations of position vectors converged into fewer groups in a two-dimensional space regardless of the initial number of nodes. We applied the tree-based validation index to estimate the optimal number of clusters from the final locations of the position vectors. As described before, the number of clusters at which the tree-based index reaches its highest peak can be used as an indicator of optimal number of clusters. The results in Figure 7.20 indicated that the optimal number of clusters is 21 regardless of different number of initial nodes. This experimental result showed that ADSOM gave fairly consistent number of clusters in this gene expression data set, thus demonstrating its effectiveness and reliability.

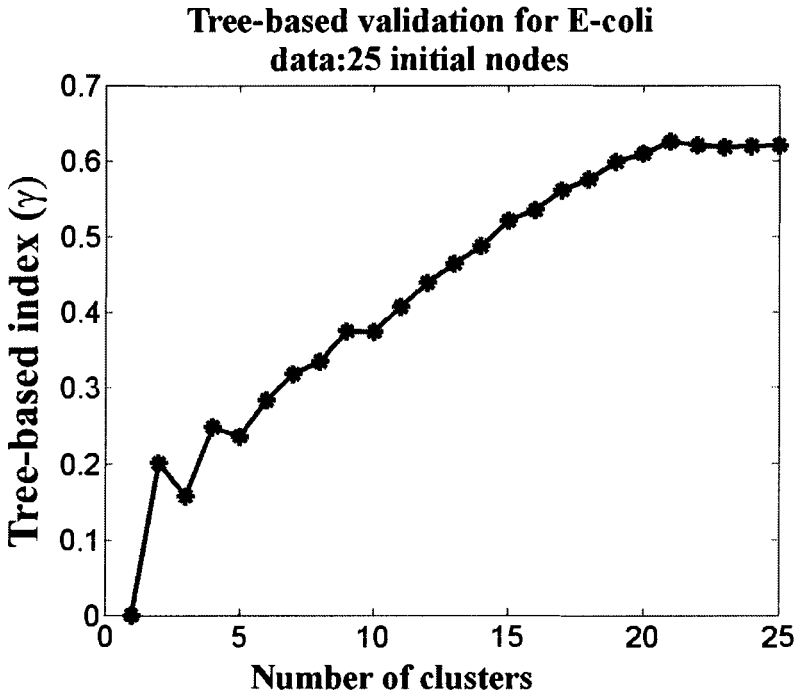


Figure. 7.20: Clustering results using ADSOM for E-coli data.

We applied MCLUST to cluster the E-coli data with four different models as described in section 2.2: EI, VI, EEE, VVV. A given number of clusters is specified as 30 and then the model parameters are estimated by the EM algorithm. The results are shown in Figure 7.21.

In Figure 7.21, for each model, the BICs for different numbers of clusters are plotted. As described before, the large BIC indicate strong evidence for the "correct" number of clusters. So for each model, we can identify the optimal number of clusters by identifying the peak of curves. In our experiments, only the third curve peaked at number of clusters as 21. This curve represents the model EEE. This result means that only EEE model can find out the "correct" number of clusters through model-based clustering.

The data were clustered using SOM for various numbers of clusters. As shown in Figure 7.22, the FOM curves started to saturate at 21, detecting the actual number of

clusters. Figure 7.23 depicts the Xie-Beni index calculated for various numbers of clusters after clustering the E-coli data using fuzzy c-means. As shown in the figure, the index didn't provide a conclusive result.

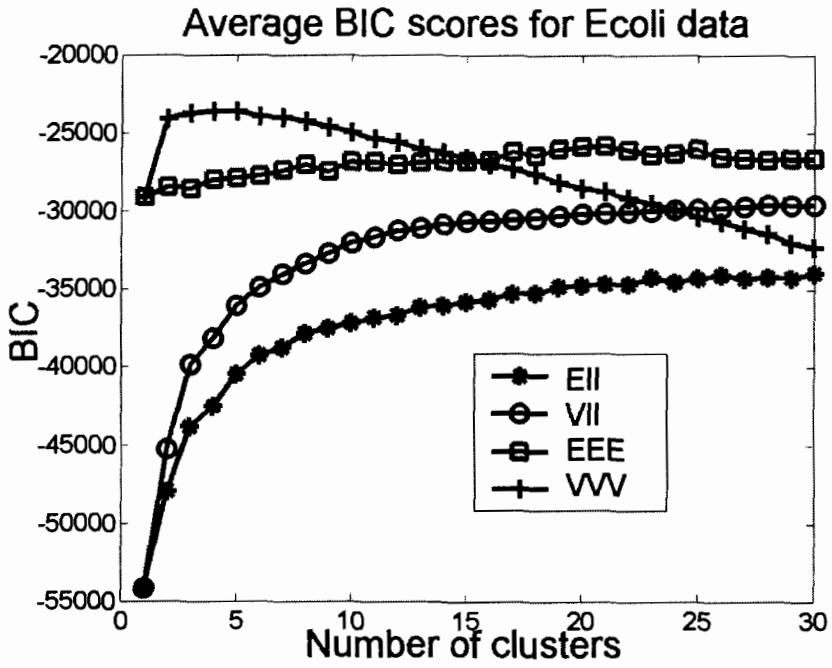


Figure 7.21: Clustering results using model-based method for E-coli data.

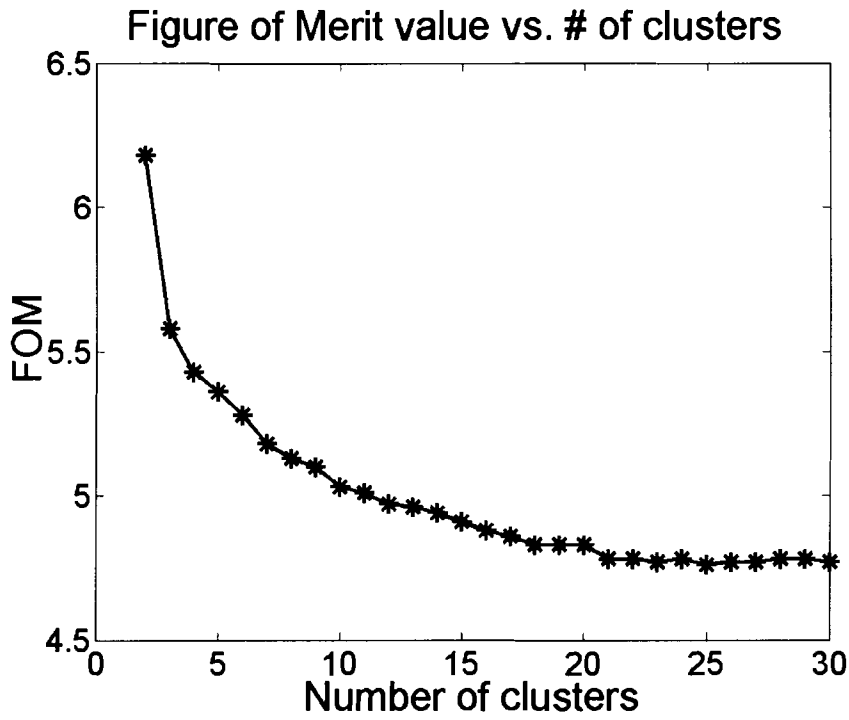


Figure 7.22: Clustering results using SOM for E-coli data.

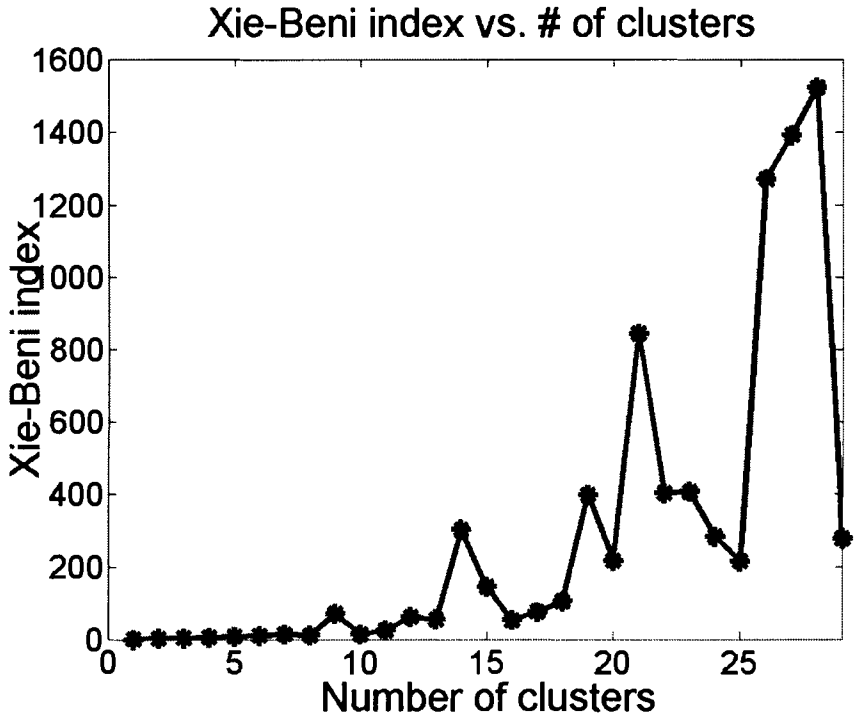


Figure 7.23: Clustering results using FCM for E-coli data.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 Conclusion

This thesis introduces a novel clustering tool known as adaptive double self-organizing map (ADSOM). It is shown that ADSOM gives a consistent final topology regardless of the initial topology. This topological flexibility of ADSOM provides credible information about the unknown number of clusters in gene expression data. ADSOM accomplishes both cluster visualization and data partitioning simultaneously without compromising the clustering accuracy. This particular feature of ADSOM is achieved by adapting its free parameters during the training process. In addition, in combination with hierarchical clustering, a hierarchical tree-based index is introduced for accomplishing cluster validation reliably.

Using model-based clustering, one can identify the "correct" number of clusters if the model of the data is known. Or one can find out the "good" model of the data if the number of clusters is known. However, if neither of the information is known, it is difficult to identify the "correct" number of clusters or identify the "good" model of the data. Other clustering methods such as ART and fuzzy c-means cannot identify the number of clusters without the aid of validating process, which is very time-consuming.

In summary, the methods described in this thesis offer two main advantages. First, ADSOM converges to a consistent number of groups regardless of the initial number of nodes. This is demonstrated by testing ADSOM on artificial data as well as real-world gene expression data. Second, ADSOM can visually present the clusters while

clustering the data. With the help of a tree-based index the number of clusters is easily identified. The approaches eliminate the trial-and-error process as well as the heuristic validating process, thus saving significant computational time.

8.2 Future Work

This thesis offers solutions to some of the challenges existing in gene expression analysis such as identifying and validating number of clusters. However, there are still several other issues in gene expression data analysis that need future investigations.

One of the future investigations may deal with the use of the clustering results obtained in this thesis to identify outliers in gene expression data. If gene expression data are clustered using ADSOM and one cluster contains very few gene expression profiles relative to other clusters, the profiles may be identified as outliers. A removal of such profiles should not affect the cluster information in other groups. In classification problems, removal of outliers can improve classification accuracy.

Furthermore, one should investigate suitable normalization methods and similarity measures. It is clear that normalization is a crucial step in pre-processing gene expression data. For the same raw data, different normalization methods may yield different clustering results. The selection of appropriate similarity measure for a normalized gene expression data set is still an unsolved issue.

ADSOM itself has some shortcomings that call for future investigations. As mentioned before, only one end of the boundary that defines the ratio between s_x and s_p was established in this thesis, see Equation (5.12). As a result, ADSOM fails to provide

satisfactory results in some cases. To alleviate this problem, one needs to carefully choose parameters m and n that are implemented in Equation (5.12) to address the other end of the boundary. Future work should focus on updating m and n together with ADSOM's weights and position vectors instead of keeping them as constant positive values through trial and error. Moreover the theory behind the proposed hierarchical tree-based index needs to be proven mathematically, although the approach has provided very good experimental results.

REFERENCES

[1] Schena M, Shalon D, Davis RW, and Brown PO. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270 (5235): 467-470, 1995.

[2] Eisen MB, Spellman PT, Brown PO, and Botstein D. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, 95: 14863-14868, 1998.

[3] Musavi MT, Wang D, and Chelian S. Gene expression data clustering using Adaptive Resonance Theory. *Proceedings of the International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences* (METMBS), 230-235, 2002.

[4] Tamayo P, Slonim D, Mesirov J, Zhu Q, Kitareewan S, Dmitrovsky E, Lander ES, and Golub TR. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA*, 96: 2907-2912, 1999.

[5] Tavazoie S, Hughes JD, Campbell MJ, Cho RJ, and Church GM. Systematic determination of genetic network architecture. *Nature Genetics*, 22: 281–285, 1999.

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

[7] Hartuv E, Schmitt A, Lange J, Meirer-Ewert S, Lehrach H, and Shamir R. An algorithm for clustering cDNAs for gene expression analysis. *Proceedings of the Third Annual International Conference on Computational Molecular Biology* (RECOMB99), Lyon, France: 1999, ACM Press, New York, 188-197, 1999.

[8] Tomida S, Hanai T, Honda H, and Kobayashi T. Gene expression analysis using fuzzy ART. *Genome Informatics*, 12: 245-246, 2001.

[9] Guthke R, Schmidt-Heck W, Hahn D, and Pfaff M. Gene expression data mining for functional genomics. *Proceedings of European Symposium on Intelligent Techniques* (*ESIT 2000*), Aachen, Germany, 170-177, 2000.

[10] Granzow M, Berrar D, Dubitzky W, Shuster A, Azuaje FJ, and Eils R. Tumor classification by gene expression profiling: comparison and validation of five clustering methods, SIGBIO Spatial Interest Group on Biomedical Computing of the ACM, ACM Press, New York, April 2001, vol. 21, no. 1, 16-22.

[11] Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, and Futcher B. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Molecular Biology of the Cell*, 9: 3273–3297, 1998.

[12] Mangiameli P, Chen SK, and West D. A comparison of SOM neural network and hierarchical clustering methods. *European Journal of Operational Research*. 93: 402-417, 1996.

[13] Yueng KY, Haynor DR, and Ruzzo WL. Validating clustering for gene expression data. *Bioinformatics* 17: 309-318, 2001.

[14] Tibshirani R, Walther G, and Hastie T. Estimating the number of clusters in a dataset via the gap statistic. Technical report 208, Department of Statistics, Stanford University, 2000.

[15] Jain AK and Dubes RC. *Algorithms for clustering data.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[16] Hubert L and Arabie P. Comparing partitions. *Journal of Classification* 2: 193–218, 1985.

[17] Lubovac Z, Olsson B, Jonsson P, Laurio K, and Andersson ML. Biological and statistical evaluation of gene expression profiles. *Proceedings of Mathematics and Computers in Biology and Chemistry*, 149-155, 2001.

[18] Fraley C and Raftery AE. MCLUST: Software for Model-Based Clustering, Discriminant Analysis and Density Estimation, Technical Report 415, Department of Statistics, University of Washington, October 2002.

[19] Fraley C and Raftery AE. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association* 97: 611-631, 2002.

[20] Ramoni M, Sebastiani P, and Kohane L. Cluster Analysis of Gene Expression Dynamics. *Proc. Natl. Acad. Sci. USA*, 99: 9121-9126, 2002.

[21] Kaski S, Nikkilä J, and Kohonen T. Methods for interpreting a self-organized map in data analysis. *Proceedings of the 6th European Symposium on Artificial Neural Networks (ESANN'98), Bruges, Belgium, April 22-24*, D-Facto, Brussels, Belgium, 1998, 185-190.

[22] Kaski S, Nikkilä J, and Kohonen T. Methods for exploratory cluster analysis. *Proceedings of SSGRR 2000, International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, L'Aquila, July 31--August 6.* Scuola Superiore G. Reiss Romoli, 2000. (Proceedings on CD-ROM, ISBN 88-85280-52-8).

[23] Ultsch A and Siemon H. Kohonen's self-organizing maps for exploratory data analysis. *Proceedings of Int. Neural Network Conf. (INNC'90)*, Dordrecht, Netherlands, Kluwer, 1990, 305-308.

[24] Nikkilä J, Törönen P, Kaski S, Venna J, Castrén E, and Wong G. Analysis and visualization of gene expression data using Self-Organizing Maps, *Neural Networks, Special issue on New Developments on Self-Organizing Maps, 2002.* vol. 15, no. 8-9, 953-966.

[25] Kaski S. SOM-based exploratory analysis of gene expression data. In: *Advances in Self-Organizing Maps*, Allinson N, Yin H, Allinson L, and Slack J (editors), 124-131. Springer, London, 2001.

[26] Herrero J, Valencia A, and Dopazo J. A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics*, 17: 126-136, 2001.

[27] Herrero J and Dopazo J. Combining hierarchical clustering and self-organizing maps for exploratory analysis of gene expression patterns. *Journal of Proteome Research*, 1(5): 467-470, 2002.

[28] Su M and Chang H. A new model of self-organizing neural networks and its application in data projection. *IEEE Transactions on Neural Networks*, 12: 153-158, 2001.

[29] Kerr MK, Martin M., and Churchill GA. Analysis of variance for gene expression microarray data. *Journal of Computational Biology*, 7: 819-837, 2000.

[30] D'haeseleer P, Liang S, and Somogyi R. Genetic Network Inference: From Co-Expression Clustering to Reverse Engineering, *Bioinformatics* 16(8): 707-726, 2000.

[31] Brown PO and Botstein D. Exploring the New World of the genome with DNA microarrays. *Nature Genetics*, 21: 33-37, 1999.

[32] Bassett DE, Eisen MB and Boguski MS. Gene Expression Informatics--It's All in Your Mine. *Nature Genetics*, 21: 51-55, 1999.

[33] Su M, Liu T, and Chang H. An efficient initialization scheme for the self-organizing feature map algorithm. *Proceedings of IEEE International Joint Conference on Neural Networks*, 1906-1910, Washington DC, 1999.

[34] Carpenter G. and Grossberg S. A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine. *Computer Vision, Graphics, and Image Processing*, 37: 54-115, 1983.

[35] Yeung KY, Fraley C, Murua A, Raftery AE, and Ruzzo WL. Model-based Clustering and Data Transformations for Gene Expression Data. *Bioinformatics*m 17: 977-987, 2001.

[36] Beni G and Xie L. A Validity Measure for Fuzzy Clustering, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8: 841-847, 1991.

[37] Cho RJ, Campbell MJ, Winzeler EA, Steinmetz L, Con-way A, Wodicka L, Wolfsberg TG, Gabrielian AE, Lands-man D, Lockhart DJ, and Davis RW. A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell*, 2: 65-73, 1998.

[38] De Smet F, Mathys J, Marchal K, Thijs G, De Moor B, and Moreau Y. Adaptive Quality-based clustering of gene expression profiles. *Bioinformatics*, 18: 735-746, 2002.

[39] Mewes HW, Albermann K, Heumann K, Liebl S, and Pfeiffer F. MIPS: A database for protein sequences, homology data and yeast genome information. *Nucleic Acid Research*, 25: 28-30, 1997.

[40] Wu H, Kerr MK, Xiangqin C, and Churchill GA. MAANOVA: A Software Package for the Analysis of Spotted cDNA Microarray Experiments. In: *The Analysis of Gene Expression Data: Methods and Software*, Parmigiani G, Garrett ES, Irizarry RA, and Zeger SL (editors), Springer, 2003.

[41] Iyer VR, Eisen MB, Ross DT, Schuler G, Moore T, Lee JFC, Trent JM, Staudt LM, Hudson J, Boguski MS, Lashkari D, Shalon D, Botstein D, and Brown PO. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283: 83-87, 1999.

[42] Banfield JD and Raftery AE. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49: 803–821, 1993.