PARTITION CLUSTERING OF HIGH DIMENSIONAL LOW SAMPLE SIZE DATA BASED ON P-VALUES

by

GEORGE FREITAS VON BORRIES

B.A., Universidade de Brasília, Brazil, 1990
M.S., Texas A&M University, USA, 2003

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Statistics
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2008

Abstract

This thesis introduces a new partitioning algorithm to cluster variables in high dimensional low sample size (HDLSS) data and high dimensional longitudinal low sample size (HDLLSS) data. HDLSS data contain a large number of variables with small number of replications per variable, and HDLLSS data refer to HDLSS data observed over time.

Clustering technique plays an important role in analyzing high dimensional low sample size data as is seen commonly in microarray experiment, mass spectrometry data, pattern recognition. Most current clustering algorithms for HDLSS and HDLLSS data are adaptations from traditional multivariate analysis, where the number of variables is not high and sample sizes are relatively large. Current algorithms show poor performance when applied to high dimensional data, especially in small sample size cases. In addition, available algorithms often exhibit poor clustering accuracy and stability for non-normal data. Simulations show that traditional clustering algorithms used in high dimensional data are not robust to monotone transformations.

The proposed clustering algorithm PPCLUST is a powerful tool for clustering HDLSS data, which uses p-values from nonparametric rank tests of homogeneous distribution as a measure of similarity between groups of variables. Inherited from the robustness of rank procedure, the new algorithm is robust to outliers and invariant to monotone transformations of data. PPCLUSTEL is an extension of PPCLUST for clustering of HDLLSS data. A nonparametric test of no simple effect of group is developed and the p-value from the test is used as a measure of similarity between groups of variables.

PPCLUST and PPCLUSTEL are able to cluster a large number of variables in the presence of very few replications and in case of PPCLUSTEL, the algorithm require neither a large number nor equally spaced time points. PPCLUST and PPCLUSTEL do not suffer from loss of power due to distributional assumptions, general multiple comparison problems and difficulty in controlling heterocedastic variances. Applications with available data from previous microarray studies show promising results and simulations studies reveal that the algorithm outperforms a series of benchmark algorithms applied to HDLSS data exhibiting high clustering accuracy and stability.

PARTITION CLUSTERING OF HIGH DIMENSIONAL LOW SAMPLE SIZE DATA BASED ON P-VALUES

by

GEORGE FREITAS VON BORRIES

B.A., Universidade de Brasília, Brazil, 1990
M.S., Texas A&M University, USA, 2003

A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Statistics
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2008

Approved by:

Major Professor
Haiyan Wang

Copyright

Abstract

This thesis introduces a new partitioning algorithm to cluster variables in high dimensional low sample size (HDLSS) data and high dimensional longitudinal low sample size (HDLLSS) data. HDLSS data contain a large number of variables with small number of replications per variable, and HDLLSS data refer to HDLSS data observed over time.

Clustering technique plays an important role in analyzing high dimensional low sample size data as is seen commonly in microarray experiment, mass spectrometry data, pattern recognition. Most current clustering algorithms for HDLSS and HDLLSS data are adaptations from traditional multivariate analysis, where the number of variables is not high and sample sizes are relatively large. Current algorithms show poor performance when applied to high dimensional data, especially in small sample size cases. In addition, available algorithms often exhibit poor clustering accuracy and stability for non-normal data. Simulations show that traditional clustering algorithms used in high dimensional data are not robust to monotone transformations.

The proposed clustering algorithm PPCLUST is a powerful tool for clustering HDLSS data, which uses p-values from nonparametric rank tests of homogeneous distribution as a measure of similarity between groups of variables. Inherited from the robustness of rank procedure, the new algorithm is robust to outliers and invariant to monotone transformations of data. PPCLUSTEL is an extension of PPCLUST for clustering of HDLLSS data. A nonparametric test of no simple effect of group is developed and the p-value from the test is used as a measure of similarity between groups of variables.

PPCLUST and PPCLUSTEL are able to cluster a large number of variables in the presence of very few replications and in case of PPCLUSTEL, the algorithm require neither a large number nor equally spaced time points. PPCLUST and PPCLUSTEL do not suffer from loss of power due to distributional assumptions, general multiple comparison problems and difficulty in controlling heterocedastic variances. Applications with available data from previous microarray studies show promising results and simulations studies reveal that the algorithm outperforms a series of benchmark algorithms applied to HDLSS data exhibiting high clustering accuracy and stability.

Table of Contents

List of Figures

List of Tables

Acknowledgments

There are many people I want to acknowledge. First, my advisor Dr. Haiyan Wang. I would like to thank her for accepting the challenge of having as her first PhD student someone working in another country. Thank you for your guidance.

In addition, I would like to thank the other members of my thesis committee: Dr. John Boyer, Dr. James Higgins, Dr. Dallas Johnson, Dr. Karen Garrett and Dr. Mitchell Neilsen. My appreciation for all your suggestions and deep interest in my work. My special thanks to Dr. John Boyer, a friend of all students who works hard to help them succeed in their graduate studies. Thanks also to Pam Schierer, another person that made a difference when I was in Kansas State University.

My deepest thanks also to Dr. Donna Davis, Dr. Susana L. Valdovinos and Dr. Carol W. Shanklin for their hard work in helping me in some dificult moments.

I really consider Kansas State University and Kansas as my home in america. Among all those people who made it possible, I would mostly like to thank my close friends Robert Poulson, Mike Anderson and Samuel Wilson who, together with their families, adopted me while in KSU. They made me understand better, appreciate and enjoy the american culture.

I made other great friends at KSU. I would like to give special thanks to Fátima, Paulo, Paulinha and Lucas for their friendly solicitude towards me and my family. I would also like to thank William Dall' Acqua, Frank and Tania for their loving support.

I am also very grateful to my colleagues in Universidade of Brasília. Special thanks to Maria Teresa Leão Costa, Raul Yukishiro Matsushita and Geraldo da Silva e Souza.

Thanks also to *Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)* for the partial support in the first years of my graduate studies, and Universidade de Brasília for the support during my studies.

Besides, my thanks to the most important people in this journey and in all my life: my family. To my wonderful and loved parents, Adolfo and Nilse, for their infinite love and motivation, and to my brother Ricardo, my sisters Rosana and Claudia, my brothers and sisters-in-law, my mother-in-law, my aunt Augusta, for their friendship and love. Also, the most important people and the reason of my life, my loved wife Michelle, my daughter Beatriz and my son Felipe.

Finally, I would like to specially acknowledge my best friend and brother Ricardo. He is my inspiration in life and in academic world. I'm so lucky to have him in my life.

Dedication

This thesis is dedicated to my wife Michelle, my daughter Beatriz and my son Felipe. They are the reason of my success, the reason of my life. I love you.

1

Introduction

The advent of new technologies for collecting and storing data has motivated the research of inference methods applied to high dimensional low sample size data in areas such as microarray experimentation (Pomeroy et al. [81]), spectrometry studies (Thiele [107]), pattern recognition (Reese [83]) and agriculture screening trials (Brownie and Boos [13]). For example, scientists have been able to study complex disorders through the monitoring of expression of thousands of genes from a single DNA chip, known as DNA microarray, [103],[76], [58], [95], [67], [41], [6], [89], [36], and one of the most important statistical learning technologies used to identify groups of differentially expressed genes has been cluster analysis, [28], [7], [72], [118], [49], [47], [33]. According to McLachlan [67], reasons for clustering of genes are: to discover genes with difference in expression in different tissues; to discover genes belonging to a particular pathway; to find common characteristics in genes declared similar through a comparison of expression patterns. Clustering can also be used as an exploratory tool to compare different experimental conditions (as a batch of reagents, technicians), to support visual methods in generating hypotheses about the existence of possible groups, to identify subgroups in complex data, to identify gene expression patterns in time or space and to reduce redundancy in prediction. More about the subject is available

in Segal et al. [90, 91].

A medium-size microarray study often contains information from thousands of genes with no more than a hundred samples for each gene. The dimensionality of the study will impose many restrictions to traditional statistical analyses. Drawbacks of available clustering algorithms are the difficulty in specifing the number of clusters in advance, their sensitiveness to outliers, the long processing time, their lack of robustness in the presence of small perturbations, their non-uniqueness, problems with inversion, distributional assumptions, and their failure to computate covariance matrices when one or more components is singular or nearly singular. Chapter 2 reviews some available clustering algorithms and discusses the advantages and disadvantages of each algorithm.

This thesis introduces a new algorithm for clustering high dimensional, low sample size (HDLSS) data and a new test that can be used when clustering high dimensional longitudinal, low sample size (HDLLSS) data. The objective of the new algorithm and test is to cluster a large number of variables in problems where there is a small number of replications per variable and when this replicated data is observed over time. Statistically, clustering of HDLSS data can be viewed as unsupervised separation of data originating from high dimensional mixtures of distributions, where each cluster is represented by data from the same distribution. For reasons stated above, this dissertation proposes a new partitional algorithm using a robust measure of similarity that can automatically determine the number of clusters. The robust similarity measure evolves from p-values obtained from the test of no nonparametric effect of groups (see Akritas and Papadatos [3]) specifically developed for the HDLSS and HDLLSS data structures. The new algorithm does not suffer from many of the drawbacks that traditional algorithms have and can obtain groups with high accuracy and stability. Additionally, both algorithms are fast and do not show memory allocation problems observed in some algorithms when the number of variables in the study is very high [1]. Applications to microarray gene expression data for colorectal cancer and to a HIV

[1] For example, when used 10000 or more variables.

study are discussed in this thesis.

A review of the terminology related to cluster analyses is presented in the next section. The objective of this review is to clarify and differentiate common terms from different scientific areas that are frequently used without a precise meaning in the literature about clustering of high dimensional data. The terms covered are: (1) data mining; (2) statistical learning; (3) supervised/unsupervised learning; (4) clustering; (5) gene-based clustering; (6) class discovery; and (7) classification.

1.1 Background

Data mining and statistical learning are two fields of research that investigate methods that search for valuable information in large volumes of data using either automatic or semi-automatic techniques to discover meaningful patterns and rules in data [12, 43]. As pointed out by Hastie, Tibshirani and Friedman [43], **data mining** works with very large amounts of data and has the challenge of extracting information through the application of multivariate techniques; **statistical learning** has a similar challenge in finding meaningful information in data with the use of supervised and/or unsupervised learning. **Supervised learning** has the goal of investigating the influence that some measured or preset *inputs* have in one or more *outputs* using a *training set* of data from previously solved problems to build a model. Examples are: linear models (least squares), linear discriminant analysis, classification, nonparametric density estimators, neural networks, probabilistic boolean networks, support vector machines, distance weight discrimination and sliced inverse regression. **Unsupervised learning** has the goal of inferring properties of a set of measured variables without using the help of a supervisor, i.e., when there are no *outputs* to either specify if the answer (prediction) is correct or to give a degree of error for each inferred observation. Examples are: cluster analysis, association rules in market basket analysis and dimensional reduction techniques. The main difference between data mining and statistical learning is that the former works with a large volume of information about a relatively

small number of variables (factors, experiments, parameters) while the latter works with an extremely small volume of information about a relatively large number of variables. In data mining problems, the main difficulty is that most statistics will be less conservative than in traditional data sets only because the number of observations is too large. In statistical learning, asymptotics do not work anymore because of the small sample sizes.

Bioinformatics uses both computational algorithms in data mining and statistical learning to solve problems in biology and medicine. The techniques studied in this monograph have their main focus in the *unsupervised statistical learning* part of bioinformatics and their application in the clustering of genes in microarray data, also known as **gene-based clustering**. Theodoridis and Koutroumbas [106] describe **clustering** as the objective of discovering patterns in data that helps the researcher form "sensible" groups of similar objects, called clusters, to derive useful conclusions about differences that resulted in formation of such clusters. The discovery of meaningful groups is defined as **class discovery** by Simon et al. [95] and it involves one step further in cluster analysis, since it looks for some meaningful clusters found by a specific algorithm. For example, there are efforts to incorporate biological knowledge in the clustering process through the use of Gene Ontology in recent years [10, 47].

This thesis studies the clustering of high dimensional data with a focus on gene expression data where the number of genes is large and the number of experiments (tissues, samples) for each gene is small. Although some authors, like Dettling and Bühlman [20], also see the dual problem of clustering a small number of experiments that have information from a large number of genes as a high dimensional clustering problem. These problems are not HDLSS data problems as emphasized by many authors (see for example Fridlyand and Dudoit [31]). Traditional clustering techniques are expected to work well in those problems.

In the clustering process some steps should be followed before and after application of clustering algorithms [106]. Selection of variables or factors to be clustered and choice of similarity (or dissimilarity) measures are two steps that must be defined before the use of

a clustering algorithm. Validation and interpretation of results should follow a clustering algorithm application. In order to quantify the degree of similarity or dissimilarity between variables, different measures of distance are used and they are fundamental to the clustering process once clustering seeks objects that are most similar or least dissimilar, i.e., objects in same cluster are more similar to one another than objects in different clusters and each cluster should be as different as possible from other clusters. A clustering process can be thought as a minimization of a loss function formed by the within-cluster dissimilarity [106, 98] with separated groups as different as possible. An overall classification of clustering algorithms is shown in Fig. 1.1. The majority of clustering algorithms can be divided into **hard** or **fuzzy** algorithms [2]. In hard clustering methods, objects are allocated to one and only one cluster, while in fuzzy methods the same object can be allocated to more than one cluster. Hard clustering methods are the most commonly used methods in the literature and can be divided into **partitional** and **hierarchical** methods. **Partitional** methods seek to optimally divide objects into a fixed (defined or not) number of clusters, while **hierarchical** methods produce a nested sequence of clusters in agglomerative or divisive ways. In hierarchical methods the choice of the number of clusters is made after the complete sequence of groups is formed. If the sequence of groups is formed starting with n groups of 1 object, each, and finishing with 1 group including all objects in it, then the hierarchical process is called agglomerative. The inverse process is called divisive hierarchical clustering. Note that in the partitional method there is no nested sequence of groups, but a fixed number of clusters is to be obtained, even though a series of different numbers of clusters can be compared in the partitional method until the best number is found according to some criteria that maximize the similarity within each group and/or the dissimilarity between groups.

Additionally, the way hard or fuzzy clustering methods work with data can be divided into combinatorial, mixture modeling, and mode seeking methods. In combinatorial algo-

[2]Some authors [34, 106, 49] divide clustering algorithms in many other categories, as center-based, graph-based, density-based algorithms, etc, but they are basically variations of decision criteria methodologies used in partitioning or hierarchical algorithms.

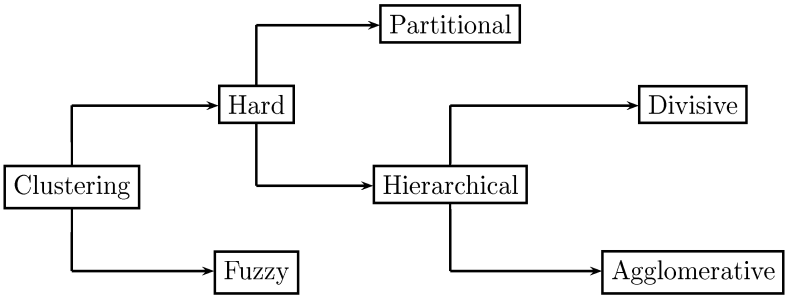

Figure 1.1. *Diagram of clustering algorithms (Gan et al. [34]).*

rithms, data is clustered without considering any underlying probability model. Mixture modeling uses the idea that the data came from a mixture of groups originating from i.i.d. samples of populations that can be described by some probability function. In mode seeking, data around the modes of a probability density function are considered to be from the same cluster.

The number of articles about clustering methods in microarray gene expression investigations has grown exponentially in recent years. Pubmed [3] revealed more than 1000 articles published in 2006. The volume of information generated by gene expression investigations has brought many challenges for scientists, not only in molecular biology, but also in areas such as informatics, mathematics and especially statistics. Zakharkin et al. [119] discuss some of the challenges faced by high-dimensional biology with the increase in studies about microarray technology.

In the context of clustering gene expression data, the following data layout can be used. The gene expression data from a microarray experiment is represented by a real-valued expression matrix $M = \{x_{ij} | 1 \leq i \leq a, 1 \leq j \leq b\}$ where each row of the matrix $(i = i, \ldots, a)$ represents a gene[4] and each column $(j = 1, \ldots, b)$ represents a tissue or condition in the

[3]Searching words "microarray" and "gene expression" and "clustering". Pubmed is a service of the U.S. National Library of Medicine that includes over 17 million citations from Medline and other life science journals for biomedical articles.

[4]Actually, rows represent DNA sequences that can be genes, cDNA clones or expressed sequence tags

microarray experiment. In longitudinal studies, replicated genes are represented in multiple rows, while time-course is represented in different columns of the matrix. Tables 1.1 and 1.2 illustrate the high dimensional replicated data layout for non longitudinal and longitudinal cases, respectively.

Table 1.1. *High dimensional replicated data layout. Here* $a \to \infty$ *and* $n_i \geq 2$.

Variable	Observations				Sample size
1	X_{11}	X_{12}	...	X_{1n_1}	n_1
2	X_{21}	X_{22}	...	X_{2n_2}	n_2
⋮	⋮	⋮	⋮	⋮	⋮
a	X_{a1}	X_{a2}	...	X_{an_a}	n_a

Table 1.2. *High dimensional replicated data layout in longitudinal design. Here* $a \to \infty$ *and* $n_i \geq 2$, $t_j \geq 2$.

		Time Points			
Variable	Observations	t_1	t_2	...	t_b
1	1	X_{111}	X_{121}	...	X_{1b1}
	⋮	⋮	⋮	⋮	⋮
	n_1	X_{11n_1}	X_{12n_1}	...	X_{1bn_1}
⋮	⋮	⋮	⋮	⋮	⋮
a	1	X_{a11}	X_{a21}	...	X_{ab1}
	⋮	⋮	⋮	⋮	⋮
	n_a	X_{a1n_a}	X_{a2n_a}	...	X_{abn_a}

In cluster analysis, the gene expression matrix can be analysed in three different ways: gene-based, sample-based and subspace clustering. Gene-based clustering is the clustering of genes considering experiments as replications. In contrast to gene-based clustering, sample-based clustering is the clustering of samples using genes as observations. Finally, subspace clustering treats both genes and samples, symmetrically such that both genes and samples

(ESTs) in the experiment. Here we focus on clustering of genes and do not distinguish between DNA sequences.

can be treated as variables, to be clustered, or replications in data. This thesis consider, gene-based clustering analysis using a partitional method based on mixture modeling data.

The literature on cluster analysis has a vast number of algorithms for different data types and structures. An excellent review of a large number of algorithms and different similarity measures used in each algorithm can be found in Theodoridis and Koutroumbas [106] and Kaufman and Rousseeuw [52].

As discussed by Gan et al. [34], data clustering is often confused with classification methods like random forest and others. However, **classification methods** use class comparison and class prediction to deal with **supervised learning**. Clustering is an unsupervised approach and it can be defined as an **unsupervised classification**, because it relies on the classification of objects into classes that were not predefined. However, clustering can also be used for class comparison. In this situation, the observed difference between paired classes is used to cluster variables (factor levels, genes). Classes with large differences will result in different groups of variables. In paired microarray data, one can consider, for example, the difference in gene expression of normal and tumor tissues. Genes with high difference will be grouped in high (positive) or low (negative) expressed genes, while genes with invariant expression in the two types of tissues are grouped in another cluster of no expressed genes. The number of clusters formed by an algorithm in class comparison will depend on the intensity of the difference between classes. The colorectal cancer in example considered in Section 5.1 is one application of clustering in a class comparison problem with paired microarray data.

Since the main application of the algorithms developed in this thesis is in microarray gene expression studies, a brief review of microarray technology is presented in Appendix A.

1.2 Contribution

This work introduces a new partitioning algorithm to cluster variables in high dimensional data, where the number of variables is large and the number of replications per variable

is small. The development of the algorithm incorporates both the high dimensional low sample size (HDLSS) case and the high dimensional longitudinal low sample size (HDLLSS) case. The measure of similarity between groups is given by the p-value from a nonparametric hypothesis test of no group effect specific to HDLSS data and from a nonparametric hypothesis test for no simple group effect developed in HDLLSS data. In the longitudinal setting, the original observations are used, but in non-longitudinal replicated data, the overall ranks of gene expressions are used in the clustering algorithm. This makes the analysis more robust and invariant to monotone transformations of data. In both the HDLSS and HDLLSS cases, the algorithm requires very few assumptions, no intervention, and no pre specification of the number of groups. Simulations show that the procedures can obtain groups with high accuracy and stability without any supervision or data reduction. Additionally, the procedures are fast and can handle ultra high-dimensional clustering without memory allocation problems. The applications to colorectal cancer and HIV microarray data show promising results that can offer insight for biologists and scientists in related areas for further experimentation.

2

Literature Review

This chapter presents a review of some benchmark algorithms used in the clustering of high dimensional data. In clustering there are two components, the proximity measure and the algorithm. The proximity measures are reviewed in Section 2.1 and the algorithms are reviewed in Section 2.2. The algorithms reviewed are used in Chapter 4 to compare their efficiency and stability with the new partitioning algorithm for HDLSS data introduced in Chapter 3. The efficiency of clustering algorithms is obtained using the Adjusted Rand Index that is reviewed in Section 2.3.

2.1 Review of Proximity Measures

In cluster analysis data is represented by a proximity measure. This measure quantifies the degree of similarity or dissimilarity (distance) between pairs of variables. For two vectors $\mathbf{x}$ and $\mathbf{y}$ in p-dimensional space, a dissimilarity measure d is defined to be a distance function if

- $d(\mathbf{x}, \mathbf{x}) = 0$;
- $d(\mathbf{x}, \mathbf{y}) \geq 0$;

- $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$.

A similarity measure s between $\mathbf{x}$ and $\mathbf{y}$ is defined if

- $s(\mathbf{x}, \mathbf{y}) = s(\mathbf{y}, \mathbf{x})$;
- $s(\mathbf{x}, \mathbf{y}) > 0$;
- $s(\mathbf{x}, \mathbf{y})$ increases as the similarity between $\mathbf{x}$ and $\mathbf{y}$ increases.

Some common dissimilarity measures used in cluster analysis are the Euclidean and Manhattan distances that are particular cases of the Minkowski distance that is defined as

$$d(x_i, y_j) = \left(\sum_{k=1}^{p} |x_{ik} - y_{jk}|^r \right)^{1/r}, \quad r \geq 1 \tag{2.1}$$

Euclidean ($r = 2$) and Manhattan ($r = 1$) distances usually are dominated by variables with the largest scales and only work well when the data set has compact or isolated clusters. One alternative is to standardize the Euclidean distance by using,

$$d(x_i, y_j) = \sum_{k=1}^{p} \frac{(x_{ik} - y_{jk})^2}{s_k^2} \tag{2.2}$$

where s_k^2 is the variance of the kth variable. This standardized distance is called the Karl Pearson distance. As a drawback of all sum of squares, this distance is expected to be sensitive to outliers.

The above distances ignore the correlation between variables. A distance that take into account the correlation between variables is the Mahalanobis distance defined as,

$$d(x_i, y_j) = \sqrt{(x_i - y_j)^T \Sigma^{-1} (x_i - y_j)}, \tag{2.3}$$

where Σ is the variance-covariance matrix of the data set. The Mahalanobis distance is invariant under nonsingular transformations, i.e., if $z_i = Cx_i$ and $r_j = Cy_j$ for all i and j, then $d(x_i, y_j) = d(z_i, r_j)$. However, to use this distance, it is necessary to estimate the

inverse of the covariance matrix, which requires a large number of samples. In case of small sample sizes, the estimated covariance matrix is often not invertible.

In clustering of microarray gene expression data it is common to use the Inner Product defined as,

$$s(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y} = \sum_{i=1}^{n} x_i y_i, \tag{2.4}$$

and the Pearson's Correlation Coefficient defined as,

$$s_r(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}_c^T \mathbf{y}_c}{\|\mathbf{x}_c\| \, \|\mathbf{y}_c\|}, \tag{2.5}$$

where $\mathbf{x}_c$ and $\mathbf{y}_c$ are the original vectors centered about their respective means.

As a similarity measure, the inner product is usually used with normalized vectors and this is not a recommended procedure in clustering gene expression data. Pearson's Correlation Coefficient has been used frequently in gene expression analysis. However, studies indicate that this measure is not robust to outliers and data from non-Gaussian distribution. Some gene expression studies (see Eisen et. al [25]) use a dissimilarity measure that is obtained from s_r by the transformation,

$$d(x_i, y_j) = \frac{1 - s_r(x_i, y_j)}{2}. \tag{2.6}$$

More details about proximity measures can be found in Mardia [64], Theodoridis and Koutroumbas [106], Kaufman and Rousseeuw [52], Allison et. al [6], Johnson and Wichern [51] and Gan et. al [34].

2.2 Review of Clustering Algorithms

2.2.1 Partitional Clustering Methods

- ***K*-means, PAM and Clara**

K-means was proposed by MacQueen [68] and is one of the most popular partition-based methods. It partitions the dataset into k disjoint subsets, where k is predetermined. For

each subset it obtains initial centers $\hat{\mu}_1, \ldots, \hat{\mu}_k$ and minimizes the sum of squared distances from each observation to its cluster center $\hat{\mu}_c, c = 1, \ldots, k$. The algorithm keeps adjusting the assignment of objects to the closest current cluster mean until no new assignments of objects to clusters can be made.

One Advantage of this algorithm is its simplicity. There are no problems with missing observations and low time complexity (see Jiang et. al [49]). One drawback is that in gene expression data it is difficult to specify the number of clusters in advance. Another drawback is that the algorithm is sensitive to outliers since it works with squared distances. A third drawback is that centroids are not meaningful in most problems.

The Partitioning Around Medoids (PAM) algorithm was introduced by Kaufman and Rousseeuw [52], It is based on the search of k representative objects, called medoids, among the objects of the dataset. The medoids are points with smallest average dissimilarity to all other points. The algorithm follows the same sequence of steps that are followed by the k-means algorithm, but the use of medoids instead of means makes the algorithm more robust to outliers. Also, the center of each cluster is now more representative since it is an element of the dataset. PAM can also be used in datasets that have categorical and/or other types of discrete data, such as binary data. One of the problems of the PAM algorithm is the requirement that the desired number of clusters must be predetermined.

When working with gene expression data, there are typically thousands of genes to be clustered. In this case, both the k-means and PAM algorithms are slow and not practical because for a fixed number k of clusters, the number of possible subsets from a objects increases exponentially at the rate k^a. One can imagine situations where different numbers of clusters are found and the algorithm has to be repeated many times. One algorithm that tries to solve this problem is CLARA (Clustering LARge Applications). CLARA is a method based on PAM that attempts to deal with large dataset applications. Instead of applying PAM to cluster all the data, CLARA uses the PAM algorithm to first cluster a sample from a set of objects into k subsets. After this first step, each object not belonging to the initial

sample is allocated to the nearest representative object, and a measure of clustering of the entire dataset is obtained. This measure is compared with n other measures obtained from the application of the algorithm in n different initial samples. The best clustering obtained from the different samples is the one selected by the algorithm.

Kaufman and Rousseeuw [52], Theodoridis and Koutroumbas [106] detail the k-means, PAM and CLARA algorithms and present some variations of each, such as CLARANS. Simulations have shown that such methods are still slow when clustering high dimensional data.

- **Self-Organing Maps - SOM**

Self-organizing maps (SOM) are partitioning algorithms introduced by Kohonen [59] as a neural network process that consists of clusters in low dimensional grids formed by cells known as neurons. Each neuron is represented by a d-dimensional reference vector (prototype vector). The dimension d is specified through an input vector space and the distributions of this input vector space directs the movement of the reference vectors to the denser areas of the vector space. In the SOM algorithm, adjacent neurons of a grid represent clusters that are close to one another. An example of such structure is presented in Fig. 2.1, using software R[1], where a SOM grid of 6 by 5 was applied, by Tamayo et al. [104], to a microarray of 6601 genes, measured at 18 time points in the yeast cells cycle.

SOM is basically a data reduction technique with an algorithm similar to k-means that has the advantage of representing similar clusters in closer points of a graphical map which is exactly the appealing feature of SOM technique. Examples of SOM applications in clustering of gene expression data are given by Tamayo et al. [104], Törönen et al. [108], Yano and Kotani [117] and Garrigues et al. [35].

[1]Copyright 2007 from The R Foundation for Statistical Computing.

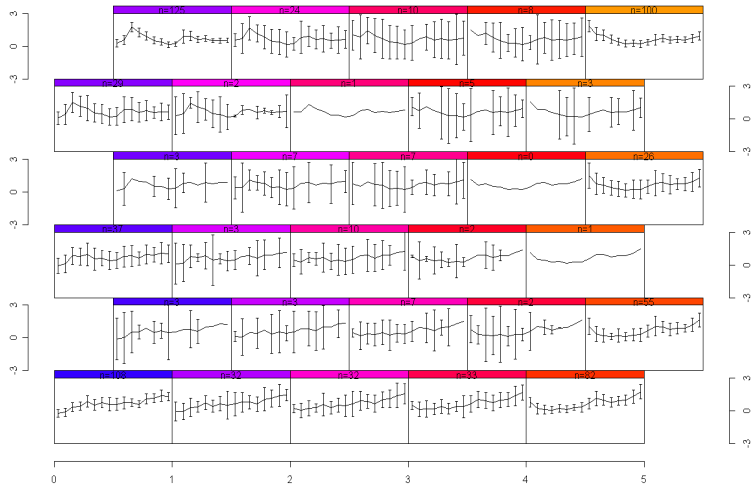

Figure 2.1. *Example of two dimensional SOM grid for a microarray data produced with R package.*

2.2.2 Hierarchical Clustering Methods

Hierarchical clustering algorithms divide or merge a dataset into a sequence of nested partitions. The way the hierarchy of nested partitions is formed is what defines agglomerative or divisive hierarchical clustering. In the agglomerative method, clustering starts with each single object in a single cluster and it continues to cluster the closest pairs of clusters until all the objects are together in just one cluster. Divisive hierarchical clustering, on the other hand, starts with all objects in a single cluster and keeps splitting larger clusters into smaller ones until all objects are separated into unit clusters. Both hierarchical methods have a natural way of representing nodes of splits or unions of clusters through a graphical tree called a dendogram. In each approach, different strategies can be used to split or merge clusters and the same data can produce a different sequence of nodes for agglomerative or divisive clusterings. One example of such result is presented in Fig. 2.2 where the same

data produced different dendograms in agglomerative and divisive algorithms[2]. Kaufman and Rousseeuw [52], Johnson [50], and Johnson and Wichern [51] describe some of the main hierarchical clustering algorithms.

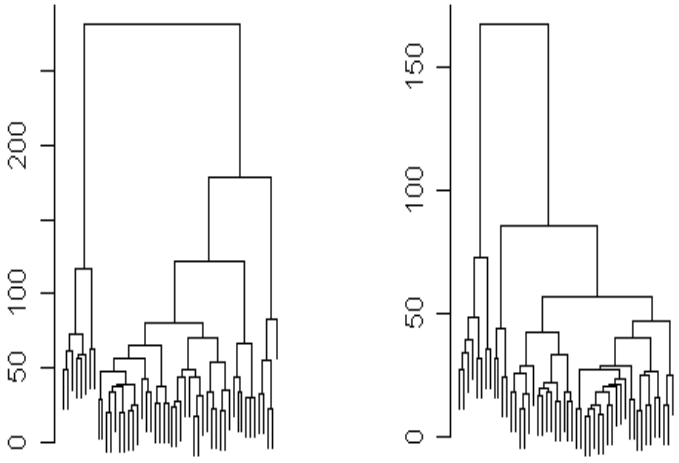

Figure 2.2. *Agglomerative (left) and divisive hierarchical (right) dendograms for the same dataset. Agglomerative dendogram should be read from bottom to top graph and divisive should read in opposite direction.*

In comparative studies with PPCLUST, two agglomerative algorithms and one divisive algorithm are used: HCLUST (Hierarchical CLUstering), and AGNES (AGglomerative NESting); and DIANA (DIvisive ANAlysis clutering).

The following descriptions of each method are taken from the R help user's guide[3]. Additional details about the algorithms are described in Kaufman and Rousseeuw [52].

- *HCLUST performs a hierarchical cluster analysis using a set of dissimilarities for the objects being clustered. Initially, each object is assigned to its own cluster and then the algorithm proceeds iteratively, at each stage joining the two most similar clusters,*

[2]It occurs also due to choice of the dissimilarity (or similarity) measure to be used in each algorithm, as for example, group average, nearest neighbor and furthest neighbor.

[3]R is a free software. Copyright from The R Foundation for Statistical Computing.

continuing until there is just a single cluster. At each stage distances between clusters are recomputed by the Lance Williams dissimilarity updated formula according to the particular clustering method being used. In simulations it was used the Ward's minimum variance method with dissimilarities between the clusters in Euclidean metric. Ward's method minimizes the increase in total within-cluster sum of squared errors [4].

- *AGNES is another agglomerative clustering method such as HCLUST. It constructs a hierarchy of clusterings where, at first, each observation is in a cluster by itself. Clusters are merged until only one large cluster remains which contains all the observations. At each stage, the two nearest clusters are combined to form one larger cluster using average linkage method, i.e., the distance between two clusters is the average of the dissimilarities between the points in one cluster and the points in the other cluster.*

- *DIANA computes a divisive hierarchy. The diana-algorithm constructs a hierarchy of clusterings, starting with one large cluster containing all n observations. Clusters are divided until each cluster contains only a single observation. At each stage, the cluster with the largest dissimilarity between any two of its observations is selected. To divide the selected cluster, the algorithm first looks for its most disparate observation, i.e., which has the largest average dissimilarity to the other observations of the selected cluster. This observation initiates the "divisive group". In subsequent steps, the algorithm reassigns observations that are closer to the "divisive group" than to the "old group". The result is a division of the selected cluster into two new clusters.*

When clustering genes or a large number of objects, usually agglomerative algorithms are chosen for hierarchical clustering. The reason is that for agglomerative algorithms the number of possible fusions of two objects in the first step is $n(n-1)/2$, while the number of possible divisions in a first step of a divisive algorithm is $2^{n-1}-1$. In high

[4] Ward's hierarchical clustering method is based on minimization of loss of information from joining two groups. Other methods of agglomerative hierarchical clustering are single linkage, complete linkage, and average linkage. Details about each method are available on Johnson and Wichern [51].

dimensional data it makes the divisive algorithm extremely slow as the number of objects to be clustered increases. For example, in a microarray data, where the objective is to cluster just 1000 genes, the agglomerative algorithm will have 499500 possible fusions, while the divisive algorithm will have 5.357543×10^{300} possible divisions, resulting in a extremely slow divisive algorithm. Although, DIANA is a divisive algorithm, it can be applied to microarray datasets just because the algorithm does not consider all possible splits in each step, but it uses, instead, an iterative procedure that is explained in details in Kaufman and Rousseeuw[52]. Other procedures for clustering objects in hierarchical algorithms have been used in the microarray literature. Some are deterministic-annealing, cluster identification by connectivity kernels (CLICK) and a cluster affinity search technique (CAST). Jiang et al. [49] has an excellent review of such techniques with a large number of references.

Besides a long processing time, the major drawback in hierarchical algorithms is the fact that a bad decision about splitting or grouping objects in one step can not be corrected in the following steps. Tamayo et al. [104] also indicates that hierarchical clustering suffers from a lack of robustness with small perturbations changing the hierarchical structure considerably. Other problems cited are non-uniqueness, and inversion problems that results in complicate interpretation of the hierarchy in these methods.

- **Hierarquical Clustering by Minimun Energy Distance (ϵ-clustering)**

Clustering using ϵ-distance was introduced by Székely and Rizzo [102] as an agglomerative hierarchical method that merges two groups of objects with minimum energy distance at each step. The minimum energy distance of two groups G_i and G_j is defined by

$$e(G_i, G_j) = \frac{n_i n_j}{n_i + n_j}(2D_{ij} - D_{ii} - D_{jj}),$$

where

$$D_{ij} = \frac{1}{n_i n_j} \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} ||Y_{ip} - Y_{jq}||^{\alpha},$$

and $||\cdot||$ denotes the Euclidean norm, and Y_{ip} is the pth observation in the ith group. The ϵ-distance measures both the heterogeneity between clusters and the homogeneity within

clusters and when $\alpha = 2$, it is equivalent to Ward's method in agglomerative hierarchical clustering. According to Rizzo and Székely [85], when $\alpha = 1$ the ϵ-clustering is particularly effective in high dimensional problems, and is more effective than some standard hierarchical methods when clusters have equal means. The authors also call attention to the fact that if clusters are characterized by their means, then Ward's method ($\alpha = 2$) is a better choice, but if clusters are characterized by their distributions, then $0 \leq \alpha < 2$ may be a better choice. Other properties of ϵ-clustering are found in Székely and Rizzo [102]. In comparisons made with PPCLUST $\alpha = 1$ was used. In simulations, the ϵ-clustering algorithm produced results very similar to those obtained using hierarchical clustering with Ward's method of computing distances.

2.2.3 Fuzzy clustering

Fuzzy clustering is a general form of partitional clustering[5] where each object in the dataset is given a probability of inclusion in a cluster through the use of membership coefficients that range from 0 to 1. The membership coefficients add to 1 for each object. The advantage of fuzzy clustering is that one object can have an equal membership coefficient for different clusters when its location in one or another cluster is not clear. An interesting example is presented by Kaufman and Rousseeuw [52], page 165, where the allocation of some elements in the data are not as evident as others. The example is reproduced in Fig. 2.3 and one should note that object 6 has 2 likely clusters for allocation while object 13 has three likely clusters for allocation.

Using the package FANNY (Fuzzy ANalysis) from R software, with 3 clusters, the membership coefficients of object 6 are equal to $0.45, 0.38, 0.17$ and for object 13 the coefficients are $0.23, 0.38, 0.39$, for clusters 1, 2 and 3 respectively. All other objects have membership coefficients of at least 0.65 to one of the clusters. Thus, fuzzy allocation spreads object 6 mostly in 2 clusters, while object 13 spreads out over 3 clusters. This description of the

[5] Actually, the fuzziness principle can be extended to many partitional algorithms, resulting in fuzzy k-means, fuzzy k-modes, and many other variations of partitional algorithms.

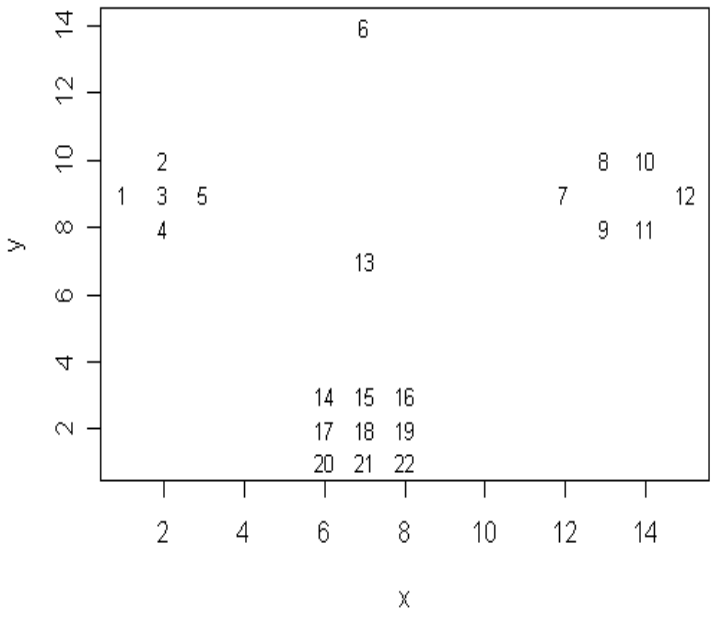

Figure 2.3. *Dataset with two intermediate objects (*6 *and* 13*). Example generate in R language, based on KaufmanRousseeuw [52].*

uncertainty of objects in real data is seen as one of the main advantages of fuzzy clustering. Unfortunately, in large datasets FANNY is computationally slow and results in very difficult comparisons of clusters. In simulations the cluster allocated to an object was the cluster with highest membership coefficient. This allows comparisons with PPCLUST.

2.2.4 Model-Based Clustering (MCLUST)

Model-based clustering is a hierarchical clustering method with a flavor of fuzzy clustering. It does a model-based agglomerative hierarchical clustering by mixtures of distributions and provides an estimated probability (or uncertainty measure) that an object i belongs to a cluster k. The MCLUST package by Fraley and Raftery [30] assumes a mixture of normal

distributions using

$$f(x) = \sum_{i=1}^{g} \pi_i f_i(x|\mu_i, \Sigma_i)$$

where the data is represented by x, π_i is the probability that an observation belongs to distribution i and $f_i(x|\mu_i, \Sigma_i)$ is the normal density of group i with mean μ_i and variance-covariance Σ_i. The parameters are estimated by Expectation-Maximization (EM) algorithm and the object is assigned to the component with maximum probability. The final components will be the clusters. In this way, the estimated probabilities of assignment of an object to a component can show if one object is highly correlated to more than one cluster, as in fuzzy clustering. In MCLUST, the normal distribution clustering modeling decomposes the variance-covariance matrix Σ_i in a set of geometric features that provide information about volume, shape and orientation of the components or clusters. Different parameterizations of the covariance matrix generate different models that are compared through the Bayesian information criterion (BIC) of Schwarz[88]. The best model is considered the one with maximum BIC over all models and numbers of components considered. Figure 2.4 shows a typical graph produced with MCLUST for different covariance parameterizations and different number of components[6].

The assignment of probabilities to components in MCLUST also helps to produce graphical outputs, as projected clusters and projected degree of uncertainty in the allocation of objects to clusters. Figure 2.5 has examples of such graphs.

As observed by Fraley and Raftery [30], model-based clustering has also many drawbacks. The assumption that the dataset fits a specific distribution is not always accepted and may be difficult to justify in microarray or high dimensional data analysis. Also, the EM computations can fail when the covariance matrix of one or more components is singular or nearly singular, or yet, if the clusters contain few observations.

Model-based clustering using mixture of distributions has been largely studied with many applications to high-dimensional and microarray data. Examples are Fraley [28], McLachlan

[6]For details about covariance parameterizations see Gan et al. [34] or Fraley and Raftery [30].

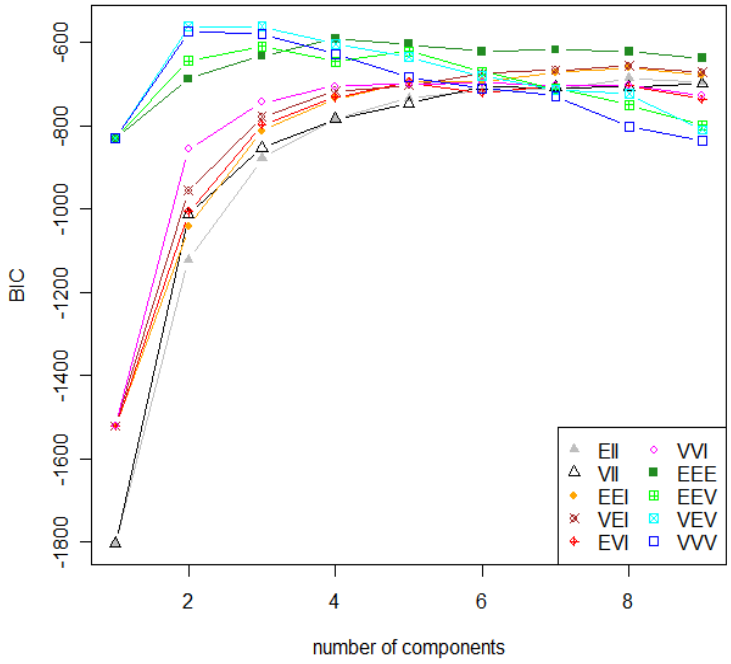

Figure 2.4. *BIC comparison of different models with different number of components in clustering of a data. The covariance structures considered are EII (spherical with equal variance), VII (spherical unconstrained), EEI (diagonal with equal variance), VEI (diagonal with equal shape), EVI (diagonal with equal volume) , VVI (diagonal unconstrained), EEE (ellipsoidal with equal variance), EEV (ellipsoidal with equal volume and shape), VEV (ellipsoidal with equal shape) and VVV (ellipsoidal unconstrained). The best model is VEV with 2 clusters.*

and Peel [65], McLachlan et al. [67], Fraley and Raftery [29], McLachlan et al. [66], Gan et al. [34], Gottardo et al. [38] and Gottardo et al. [39][7].

In comparison with other methods, MCLUST shows superior results when considering clusters with symmetric distributions and a larger number of replications. However, MCLUST has poor results for simulated clusters with asymmetric distributions, since it is based on the assumption of having a mixture of normally distributed clusters. In all

[7]The last two articles more directed related to Bayesian methods for gene expression in microarrays.

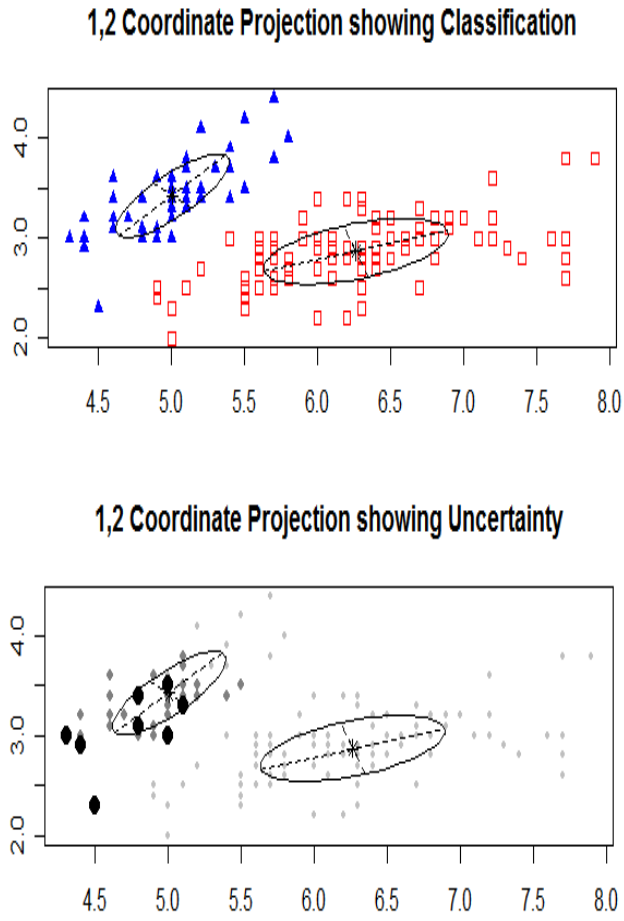

Figure 2.5. *Example of coordinate projections of clustered objects showing clustering (named as classification) and uncertainty of allocation.*

simulations presented in this thesis, MCLUST was less efficient than PPCLUST.

In terms of clustering genes in microarray data, ANOVA methods have been used to declare genes as differentially expressed or not. In this thesis, all the clustering methods are based on similarity measures that come from p-values obtained from hypothesis testing in ANOVA specially developed under the setting of large number of factor levels and small sample sizes. The next subsection gives a brief review of the use of ANOVA in gene

expression studies.

2.2.5 ANOVA methods in clustering analysis

ANOVA methods (see Pavlidis [78], Kerr et al. [53], Milliken et al. [71]) have been proposed to determine which genes are differentially expressed over samples obtained under different experimental conditions or across different kinds of tissue samples. However, few studies have been concerned with the use of ANOVA methods when the number of genes is large and the number of observations is small. Usually the application of ANOVA methods for discovering differentially expressed genes in replicated microarray data is reduced to the identification of just two categories, expressed and non-expressed genes through the comparison of the expression of pairs of genes (Pan [75], Pavlidis et al. [78], and Sykacek et al. [101].). However, traditional ANOVA imposes strong assumptions that make it impractical and not powerful enough. The assumptions include normality, independence and homocedasticity of errors. The assumption of normality of gene intensity distributions is difficult to justify in many microarray applications even after applying the common log transformation used in most studies. To solve this problem, authors use permutation tests or bootstrap methods, as in Kerr et al. [53], but then more replicates are required than are usually found in HDLSS or HDLLSS datasets as explained by Simon et. al [95]. Some other ANOVA methods with large numbers of factor levels are given in Boos and Brownie [14], [13], Akritas and Arnold [2], Akritas and Papadatos [3], and Wang and Akritas [112][8]. Another assumption that requires attention is homocesdasticity. Dubin et al. [24] showed that usual log transformations are subject to problems such as not stabilizing data in microarrays uniformly. Finally, multiple testing in ANOVA studies generate significance levels with extremely low p-values due to the high dimensionality of the problem and experimentwise Type I error rates are too conservative. In this situation it is frequently used in applications of false discovery error rate (FDR) by Benjamini and Hochberg [11]. FDR is defined as the

[8]All cited references are not concerned directly with microarray experiments, but with ANOVA when the number of factors is large.

expected proportion of false positives among the number of rejections. However, methods using FDR are still restricted to a gene by gene analysis and cannot borrow information from other genes.

The use of non-parametric methods may correct many of the problems mentioned above, but in many cases it results in a loss of power and if, still requires some assumptions to be valid as mentioned by Mehta et al. [69] and Roy [86].

In this thesis, two specific nonparametric hypothesis tests of no group effect are used in presence of a large number of factors levels when the number of observations is small. In this case, the p-values from the tests can be used as similarity measures in algorithms for clustering. Chapter 3 introduces the test used for clustering HDLSS data and a novel partitioning algorithm that has many advantages over existing algorithms in the literature. In Chapter 5, the developed algorithm is evaluated through numerical comparisons and applied to a microarray data. In Chapter 6 a new test is developed for simple effects in high dimensional low sample size data in longitudinal studies. The asymptotic distributions of the test statistics are derived. Simulation studies concerning Type I error rates and power estimates of the new test are reported. This new test is used for implementation of the clustering algorithm that allows clustering of a large number of variables when only few replications over time are available. Finally, Chapter 7 presents conclusions about results obtained and discusses future research possibilities and perspectives. Each chapter identifies important references in the recent literature, but the reference list is not exhaustive due to the huge and even daily amount of new publications in this area.

Comment:

Other algorithms for clustering replicated microarray gene expression data have been used for specific situations with relative success. Examples: gene-shaving (Hastie et al. [42], and K-A. Do et al. [23]), density-based hierarchical clustering (Jiang et al. [48]), clustering via iterative feature filtering or CLIFF (Xing and Karp [116]), plaid models (Lazzeroni and Owen [60]), subspace clustering (Parsons et al. [77]), and coupled two-way clustering

analysis of gene microarray data (Getz et al. [37]). However, comparisons by simulation using those methods were not possible for the following reasons: lack of flexibility to do the simulations (no code access), poor documentation or/and incompatibility with different operational system platforms and versions of statistical packages.

2.3 Cluster Quality by Adjusted Rand Index

The Adjusted Rand Index (ARI) from Hubert and Arabie [46] is a measure of agreement in order to compare clustering results against an external criteria. An external criteria is some standard result for clustering that is judged to be correct, or even the result of clustering with a different methodology that someone wants to compare with other methods.

Consider, for example, a partition $P_1 = \{rc_1, rc_2, \ldots, rc_k\}$ representing k reference clusters that will be used to compare clustering procedures. Let $P_2 = \{oc_1, oc_2, \ldots, oc_c\}$ be a partition of c clusters obtained from some clustering algorithm (k-means, som, etc.). The Contingency Table[9] 2.1, represents the results of both clustering procedures, where n_{ij} is the number of objects that are in both clusters rc_i and oc_j, with $i = 1, \ldots, k$, $j = 1, \ldots, c$, and $n_{i.} = \sum_{j=1}^{c} n_{ij}$, $n_{.j} = \sum_{i=1}^{k} n_{ij}$.

Table 2.1. *Contingency Table of clustering agreement.*

	oc_1	oc_2	$\ldots$	oc_c	
rc_1	n_{11}	n_{12}	$\ldots$	n_{1c}	$n_{1.}$
rc_2	n_{21}	n_{22}	$\ldots$	n_{2c}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
rc_k	n_{k1}	n_{k2}	$\ldots$	n_{kc}	$n_{k.}$
	$n_{.1}$	$n_{.2}$	$\ldots$	$n_{.c}$	n

[9] Table adapted from Fridlyand and Dudoit [31].

Using this notation, the ARI can be calculated as,

$$\frac{\sum_i \sum_j \binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right] / \binom{n}{2}}. \tag{2.7}$$

ARI = 1, if the partitions agree completely, regardless of the permutation of the labels, or equivalently, if all elements in the same cluster in one partition are also together in some cluster in the other partition, for all clusters in both partitions. ARI = 0 when the elements in each partition are randomly assigned to each cluster. As an example, consider Table 2.2 with results from a clustering algorithm that is supposed to cluster 2000 objects in groups of sizes 200, 200, 800, 400, and 400 (reference clusters). The result obtained in Table 2.2 results in ARI = 0.93123.

Table 2.2. *Contingency Table of clustering agreement of* 2000 *clustered objects.*

	1	2	3	4	5	6	Total
1	0	6	182	7	4	1	200
2	0	3	1	0	0	192	200
3	6	791	2	0	0	1	800
4	2	2	0	0	396	0	400
5	0	19	0	375	2	4	400
Total	8	821	185	382	402	202	2000

A SAS© macro to calculate ARI was implemented for SAS© Version 9.1.3 and used in our simulations. The macro was adapted from Fisher and Hoffman [27] and corrected for the latest version of SAS© language[10]. R software has ARI implemented in package MCLUST.

[10]Instructions available in Appendix B.

3

Partition Clustering of HDLSS Data Based on p-Values

This chapter introduces a computational algorithm for partition clustering of a large number of variables with few observations per variable [1]. The algorithm uses the one-way ANOVA model based on ranks developed by Wang and Akritas [112]. Clustering of High Dimensional Low Sample Size (HDLSS) data has been extensively used in the analysis of replicated data represented by a mixture of unknown distributions where individuals from a cluster are all generated from the same distribution. However, existing clustering algorithms applied to HDLSS data suffer from several drawbacks as noted in Chapter 2. The algorithm introduced in this thesis does not suffer from such drawbacks. Additionally, the new algorithm has better performance in finding the correct groups of factor levels (variables).

The new procedure, based on ANOVA methods, uses the p-value of the test statistic from Wang and Akritas [112] as a measure of similarity between groups. Classical ANOVA models and ANOVA based on ranks have been used in many studies with high dimensional

[1]For simplicity of description, we refer to a variable as a level of a factor. For example, different genes from a microarray data are viewed as different factor levels of a factor.

data. See Brownie and Boos [14], Kerr et al. [53] and Wu et al. [115] for examples.

In HDLSS data problems, differences among factor levels can be reflected in many different ways. The observations from different factors may have different mean values or the replicated data may have different variances. In this thesis, the problem of clustering on observations from all factor levels is stated as the problem of detecting a significant difference on the distribution of the observations from each factor level. Let X_{ij} denote the j^{th} observation from the i^{th} factor level where $\{X_{ij}, 1 \leq j \leq n_i\}$ are independent observations from some unknown distribution $F_i(x)$, $i = 1, 2, \ldots, a$. We first test to see if these observations are from the same distribution, that we test

$$H_0 : F_1(x) = \ldots = F_a(x). \tag{3.1}$$

Classical ANOVA tests whether the means of observations from each factor level are the same. However, these methods require error terms to be i.i.d. normal and with a constant variance (homocedasticity). The Kruskal-Wallis test can be used where the data are not normal by computing th usual analysis of variance test statistic from the ranks, rather than on the original observations. This test can be applied when the number of treatments is small; but the test is not valid in a high dimensional setting since the inference is based on large sample size and small number of distributions. Akritas and Arnold [2] showed that the ANOVA F test is robust to departure from homocedasticity when there is a large number of factors, but it is not asymptotically valid for unbalanced data with small sample sizes, even under homocedasticity. Later, Akritas and Papadatos [3] considered test procedures for unbalanced and/or heterocedastic situations when the number of factors tends to infinity. However, their test still requires that the number of replications goes to infinity suitably fast. Finally, trying to solve all such limitations, Wang and Akritas [112] considered a nonparametric rank test of the null hypothesis of equality of distribution functions for each factor level when the number of factors is large and the number of replications is either small[2] or large. The use of Wang and Akritas [112] procedure in the new clustering algorithm gives

[2]This is the situation defined here as HDLSS data.

a large amount of flexibility considering balanced and unbalanced data, homocedastic or heterocedastic cases, small or large sample sizes and non-normality of error terms.

3.1 The Nonparametric Test of No Group Effect

This section explains the nonparametric test that allows one to detect the effect of the group factor by considering a large number of factor levels simultaneously. The p-value produced from such a test provides a measure of homogeneity among the levels considered in the test.

First, let X_{ij} denote the j^{th} observation from the i^{th} factor, where $\{X_{ij}, 1 \leq j \leq n_i\}$ are independent observations from some unknown distribution $F_i(x)$, $i = 1, 2, \ldots, a$. When the distributions from all factor levels are the same, all observations are i.i.d. realizations of a common distribution. A matrix of elements X_{ij} with rows representing factors, $i = 1, \ldots, a$, and columns representing observations (replications), $j = 1, \ldots, n_i$ is shown in Table 3.1.

Table 3.1. *High dimensional data layout, where $a \to \infty$ and $n_i \geq 2$.*

Factor Level	Observations				Sample size
1	X_{11}	X_{12}	$\ldots$	X_{1n_1}	n_1
2	X_{21}	X_{22}	$\ldots$	X_{2n_2}	n_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
a	X_{r1}	X_{r2}	$\ldots$	X_{rn_r}	n_a

Let R_{ij} represent the rank of observation X_{ij} in the set of all $n_1+n_2+\ldots+n_a$ observations. Then, under H_0, these ranks are discrete uniformly distributed random numbers between 1 and $\sum_{i=1}^{a} n_i$. Now, define the test statistic,

$$F_R = \frac{MST_R}{MSE_R} \tag{3.2}$$

where MST_R is the mean square error due to factor levels, calculated over ranks. That is,

$$MST_R = \frac{1}{a-1} \sum_{i=1}^{a} (\bar{R}_{i\cdot} - \tilde{R}_{\cdot\cdot})^2, \tag{3.3}$$

and MSE_R is the estimate of sample variance, also obtained over ranks, or

$$MSE_R = \frac{1}{a}\sum_{i=1}^{a}\frac{1}{n_i}S^2_{R,i}. \tag{3.4}$$

Note that $\bar{R}_{i\cdot} = n_i^{-1}\sum_{j=1}^{n_i} R_{ij}$ is the mean rank of each factor level, $\tilde{R}_{\cdot\cdot} = a^{-1}\sum_{i=1}^{a}\bar{R}_{i\cdot}$ is the overall mean of factor levels, and $S^2_{R,i}$ is the sample variance calculated for each factor.

The following theorem, by Wang and Akritas [112], uses the test statistic defined in (3.1) to give the asymptotic distribution of $\sqrt{a}(F_R - 1)$ under H_0 in HDLSS data where F_R is the test statistic defined in (3.2).

Theorem 3.1. *Let $H_0 : F_1(x) = \ldots = F_a(x)$ be satisfied, with $F_i(x)$ arbitrary . If $n_i \geq 2$ fixed, assuming the observations are independent, the following limits exist*

$$v_2^2 = \lim_{a\to\infty}\frac{1}{a}\sum_{i=1}^{a}\frac{1}{n_i}\sigma_i^2 > 0$$

and

$$\tau_2 = \lim_{a\to\infty}\frac{1}{a}\sum_{i=1}^{a}\frac{2\sigma_i^4}{n_i(n_i-1)}.$$

Then, as $a \to \infty$,

$$\sqrt{a}(F_R - 1) \xrightarrow{d} N(0, \tau_2/v_2^4). \tag{3.5}$$

The statistic $\sqrt{a}(F_R - 1)$ can be used to obtain a p-value for the test and compare it with some specified significance level α, such that the p-value works as a similarity measure in the algorithm. In this way, large p-values indicate the factor levels being tested are similar in distribution, and factors levels belong to the same group. In contrast, a small p-value gives evidence against H_0 indicating that at least two groups of factors are required.

The performance of the test is verified here through analysis of Type I error rates in simulated data.

3.1.1 Type I Error Rate

Wang and Akritas [112] reported simulation studies in two-way ANOVA showing that their test statistic has a stable Type I error rate and the test works even when the error distribution is not normal. A numerical study on performance of the test statistic under the one

way model has not been reported in literature. Table 3.2 reports the approximate Type I error rates using the asymptotic distribution of the test statistic in 3.5 at levels 0.10, 0.05 and 0.01. For performance of other nonparmetric tests in such a setting, one should see Akritas and Papadatos [3]. In the simulations a takes on the values 1000, 2000 and 4000, and the number of observations per factor level is set to 4. The simulations are based on 2000 replications and observations were generated from normal, lognormal, exponential, and Cauchy distributions. The jackknife estimators[3] of σ_i^4 were used in the estimation of the asymptotic variances from (3.5).

Table 3.2. *Estimated levels for one-way ANOVA*

		Nominal levels		
	Number of	0.10	0.05	0.01
Distribution	Factors	Type I errors		
Normal(0,1)	1000	0.0965	0.0500	0.0130
	2000	0.1060	0.0620	0.0110
	4000	0.1075	0.0535	0.0135
Lognormal(0,1)	1000	0.1040	0.0605	0.0155
	2000	0.1150	0.0610	0.0140
	4000	0.1140	0.0620	0.0170
Exponential(1)	1000	0.1160	0.0625	0.0130
	2000	0.1100	0.0585	0.0130
	4000	0.1135	0.0605	0.0155
Cauchy(0,1)	1000	0.1105	0.0545	0.0165
	2000	0.1005	0.0555	0.0150
	4000	0.1100	0.0625	0.0105

The Type I error rates reported in Table 3.2 are close to the true α levels, indicating that the test statistic $\sqrt{a}(F_r - 1)$ performs well in testing the hypothesis in (3.1) regardless

[3]Let $(x_1, \ldots, x_n)$ be a random sample, $\sigma_n^4 = \left(\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n}\right)^2$, and $\sigma_{n(i)}^4$, x_i, $i = 1, \ldots, n$, be calculated as σ_n^4, excepted that x_i is removed from the sample. The Jackknife estimator of σ^4, $\hat{\sigma}_{jack}^4$ is given by,

$$\hat{\sigma}_{jack}^4 = n\sigma_n^4 - \frac{n-1}{n}\sum_{i=1}^n \sigma_{n(i)}^4.$$

This estimator has a smaller bias than σ_n^4. See Pawitan [79].

of whether the distribution is symmetric (normal), or skewed (lognormal, exponential), or heavy tailed (Cauchy). These results are expected because according to Theorem 3.5 the only assumption required is that observations are independent and from any arbitrary distribution $F_i(x)$.

Once a measure of dissimilarity for the clustering algorithm is established, a new clustering algorithm can be developed through a partitioning of the factor levels into groups until the homogeneity measure of each group is above a given threshold. Section 3.2 will introduce the details of the new partitioning algorithm.

3.1.2 Power Curves

To study the power of test given in Theorem 3.5, datasets were generated with 2000 factor levels and 4 observations for each factor level. Simulations considered random values of one of the following distributions: Normal$(0, 1)$, Lognormal$(0, 1)$, Exponential(1) and Cauchy$(0, 1)$. In each case 1900 values were generated from one of the distributions and 100 values from the same distribution had the mean shifted for some value d. The Type I error rate was $\alpha = 0.05$. Figure 3.1 shows the power curves for each case. It is possible to note that the power increases faster for asymmetric distributions (Exponential and Lognormal) and the test is less powerfull in the case of symmetric heavy tailed distribution (Cauchy). Overall PPCLUST showed to be very powerful in detecting small differences in all cases.

3.2 Partition Clustering Algorithm Based on p-Values

The p-values obtained from the test in (3.1), using the statistic F_R in (3.2), can serve as similarity measure in a partitional clustering algorithm with high dimensional data having the structure shown in Table 3.1. The algorithm introduced in this thesis, called PPCLUST (p-values based Partitional Clustering), iteratively tests to see if each group that contains multiple factor levels has similarity, i.e. homogeneity measure above a given threshold. A partition occurs whenever the similarity is below the threshold. That is, a group is

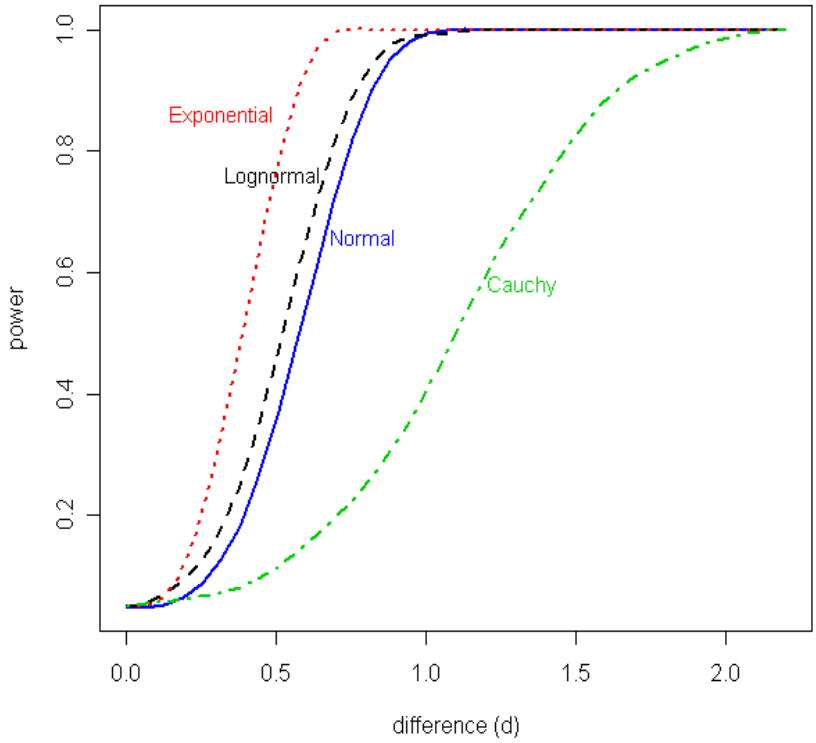

Figure 3.1. *Achieved power for HDLSS data with* $\alpha = 0.05$*, considering shifted differences in mean (d) in a group of* 100 *factor levels in a total of* 2000 *factor levels and data generated from four distributions: Normal*$(0, 1)$ *(continuous line in blue), Lognormal*$(0, 1)$ *(dashed line in black), Exponential*(1) *(dotted line in red) and Cauchy*$(0, 1)$ *(dotted-dashed line in green).*

partitioned into two smaller groups when the test for H_0 in (3.1) is rejected and the group remains intact if the test is not rejected. When H_0 is rejected, smaller groups of factor levels are created and tested. The algorithm stops when there are no groups with similarity measures below the threshold. Factor levels that cannot be assigned to any of the created groups are labeled as group 0 in PPCLUST. This means that factor levels with group label equal to 0 are not necessarily similar (or dissimilar), but they only did not belong to any other created group according to PPCLUST, i.e., factor levels in group 0 resulted in a rejection

of H_0 when tested with factor levels of any other created group. Groups labeled with lower values in the output usually contain factor levels with lower observation values than those in groups labeled with higher numbers. In microarray expression data this means, for example, that lower groups are formed by low expressed genes, high groups by high expressed genes and intermediate groups by genes not expressed.

Let g stand for a current group label that the test is to be applied, $D1$ contains the observations in the current working group g and nf is the number of factors in $D1$. PPCLUST is described below in 8 steps, and a detailed block diagram of the algorithm is represented in Fig. 3.2 (see end of this chapter).

Step 1: Set $g = 1$, $D1$ = Data.

Step 2: Rank all data in $D1$.

Step 3: Calculate the median rank for each factor in $D1$.

Step 4: Sort factors in $D1$ by median ranks.

Step 5: Test $D1$.

If H_0 is not rejected: finish.

If H_0 is rejected: continue.

Step 6: Take a subset of $D1$ and call $D2$.

Step 7: Calculate number of factors in $D2$ and give the nf of $D2$.

If $nf = 1$:

Allocate factor in $D2$ to group 0.

Remove factors in $D2$ from $D1$.

If nf in$D1 = 0$ then finish.

If nf in $D1 > 0$, then make $D2 = D1$ and return to Step 7.

Step 8: Test D2.

If H_0 is not rejected:

Assign factors of D2 to group g.

Make $g = g + 1$.

Remove factors in D2 from D1.

If nf in $D1 = 0$ then finish.

If nf in $D1 > 0$ then do:

Test to see if each factor of $D1$ belongs to the new assigned group by testing the hypothesis that the observations in all factor levels have the same distribution. Remove the factor from $D1$ when H_0 is not rejected and put it into the newly assigned group.

Make $D2 = D1$ for the remaining factors in $D1$ and return to Step 7.

If H_0 is rejected:

Take a subset of $D2$ and call it $D3$.

Return to $D1$ all factors that are not in $D3$.

Make $D2 = D3$ and delete $D3$.

Return to Step 7.

The accuracy and stability of PPCLUST is explored in Chapter 4 through simulations that reproduce the basic structure of microarrays in terms of the distribution of gene expression levels. PPCLUST is compared with some benchmark algorithms found in literature and the properties of PPCLUST are listed in the end of the chapter. Also, in Chapter 5, two examples with gene expression data are explored with the new algorithm.

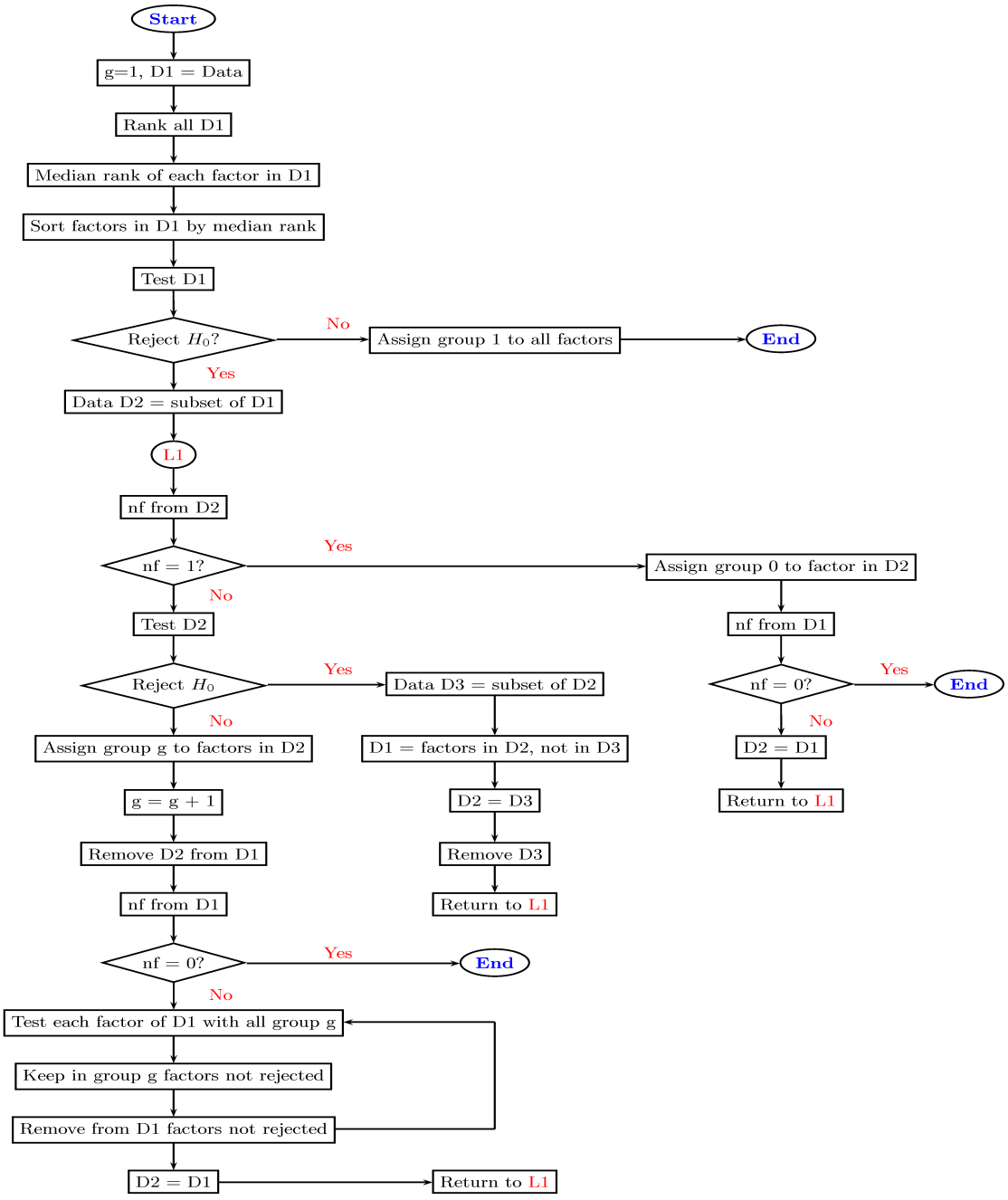

Figure 3.2. *PPCLUST block diagram. In the diagram nf stands for "number of factors" and g for "group label". Group 0 is reserved to factors that cannot be allocated to any created group.*

4

Simulations and Properties of PPCLUST in Replicated Data

When performing clustering on simulated data, groups of factor levels can be generated from known distributions and the quality of a clustering algorithm can be evaluated by using some measure that allows one to compare generated groups with the groups obtained by a clustering algorithm. In this chapter, simulations are performed to verify the quality of PPCLUST in grouping factors, and also to compare PPCLUST with some benchmark algorithms found in microarray clustering literature. The quality of each algorithm is obtained through the Adjusted Rand Index (ARI) described in Section 2.3.

4.1 Clustering Simulations with HDLSS Data

When clustering gene expression data there is no guarantee that an algorithm worked successfully unless further biological study about functions of genes is performed for each group of genes to see if the obtained groups are meaningful. For this reason, simulated data that represents real data having previously known group structures are used to test clustering algorithms. In this way, the performance of PPCLUST on real gene expression data can be inferred from its performance on the simulated data. Generate simulated data to closely

reproduce real gene expression data alone is a challenging topic as the nature of such data is not well understood (see e.g., Grant et al. [40], Lönnstedt and Speed [61], Purdom and Holmes [82], Nykter et al. [73], Albers et al. [4], and Shaik and Yeasin [93]). The data involves static or dynamic state. Examples that model such data include stochastic differential equations (Mendes et al. [70]), Bayesian networks (Friedman et al. [32]), relevance networks (D'haeseleer et al. [21]; Butte and Kohane [15]), and graphical models (Kishino and Waddell [56]; de la Fuente et al. [19]; Magwene and Kim [63]).

In the following simulations we only try to simulate the basic structure observed in some colorectal data studied by Notterman et al. [72]. The objective is to reproduce the overall distribution of genes observed in large groups of similarly expressed genes. The groups are obtained using both PPCLUST and some other standard algorithms. It is noted that the overall distribution in large groups is similar to a t-distribution shifted by some location parameter μ and stretched by a scale parameter σ. Figure 4.1 compares the density functions of three groups of genes obtained from colorectal data and three simulated groups generated from t-distributions with 15 degrees of freedom shifted by -0.25, 0 and 0.2, and stretched by 0.25. The simulated distributions based on original data satisfy the objectives of clustering comparisons intended here and were adopted in the simulation studies.

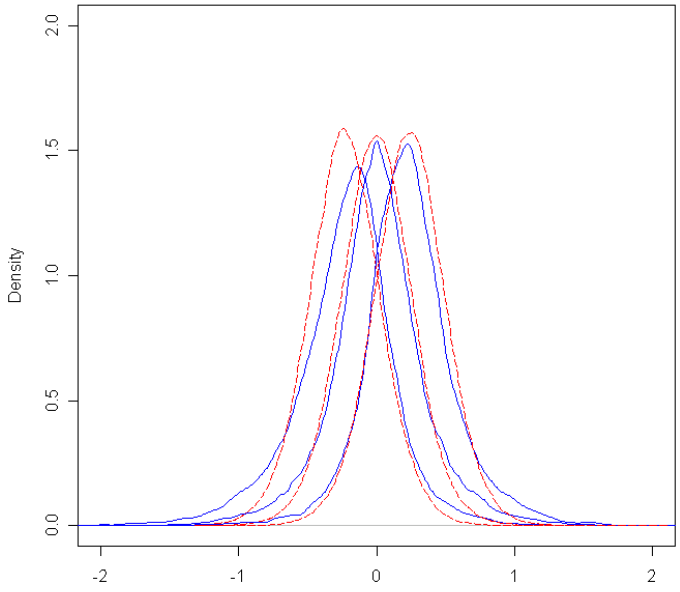

Figure 4.1. *Original (continuous blue line) and simulated (dashed red line) density functions for three groups of genes in colorectal expression data .*

4.1.1 Study I: Symmetric Groups

Once the overall structure of the colorectal data was observed, the simulation study was performed considering 200 datasets, with 4000 factor levels each. The data was divided into 5 groups of factors from *location-scale families* with standard pdf from a t-distribution with 15 degrees of freedom (t_{15}), according to the following scheme:

- Group 1: 300 factor levels equal to $0.25 \times t_{15} - 0.5$.
- Group 2: 200 factor levels equal to $0.25 \times t_{15} - 0.2$.
- Group 3: 2500 factor levels equal to $0.25 \times t_{15}$.
- Group 4: 800 factor levels equal to $0.25 \times t_{15} + 0.5$.
- Group 5: 200 factor levels equal to $0.25 \times t_{15} + 1$.

In this case, the data for each group comes from symmetric distributions, as can be observed in Figs. 4.2 and 4.3, where the sample size for each factor level was set equal to 5.

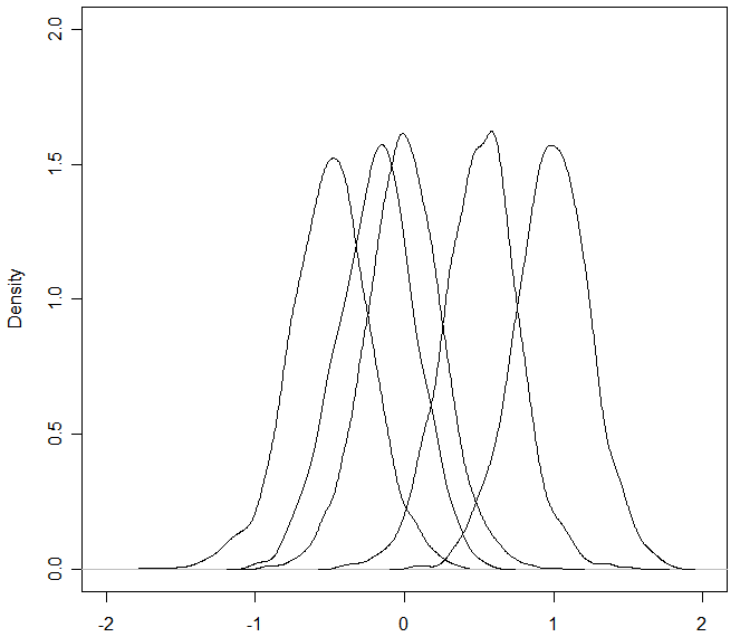

Figure 4.2. *Study I: Density functions used to generate groups of data.*

After the data were generated, PPCLUST[1] and the following 10 benchmark clustering algorithms are applied to the data: PAM, K-means, Energy (with Euclidean norm $||x-y||$), Mclust (with automatic choice of best model), CLARA (with Euclidean metric), DIANA (with Euclidean metric), HCLUST (with Ward's agglomeration method), AGNES, FANNY and SOM (with dimension 5×1). ARI was used to compare the performance of these algorithms.

In all algorithms that need pre-specification of the number of clusters, the number is set to be 5, the real number of groups. *R* software (version 2.4.1) with packages energy, mclust, cluster, and SOM were used. PPCLUST was written in SAS© macro language (version 9.3.1), and the Adjusted Rand Index was calculated using both R and SAS© [2].

[1]PPCLUST used significance level $\alpha = 10^{-8}$.

[2]The simulations and all calculations were performed using Windows XP with Intel Pentium M processor, 1.6GHz, and 1Gb of RAM memory.

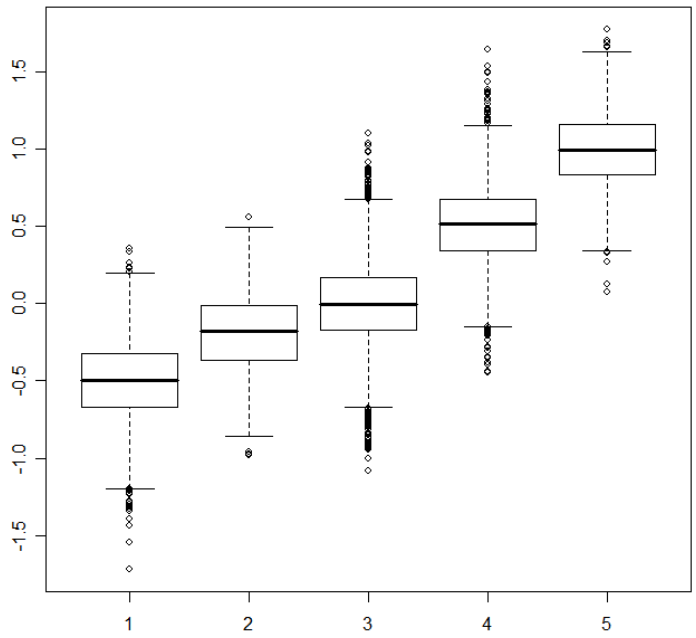

Figure 4.3. *Study I: Boxplots of generated data for each group.*

In order to verify the performance of PPCLUST under different sample sizes, the complete simulation study was repeated considering also samples of sizes 10, 15, and 20.

4.1.2 Study II: Asymmetric Groups

In a second study, the data generated for study I was transformed using the equation $\exp((x + 1) * 4)$, where x is a value generated in study I. The resulting distribution of the data is like a lognormal but with more extreme points since x was generated from t-distribution and not a normal distribution. The resulting groups are presented in Fig. 4.4. Note that the distributions now are asymmetric with outliers and some overlap in many points.

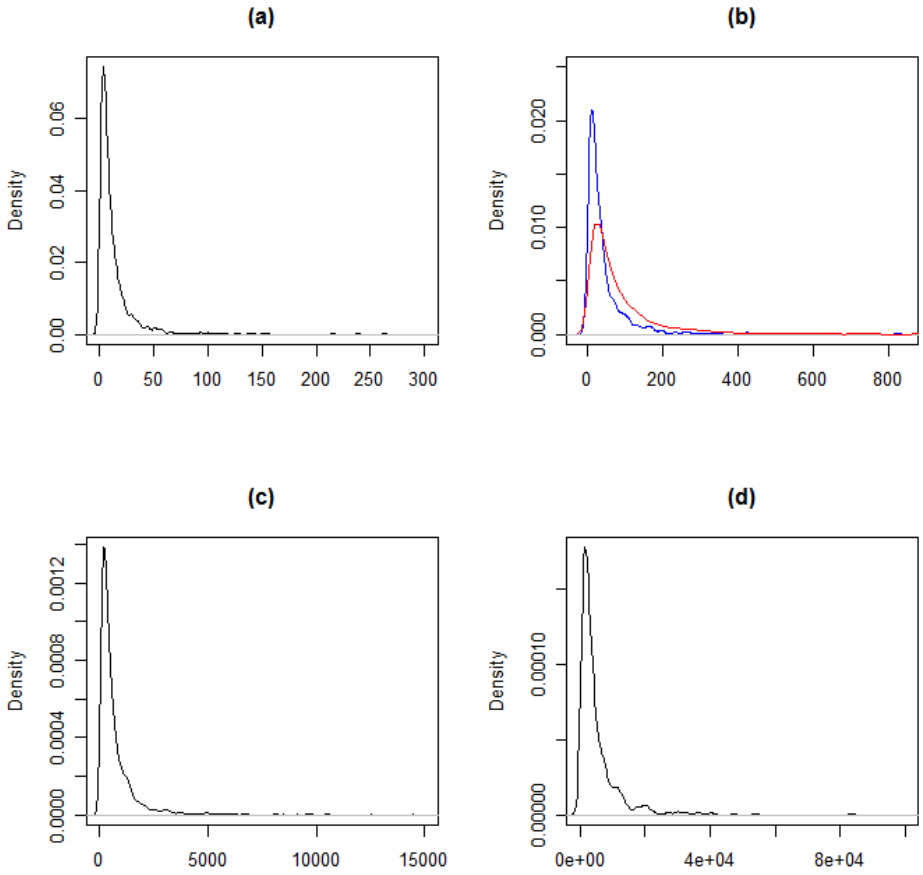

Figure 4.4. *Study II: Density plots for simulated groups, where (a) shows group 1, (b) shows groups 2 (blue tall curve) and 3 (red short curve), (c) group 4, and (d) group 5. Groups separated into 4 plots due to large difference in their respective ranges.*

4.1.3 Results

Table 4.1 reports the mean and standard deviations of ARI for PPCLUST and each of the 10 other algorithms based on 200 simulations for different sample sizes, when groups are simulated from symmetric distributions. The two best (highest means) and the two most stable (lowest standard deviations) algorithms under each sample size are indicated in bold face fonts. In terms of quality of adjustment, i.e., grouping together the correct factor

Table 4.1. *Mean and standard deviations (std) of Adjusted Rand Index for PPCLUST and* 10 *other algorithms, over* 200 *simulated datasets and considering different sample sizes. Generated groups from symmetric distributions (Study I).*

Adjusted Rand Index								
	Sample Sizes							
	5		10		15		20	
Algorithm	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
PPCLUST	**0.8600**	**0.0104**	**0.9205**	**0.0110**	**0.9541**	0.0084	**0.9688**	0.0059
PAM	0.4709	0.0323	0.5342	0.0166	0.5352	0.0135	0.5346	0.0121
K-means	0.4652	0.0416	0.4938	0.1585	0.5476	0.1697	0.5407	0.1362
Energy	0.4846	0.0469	0.6456	0.1149	0.8562	0.0851	0.9184	0.0262
Mclust	**0.6184**	0.1163	**0.8826**	0.0894	**0.9425**	**0.0056**	**0.9613**	**0.0043**
Clara	0.4219	0.0927	0.5315	0.0656	0.5516	0.0565	0.5527	0.0560
Diana	0.5662	0.1477	0.6182	0.1387	0.6401	0.1151	0.7145	0.1052
HCLUST	0.4846	0.0469	0.6456	0.1149	0.8559	0.0852	0.9184	0.0262
Agnes	0.4963	0.0557	0.6915	0.1193	0.8551	0.0841	0.9150	0.0332
Fanny	0.3820	0.1368	0.6122	0.0431	0.6067	0.0378	0.5856	0.0516
SOM	0.2920	**0.0048**	0.3750	**0.0057**	0.4280	**0.0059**	0.4622	**0.0045**

levels, PPCLUST is the best for all sample sizes. The stability of PPCLUST is also the best for small sample sizes (5 and 10) with the algorithm very stable in large sample sizes (15 and 20). Stability is very important because it means that the algorithm will not produce considerably different results under similar data structures.

Among the compared algorithms, MCLUST achieved ARI superior to 85% for samples of sizes greater than 5. This makes MCLUST a good alternative to PPCLUST in such situations. SOM showed consistent stability but with very low quality of grouping, since ARI mean values were less than 0.5 for all sample sizes. Algorithms Energy, HCLUST, and Agnes, are competitive methods to PPCLUST for samples of size 15 or higher, but those algorithms are not as stable as PPCLUST and MCLUST. Diana and Fanny showed the lowest stability of all algorithms and should not be used with HDLSS data. Figure 4.5 shows a visual comparison of the algorithms through the use of boxplots. Overall, PPCLUST produced the best results as it is very stable and has a high capacity of correctly allocating

factor levels to groups. For larger samples, MCLUST is a good alternative to PPCLUST.

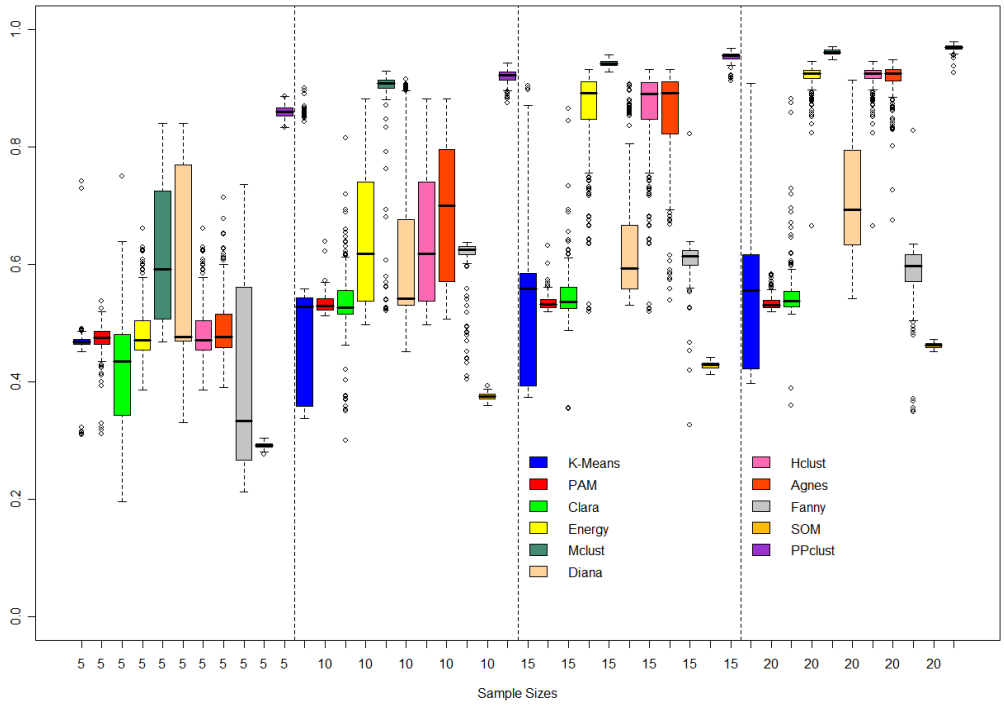

Figure 4.5. *Boxplots of Adjusted Rand Index for PPCLUST and* 10 *other algorithms, over* 200 *simulated datasets and considering different sample sizes. Generated groups from symmetric distributions (Study I).*

Additionally, Table 4.2 and Fig. 4.6 consolidate the results obtained when simulated groups have asymmetric distributions. In this case it is clear that PPCLUST is considerably better than all other algorithms in all sample size situations. Clara gave the worst results and PAM was the best algorithm among the one's compared with PPCLUST, but never had ARI higher than 0.65 for all sample sizes. PPCLUST presented the same results as in simulations of groups with symmetric distributions, because it is invariant to monotone transformations.

In microarray data it is usually recommended to log transform the data before its analysis. The previous results seem to agree with this strategy, if the data is really asymmetrical.

Table 4.2. *Mean and standard deviations (std) of Adjusted Rand Index for PPCLUST and* 10 *other algorithms, over* 200 *simulated datasets and considering different sample sizes. Generated groups from asymmetric distributions (Study II).*

Adjusted Rand Index								
	Sample Sizes							
	5		10		15		20	
Algorithm	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
PPCLUST	**0.8600**	**0.0104**	**0.9205**	0.0110	**0.9541**	**0.0084**	**0.9688**	0.0059
PAM	**0.5396**	0.0257	**0.6073**	0.0218	**0.6347**	0.0178	**0.6454**	0.0377
K-means	0.1321	0.0192	0.2670	0.1524	0.5194	0.1164	0.5700	0.1410
Energy	0.4923	0.0370	0.5905	0.0184	0.6213	0.0114	0.6318	0.0097
Mclust	0.3495	0.0547	0.4126	0.0372	0.4656	0.0813	0.5028	0.1040
Clara	0.3860	0.1515	0.3696	0.0213	0.3256	0.2134	0.3390	0.2171
Diana	0.0130	0.0079	0.0066	**0.0037**	0.0045	**0.0019**	0.0039	**0.0013**
HCLUST	0.4923	0.0370	0.5905	0.0184	0.6211	0.0114	0.6318	0.0097
Agnes	0.1420	0.0320	0.1553	0.0181	0.1580	0.0052	0.1579	**0.0036**
Fanny	0.2990	0.0250	0.4236	0.0375	0.4716	0.0457	0.5221	0.0235
SOM	0.4766	**0.0112**	0.5412	**0.0083**	0.5775	0.0087	0.5999	0.0060

PPCLUST has the advantage of being invariant to transformations with high quality of grouping whether the data are symmetrical or not. PPCLUST is very stable in any situation and does not need a previous specification of the unknown number of groups, as required by other algorithms. The specification of the significance level α in PPCLUST allows a fine tunning of the number of groups to be obtained. Too many small groups with intermediate values means that α should be increased, extreme groups in data means that α should be decreased.

An additional investigation was made to compare the processing times of the algorithms. Table 4.3 presents compilation times among algorithms with best results in simulations with symmetric groups. PPCLUST processing time was always less than 1 minute for each run on 4000 levels and PAM was the only faster algorithm. MCLUST, the closer competitor to PPCLUST, showed processing times at least 3 times higher than PPCLUST.

Finally, a study about efficiency was made when data in a row (or column) is stan-

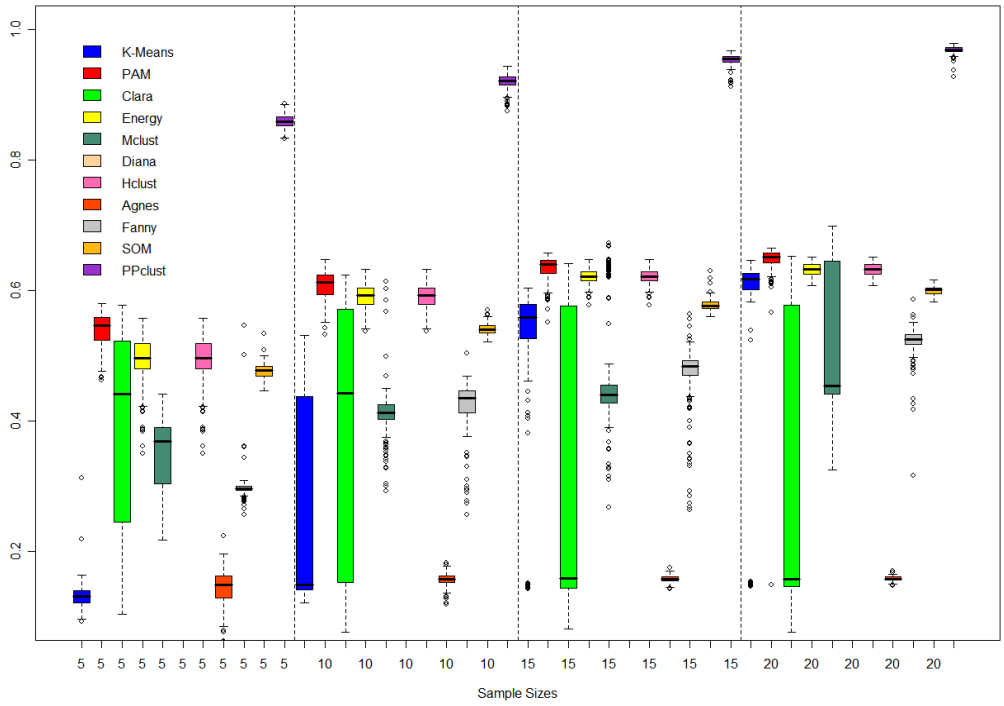

Figure 4.6. *Boxplots of Adjusted Rand Index for PPCLUST and* 10 *other algorithms, over* 200 *simulated datasets and considering different sample sizes. Generated groups from asymmetric distributions (Study II).*

dardized. Data standardization was shown to be a bad strategy in all situations even for PPCLUST, with ARI very small or around 0, meaning that in this case factors are randomly allocated to groups. That is, there is no pattern in allocation of elements to groups when data are standardized and standardization is not recommended when clustering HDLSS data.

Table 4.3. *Compilation time, in seconds, for PPCLUST and 4 other algorithms: PAM, MCLUST, HCLUST, and Energy. Datasets with 4000 factors and equal sample sizes.*

		Compilation Time for Sample Sizes			
Algorithm	Description	5	10	15	20
PPCLUST	Partition clustering based on p-values	18	30	43	54
PAM	Partitioning around medoids	24	12	13	13
MCLUST	Model based clustering	135	151	141	158
HCLUST	Hierarchical clustering	276	274	275	277
Energy	Hierarchical clustering by minimum energy	321	306	309	312

4.2 Properties of PPCLUST

A close inspection of the PPCLUST algorithm allows one to detect some of its properties. Others are revealed only with simulations performed in this chapter. Interesting properties that make PPCLUST appealing for use with HDLSS data in real applications are listed bellow.

1. **Invariance to monotone transformations** – the use of overall ranks observations in the test statistic leads to similarity measure that is invariant to monotone transformation of data and this in turn makes PPCLUST to have such property. In microarray studies it is common to have data log transformed before analysis. Many clustering algorithms can produce different results before and after monotone transformations on the same data. This is because monotone transformations change distances between variables and therefore modify similarity matrices used in clustering. PPCLUST does not have this drawback.

2. **Automatic specification of number of groups** – PPCLUST does not require the number of clusters to be specified in advance. It will determine the number of clusters automatically by specification of significance levels as a threshold that will be com-

pared with the p-values for testing the hypothesis of no group effect. The specification of significance levels is not as intrusive as the specification of number of groups, which is one of the objectives of clustering analysis. Actually, significance levels can be used as a guidance in finding the number of groups in a real dataset. For example, decreasing the significance level in PPCLUST will decrease the number of groups found because it decreases the Type I error committed by the test[3]. The use of different significance levels can serve as a fine tuning parameter in revealing the number of groups g where the algorithm tends to stabilize, i.e, find g that is more common to different α levels. This can be used as an indication of the true number of groups in the data. As an example, Table 4.4 show the number of groups and factor levels in each group for different significance levels used in one real study. Note that for very small significance levels there is no modification in group sizes and the number of groups is smaller than for higher significance levels.

Table 4.4. *Number of groups and group sizes for different thresholds (α levels) in a real data example.*

	thresholds (α)				
Group	10^{-1}	10^{-5}	10^{-10}	10^{-20}	10^{-30}
0	5	4	4	1	1
1	8	98	91	89	89
2	124	776	774	728	728
3	4	6	9	2726	2726
4	1007	2614	2673	9	9
5	23	13	5	661	661
6	2438	9	655	20	20
7	7	675	23	-	-
8	574	39	-	-	-
9	44	-	-	-	-

[3]Actually, lower significance levels will also decrease the power of the test in finding new and small groups.

3. **High quality in grouping factors** – in simulation studies the Adjusted Rand Index is used to evaluate the quality of clustering. This measure was introduced by Hubert and Arabie [46] and is described in Chapter 2. The simulations revealed that the algorithm has an outstanding performance in correctly grouping factors and outperforms all the algorithms used in comparison with PPCLUST in both clustering accuracy and stability.

4. **No multiple comparison problems** – control of global errors in multiple testing is one of the challenges that statisticians one concerned about with HDLSS data analysis (Sabatti[87]). This is less of a problem in PPCLUST since the test is applied to groups of variables instead of on a one-by-one basis.

5. **No need for dimension reduction** – in high-dimensional studies it is common to apply some dimension-reduction technique such as principal components analysis before clustering data (see PCA in Johnson [50]). Studies, as in Yeung and Ruzzo [118], do not recommend the use of PCA before clustering, except in very special situations. Simulations in Yeung and Ruzzo show that clustering principal components instead of original data produce different results on many algorithms using different similarity metrics. PPCLUST does not require previous dimension reduction to the analysis. Instead, PPCLUST relies on the high dimension to provide power to give good similarity measure. This is specially appealing with only very small number of replications.

6. **No requirement of balanced data** – the algorithm works with both balanced or unbalanced data. The only requirement is that the number of replications for a factor be at least 2. There is no need that all factor levels have the same number of replications. Unbalanced data is common in studies of microarray gene expression data and some algorithms require balanced data. Solutions like elimination from the study of factor levels with incomplete information or imputation of data can hide or seriously compromise the result of the study.

7. **Fast and easy to use** – in simulations, PPCLUST took less than a minute to complete the clustering of datasets. The performance was observed with different sample sizes (5 to 20 observations) and number of factor levels up to 7000[4]. Additionally, the algorithm is very easy to use, since PPCLUST was written in SAS© macro language using a friendly screen for specification of necessary information, like dataset name, observations and significance level of each test[5].

8. **No memory allocation problems** – Since PPCLUST is developed using SAS© macro language, it takes advantadge of SAS memory allocation and does not have problem in handling as many factor levels as necessary.

[4]Performance observed with PC machine running Windows XP with Intel Pentium M processor, 1.6Ghz, and 1Gb of RAM memory.

[5]Instructions in Appendix B.

5

Applications of PPCLUST in Microarray Data

Clustering of HDLSS data has been studied in many areas such as chemometrics, image analysis, and agriculture. One of the major applications has been in genomics where the study of gene expression data using DNA microarrays [1] involves data about a large number of genes for a small number of biological samples. PPCLUST can be applied to microarray data analysis when one wants to cluster genes (factor levels). In microarray studies, thousand of genes are observed for a small number of tissues. The reason for clustering genes is the idea that genes that are responsible for disease progression or in their response to treatment may have differentially expressed levels. Clustering information can be used for biomarker identification in drug development. Additionally, the differentially expressed genes can be used to classify patient disease status. For example, using all genes from the whole genome can lead to inefficiency in classifying tumor patients as no methodology can deal with high dimensional prediction without posing strong assumptions. Instead, using only the genes found to be differentially expressed from the clustering algorithm can significantly reduce the complexity of the classification problem. That is, results from the clustering can serve as a

[1] See Appendix A for description of microarray technology.

dimension reduction tool for classification. These studies would allow to improve treatments by identification of targets for therapy in many diseases.

In this chapter, PPCLUST is applied to two HDLSS datasets found in the microarray literature: (1) Notterman et al. [72] study about transcriptional gene expression profiles of colorectal cancer and; (2) van't Wout et al. [110] HIV cellular gene expression data. In real data examples, graphical procedures, such as heatmaps and principal components plots, are used to verify the efficiency of PPCLUST. Heatmap is a matrix that maps the expression levels of each gene to a color intensity value.

5.1 Real Data Study I: Clustering Genes in Colorectal Cancer

Colon and rectal cancer have many features in common and for this reason both are often referred to as colorectal cancer. Figure 5.1 shows the area in the organism where colorectum cancer starts, i.e., the large intestine (colon) and rectum (end of colon). This cancer begins in most cases as a growth of tissue, called polyp, inside the wall of the colon or rectum (see Figure 5.2). If the cells of a tumor (adenomas) acquire the ability to invade and spread into the intestine and other areas a malignant tumor develops (carcinoma or adenocarcinoma).

According to the American Cancer Society [97], colorectum cancer is the fourth most diagnosed cancer in the United States and Canada. Only skin cancer, lung and breast in women and lung and prostate in men have more cases.

The Colon Cancer Alliance Group [5] estimates that about 20% of colorectal cancers are found after the disease has spread to more distant organs such as the lung, and only 37% of colorectal cancers found are confined to colon or rectum. When the cancer has spread to other sites, only 8.5% of those diagnosed will live for five-years or more. This drammatic statistic becomes worse if it is considered that the American Cancer Society estimates $112,340$ new cases of colon cancer and $41,420$ new cases of rectal cancer will be diagnosed in 2007, and that both malignancies will result in $52,180$ deaths. More de-

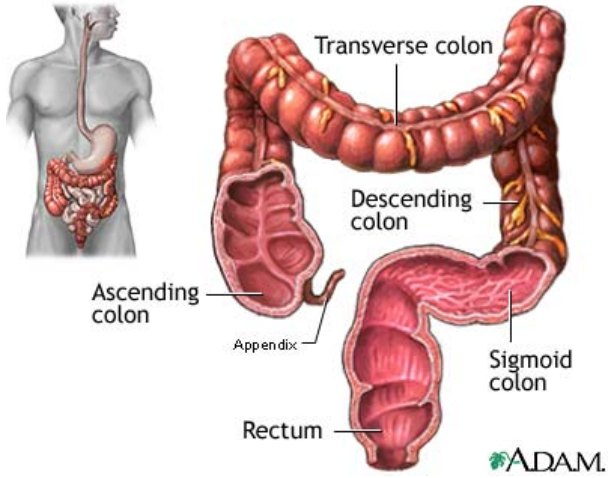

Figure 5.1. *Areas of development of colorectal cancer (nlm.nih.gov/medilineplus)*

Figure 5.2. *Polyp in colon wall (endoatlas.com)*

tailed information about colorectal cancer can be found in publications by American Cancer Society [97], Colon Cancer Alliance [5] and others [80, 17].

Understanding how change in DNA causes cells of the colon and rectum to become cancerous could guide scientists in the development of new drugs, treatments and actions during early stages of the disease.

Alon et al. [7] used a colon cancer dataset to study the clustering of tumor and nor-

mal colon tissues probed by oligonucleotide arrays. This dataset has expression levels for 40 tumor and 22 normal colon tissues for $2,000$ human genes measured using Affimetrix technology. Later, Notterman et al. [72] studied transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissues, using oligonucleotide arrays. The advantage of the Notterman et al. [72] study is that normal tissues were paired with the two types of tumors, adenoma and andenocarcinoma. The Notterman data [2] consists of mRNA expression patterns probed in 4 colon adenoma tissues, 18 adenocarcinoma and 22 paired normal colon samples. In their study, a two-way hierarchical clustering algorithm was used to show that genome-wide expression profiling may permit a molecular classification of the three different types of tissue. PPCLUST is used to cluster genes, not tissues.

Since some of the genes in the original data were observed more than once, the median of expression levels of duplicated genes in each database (adenoma and paired normal tissues database, and adenocarcinoma and paired normal tissues database) was calculated. Then the same transformations described in Notterman et al. [72] study were performed prior to the application of PPCLUST in the composite database, i.e., the following steps were applied to each dataset:

- Deletion of expression levels ≤ 0;
- Calculation of the logarithm of the expression level;
- Deletion of genes having more than 25% of their values missing. In Notterman the percentage cuttof was 15% resulting in a smaller sample.

Note that in Notterman et al. the data was also centered about the mean and normalized. However, this procedure is not recommended as was indicated by simulation studies in Section 4.1.

Two unbalanced datasets are obtained, the first one with 4 adenoma and paired normal tissues for 4175 genes and the second one with 18 adenocarcinomas and paired normal tissues

[2] Available in microarray.princeton.edu/oncology.

for 4234 genes. Only 1038 genes are common to both datasets.

The existence of paired data allows the application of PPCLUST to the difference in gene expression levels of cancer (adenoma or adenocarcinoma) and normal tissues. The idea is that genes not related to the disease should not have a change in expression levels for cancer and normal tissues. However, genes that have a significant change in expression level should produce meaningful clusters with the application of the algorithm.

PPCLUST was used with significance level of 1×10^{-10}, because it was observed that higher significance levels resulted in too many small clusters of genes. Overall, the main structure of groups was not altered using a very small significance level. After applying the algorithm, the heatmaps of original and clustered data were compared [3]. Then the plot of the first two principal components [4] was obtained to visualize the results of clustering. PCA was used to reduce the dimensionality of the problem to just two principal components and project genes in the two dimensional graph identifying the classification groups.

Figure 5.3 presents the heatmap of original differences in expression levels of adenoma and normal tissues in paired samples. There is a concentration of zero to positive expression levels for this data with no clear existence of any gene groups. After applying PPCLUST, genes were clustered into 6 groups with 38 (0.91%), 316 (7.57%), 9 (0.22%), 3573 (85.58%), 221 (5.29%), and 15 (0.36%) genes in each group. Lower groups had low expressed genes, while higher groups had the highest differences between expression levels of adenomas and paired normal tissues. There was also a set of 3 (0.07%) genes that could not be grouped with any other gene.

Figure 5.4 presents the heatmap with genes ordered by the groups to which they were allocated [5]. It can be noted that the extreme groups were formed by genes with highest

[3]Technical note 1: Usually heatmaps are presented in different shades of green to red colors. Here we used the red to blue scale since the values of expression levels are very similar. Other color scales have been used in literature, as yellow to blue, grey to blue, etc.

[4]Technical note 2: To obtain the principal components it was used the covariance matrix of the data. This way the variables were not forced to have equal variance, keeping it consistent with other procedures used. Also, if the correlation matrix was used, it could defeat the purpose of identifying those variables that contribute most to the total variability as explained in Khattree and Naik [54].

[5]Note: A longer black line on the right side of the graph indicates the start of a new group.

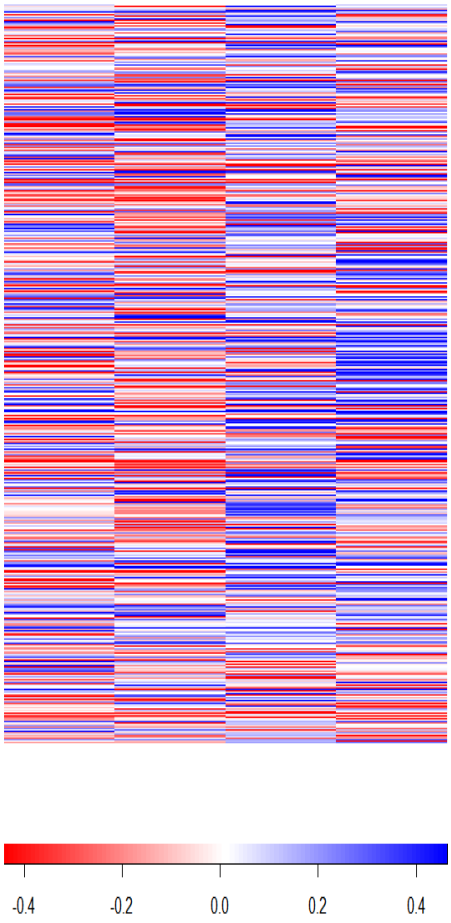

Figure 5.3. *Heatmap for Adenoma-Normal Tissues.*

differences in expression levels between adenoma and normal tissues, and group 4 was formed mostly by genes that had no change in their expression levels.

Figure 5.5 shows the plot of the first two principal components for the difference in expression levels of adenoma and paired normal tissues. Genes were labeled according to results obtained from PPCLUST giving a visual idea of the quality of clustering with PPCLUST in this dataset [6]. The first two principal components are responsible for 68.47% of the total variation in the data where 48.68% of the total variation is explained by the

[6]Due to the high number of genes, some genes are not visible in the PCA plot. Example are genes from group 3 in Fig. 5.5.

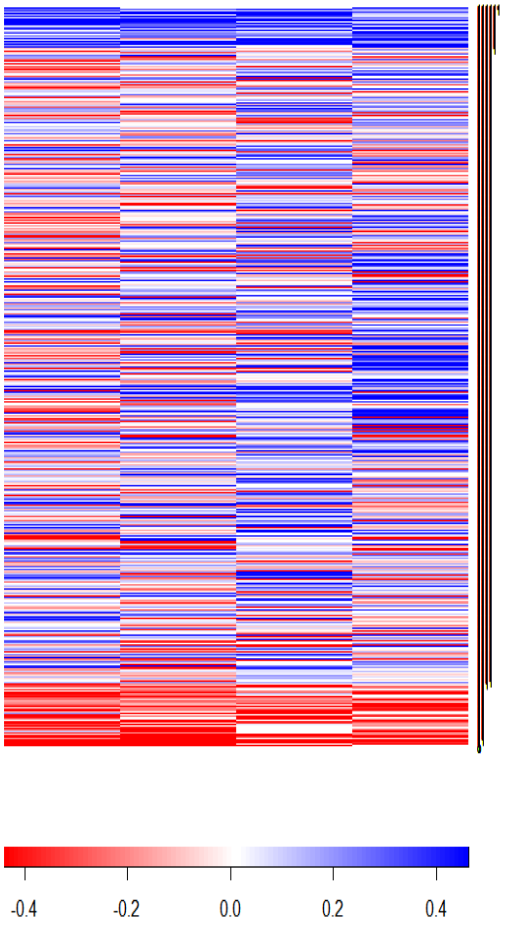

Figure 5.4. *Heatmap for Grouped Adenoma-Normal Tissues.*

first component. The algorithm seems to work in very reasonable way since it is possible to observe that groups 1 and 2 are concentrated in the lower extreme of the first dimension, groups 5 and 6 in the higher extreme of the first dimension and group 4 is distributed around value 0 of the first dimension.

The previous study was repeated considering the difference in expression levels of adenocarcinoma and normal tissues. In this case, 7 groups were obtained with only 4 (0.09%) genes not assigned to any group. The number of genes in each group were 91 (2.15%), 774 (18.28%), 9 (0.21%), 2673 (63.13%), 5 (0.12%), 655 (15.47), and 23 (0.54%). Special

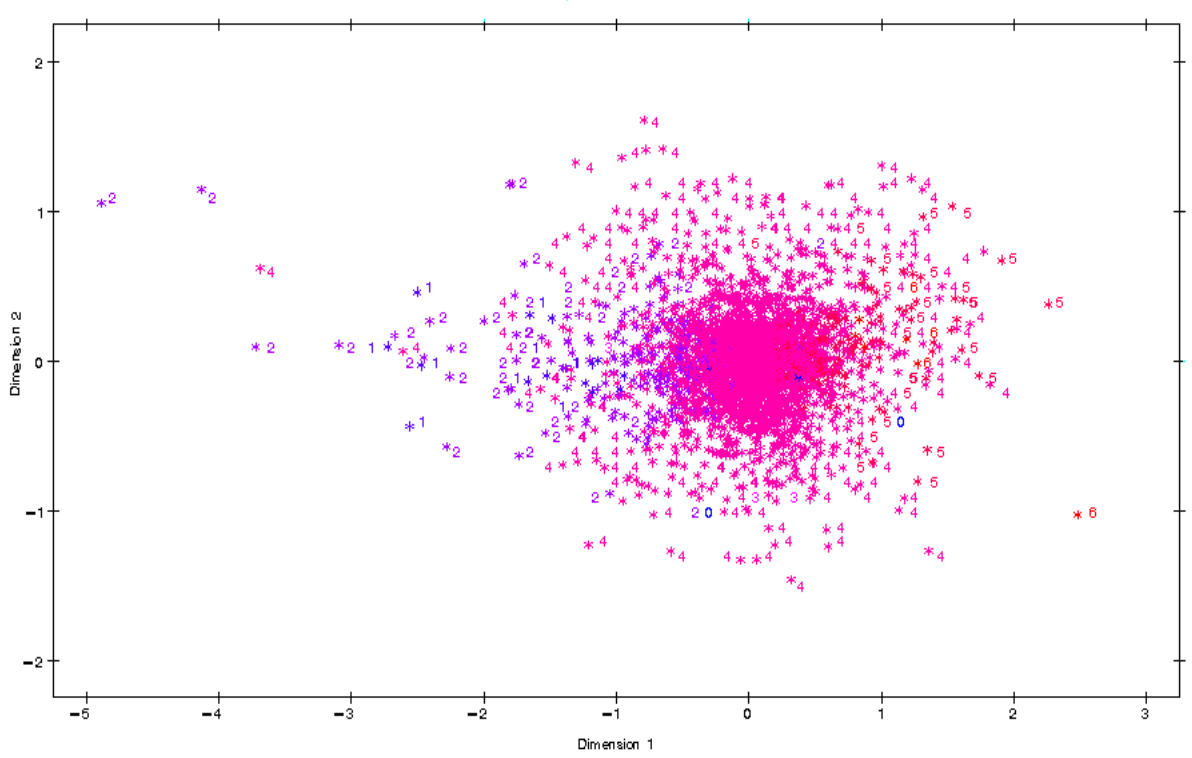

Figure 5.5. *PCA Plot of Adenoma-Normal Tissue Clusters.*

attention should be given to genes that are not in the largest group, since the largest group seems to be the group formed by genes with no difference in expression levels when comparing carcinoma tissues and normal tissues. The heatmaps before and after clustering are represented in Figs. 5.6 and 5.7. The two extreme groups indicating lower expressed values in red/white and higher expressed values in blue are more evident than other groups (2 to 5). Again, with the PCA it is possible to verify that the clustering algorithm seems to work in a very reasonable way (Fig. 5.8). The two first principal components now are responsible for 42.22% of the total variation in the data, with the first component responsible for 32.87% of the total variation and this could explain why some points apparently clustered into wrong groups.

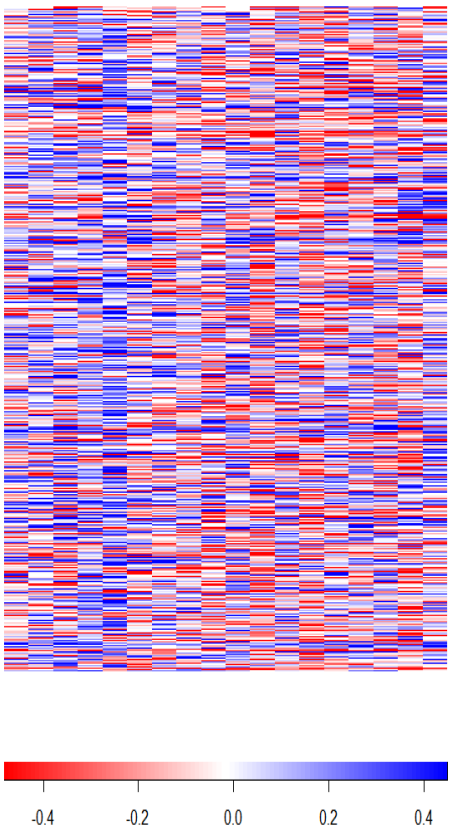

Figure 5.6. *Heatmap for Adenocarcinoma-Normal Tissues.*

Table 5.1 shows the distribution of genes among groups for both tissue types. From the 1038 genes that are present in both datasets, it was possible to observe that 558 genes had no expression in both, adenoma and adenocarcinma tissues. For the other genes not expressed in adenoma tissues, usually, there is no pattern of expression in carcinoma tissues. It could be also observed that there is a tendency of genes with low expression in adenoma tissues to be also low expressed in carcinoma tissues. The same property is also observed for highly expressed genes. Only 10 genes had opposite expression levels in both types of tissues.

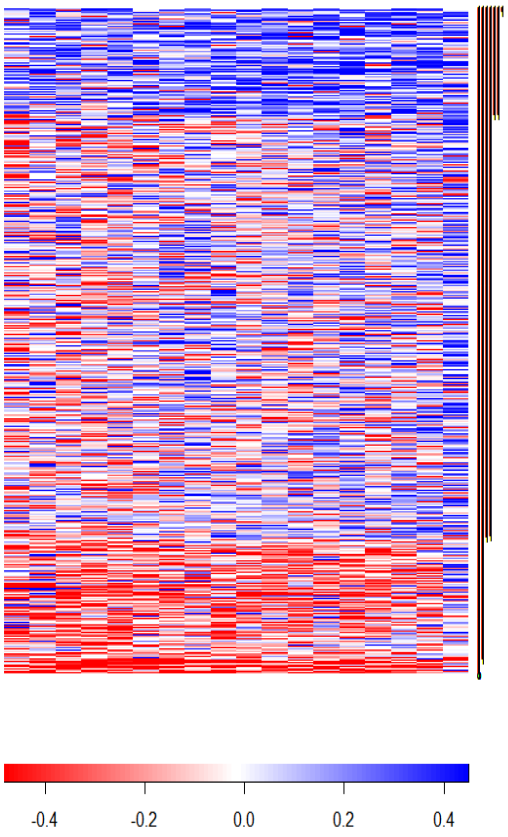

Figure 5.7. *Heatmap for Grouped Adenocarcinoma-Normal Tissues.*

Comparing the heatmaps obtained before and after clustering in both tissue types reveals that in carcinoma tissues the clustering of genes is more evident than in adenoma tissues. This is due to the larger differences in expression levels of carcinoma genes.

The clustering of genes in the colorectal cancer study suggests many interesting questions to be answered by molecular biologists. Some of them are:

1. Why do some genes have opposite expression values in benign epithelial tumor (adenomas) and cancer cells (adenocarcinomas)?

2. What are the characteristics of genes expressed in both types of tumor cells?

3. What are the characteristics of genes that are highly expressed in cancer cells?

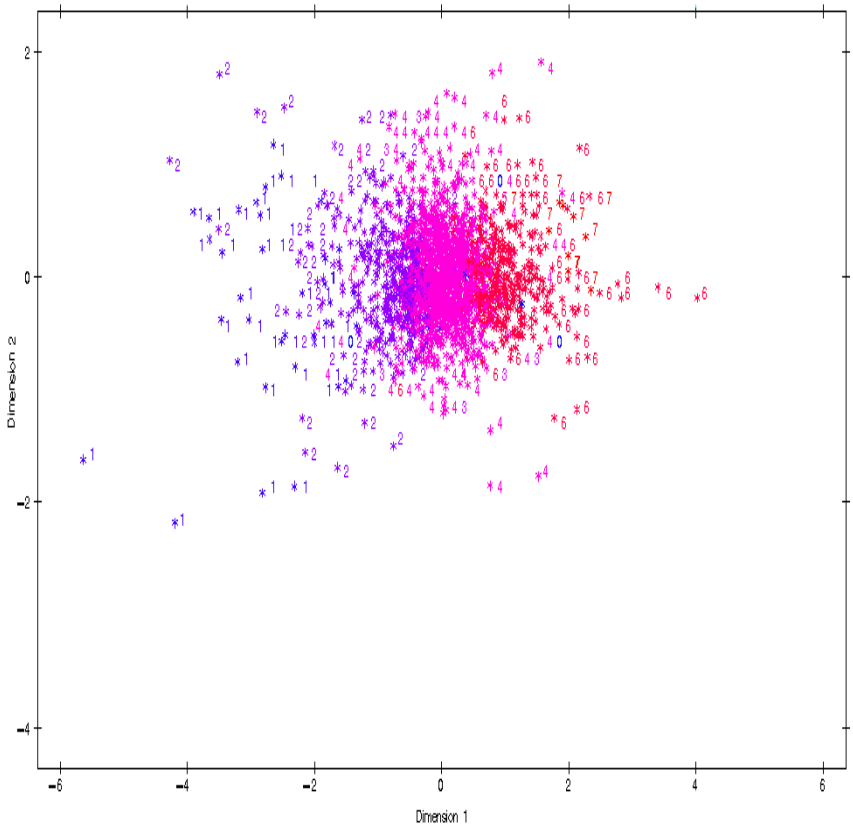

Figure 5.8. *PCA Plot of Adenocarcinoma-Normal Tissue Clusters.*

4. What can be done with the list of expressed genes in both types of tumor cells?

For the last question Osier [74] noted that the answer is not so easy and that an expert biological researcher should decide how to interpret the results, develop new questions, and design new experiments. An overview of current methods of approach to analyze the expression profiles and aid in interpretation of the data is also described in Osier [74]. An interesting paper by Cooper [18] calls attention to the use of precise criteria in the interpretation of microarray data. This work is left for biologists.

Table 5.1. *Distribution of* 1038 *genes present in both Adenoma and Adenocarcinoma tissue types. Genes in group* 0 *are genes not grouped by PPCLUST, and genes in group* 4 *are not expressed in either tissue types.*

	Adenocarcinoma Groups							
Adenoma Groups	0	1	2	3	4	5	6	7
0	0	0	1	0	1	0	0	0
1	0	2	4	0	3	0	0	0
2	0	10	41	0	38	0	4	0
4	0	12	148	2	558	1	152	3
5	1	0	6	0	24	0	24	2
6	0	0	0	0	0	0	1	0

5.2 Real Data Study II: Cellular Gene Expression upon HIV data

The analysis of gene expressions in HIV data is another important application of HDLSS clustering algorithms. According to Wout et al. [110], "human immudeficiency virus type 1 (HIV-1) is an enveloped retrovirus that causes severe depletion of the immune system in humans, leaving them susceptible to infection with other pathogens". HIV can lead to acquired immunodeficiency syndrome (AIDS) that results in immune system failure and leads to serious infections. A representation of a cross section of the human immunodeficiency virus is given in Fig. 5.9. The number of people killed by AIDS has been estimated by World Health Organization to be more than 25 million since it was first detected in 1981.

In this section, the dataset described by van't Wout et al. [110] and McLachlan et al. [66] is used. The data consist of expression levels of 7680 genes in $CD4^+$-T-cell lines, after 24 hours of infection with the HIV-1 virus. There are four control slides of pooled mRNA from uninfected cells and four slides of pooled mRNA from the infected cells, so that there is a total of eight arrays for each of the 7680 genes. The dataset includes 12 HIV-1 genes that can be used as positive controls as they are known to be differentially expressed in infected and uninfected cells.

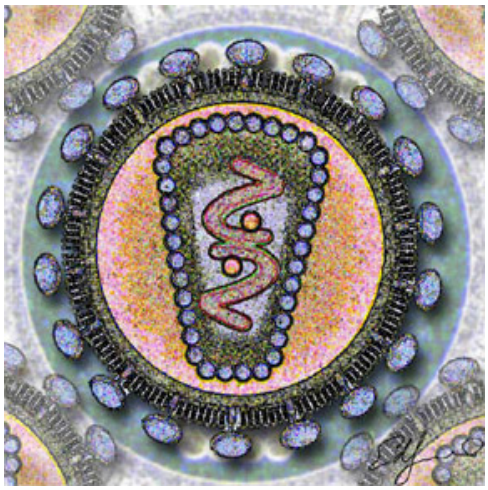

Figure 5.9. *Stylized rendering of a cross section of the human immunodeficiency virus (Wikipedia).*

The log expression values were column normalized [7], as in McLachlan et al. [66]. PPCLUST was applied to the absolute value of the difference in expression values of uninfected and infected cells. The objective was to see how PPCLUST deals with the 12 genes known to be differentially expressed and observe whether the algorithm could detect other genes as being differentially expressed. PPCLUST was used with a significance level of 10^{-5}. The algorithm detected 6 groups. Only 7 genes could not be allocated to any of the groups. The two highly expressed groups had 1062 (13.83%) and 48 (0.63%) genes, respectively, and the 12 HIV-1 genes were allocated to those two groups. Figure 5.10 shows the original heatmap for the HIV data with no clear pattern in gene expression levels. After the genes were clustered and ordered according to respective groups, the new heatmap in Fig. 5.11 shows a clear pattern in distribution of low expressed and high expressed genes for the HIV data indicating that PPCLUST worked well in this case too.

[7]For each column of log expression, it was subtracted the column mean and divided by the column standard deviation.

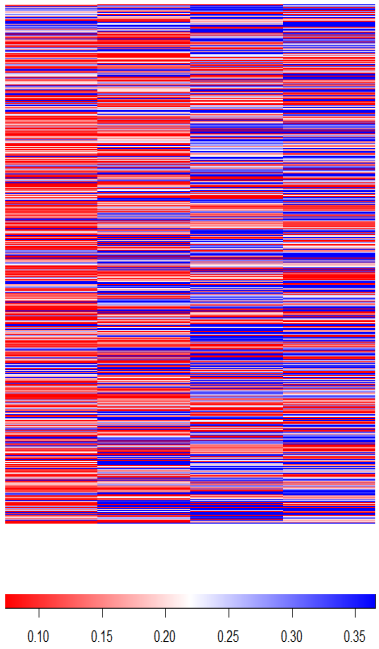

Figure 5.10. *Heatmap for original HIV data.*

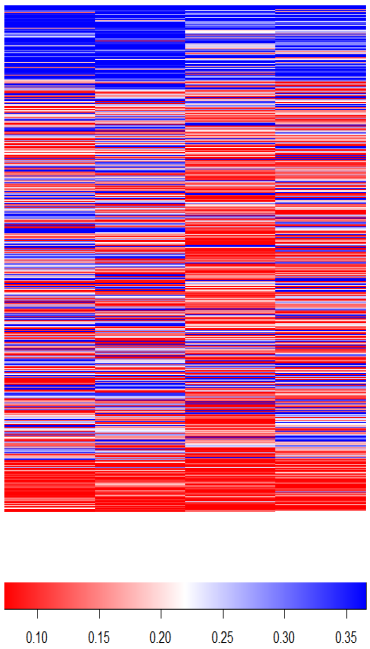

Figure 5.11. *Heatmap for grouped HIV data. Genes ordered by groups.*

6

Partition Clustering of HDLLSS Data Based on p-Values

6.1 Introduction

In this chapter, the PPCLUST algorithm is extended to consider a large number of factor levels with a few replications over time. Wang [111] uses a marginal model for making inferences about the effects of a fixed small number of factor levels replicated over a large number of time points. Wang's test is extended to the situation where there is a large number of factor levels: $a \to \infty$. Each factor level is observed over a fixed number of time points (b) with a few replications at each time point (n_i). This data setup is denoted as high dimensional longitudinal low sample size (HDLLSS) data and the new algorithm is named PPCLUSTEL (p-values based partitional clustering of longitudinal data).

There are many examples of studies in microarray technology that use HDLLSS data. Some are cell cycle studies of yeast, primary fibroblasts, human cancer, HeLa cells, mouse embryo fibroblasts and others. Statistical analysis techniques used in the cell cycle study[1] include Fourier transformation (Spellman et al. [99]), visual inspection (Cho et al. [16]), self-

[1]Most of the techniques are described in Hastie et al. [43]

organizing maps (Tamayo et al. [104]), k-means clustering (Tavazoie et al. [105]), single pulse modeling (Zhao et al. [120]), QT-clustering (Heyer et al. [44]), singular value decomposition (Holter et al. [45], Alter et al. [8]), correspondence analysis (Fellenberg et al. [26]), wavelet analysis (Klevecz [57]), and warping algorithms (Aach and Church [1]).

As in HDLSS studies, assumptions like constant variance and normality also restrict full generality in HDLLSS data studies. Shedden and Cooper [94] perform a numerical characterization of the time points for each gene by least squares fitting. Additionally, issues like significance level for multiple comparison are also present in most studies, as in Wang et al. [113] which describes an agglomerative method for clustering cell cycle related genes through a dissimilarity measure using gene by gene comparison. More than that, the results vary so widely from method to method that some of them are practically impossible to reproduce.

For the purpose of clustering factor levels in HDLLSS data, we developed a procedure similar to the one discussed in Chapter 3, that applies to longitudinal data . The idea is to obtain a statistic to test for no simple effect of group of factor levels (genes in microarray), or equivalently, no main effect of group and no group/time interaction effect when genes in each factor level are repeatedly observed. Once this statistic is obtained, the algorithm developed in Chapter 3 is adapted to cluster factor levels using p-values, again, as a similarity measure. The efficiency of this new algorithm is studied through simulations.

6.2 Theory Development for Testing No Simple Effect of Group in HDLLSS Data

Consider $X_{ik} = (X_{i1k}, \ldots, X_{ibk})$ representing a subject k nested in a factor level i, where $i = 1, \ldots, a$, $k = 1, \ldots, n_i$; and each subject is measured at b time points $t_1, t_2, \ldots, t_b$. Also, let a be large ($a \to \infty$), b fixed, and n_i small. The data structure can be easily seen in Table 6.1.

Modeling X_{ijk} as $\mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}$, for $i = 1, \ldots, a$; $j = 1, \ldots, b$; $k = 1, \ldots, n_i$; with

Table 6.1. *Data structure in longitudinal repeated measures design.*

Factor Level	Subject	Time Points t_1	t_2	...	t_b
1	1	X_{111}	X_{121}	...	X_{1b1}
	2	X_{112}	X_{122}	...	X_{1b2}
	⋮	⋮	⋮	⋮	⋮
	n_1	X_{11n_1}	X_{12n_1}	...	X_{1bn_1}
2	1	X_{211}	X_{221}	...	X_{2b1}
	2	X_{212}	X_{222}	...	X_{2b2}
	⋮	⋮	⋮	⋮	⋮
	n_2	X_{21n_2}	X_{22n_2}	...	X_{2bn_2}
⋮	⋮	⋮	⋮	⋮	⋮
a	1	X_{a11}	X_{a21}	...	X_{ab1}
	2	X_{a12}	X_{a22}	...	X_{ab2}
	⋮	⋮	⋮	⋮	⋮
	n_a	X_{a1n_a}	X_{a2n_a}	...	X_{abn_a}

$E(\epsilon_{ijk}) = 0$ and $X_{ijk} \sim F_{ij}$ arbitrary, the parametric hypothesis of no simple effect can be written as,

$$H_0(\phi): \quad \text{all} \;\; \phi_{ij} = \alpha_i + \gamma_{ij} = 0, \tag{6.1}$$

where $\sum_{i=1}^{a} \alpha_i = \sum_{j=1}^{b} \beta_j = \sum_{i=1}^{a} \gamma_{ij} = \sum_{j=1}^{b} \gamma_{ij} = 0$.

Assume that

$$\text{cov}(X_{ijk}, X_{i'j'k'}) = \begin{cases} \sigma_{ijj'} & \text{if } i = i',\ k = k' \\ 0 & \text{if } i \neq i' \text{ or } k \neq k' \end{cases}$$

meaning that observations for the same factor level and subject are correlated, while observations for different factor levels and/or different subjects are not correlated[2].

Throughout this chapter the following notation is used:

$\bar{X}_{ij.} = \frac{1}{n_i} \sum_{k=1}^{n_i} X_{ijk}, \qquad \tilde{X}_{.j.} = \frac{1}{a} \sum_{i=1}^{a} \bar{X}_{ij.},$

$\bar{X}_{...} = \frac{1}{N} \sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{k=1}^{n_i} X_{ijk}, \qquad \tilde{X}_{...} = \frac{1}{ab} \sum_{i=1}^{a} \sum_{j=1}^{b} \bar{X}_{ij.}$

where $N = b \sum_{i=1}^{a} n_i$. The estimates for the parametric model are,

[2]Note that the only assumption is about independence of X_{ijk} for different factor levels and/or different subjects.

$\hat{\mu} = \bar{X}_{...}, \qquad \hat{\alpha}_i = \bar{X}_{i..} - \bar{X}_{...} = \tilde{X}_{i..} - \bar{X}_{...}$

$\hat{\beta}_j = \bar{X}_{.j.} - \bar{X}_{...} = \tilde{X}_{.j.} - \bar{X}_{...}, \qquad \hat{\gamma}_{ij} = \bar{X}_{ij.} - \bar{X}_{i..} - \bar{X}_{.j.} + \bar{X}_{...}$, and

$\hat{\alpha}_i + \hat{\gamma}_{ij} = \bar{X}_{ij.} - \bar{X}_{.j.} = \bar{X}_{ij.} - \tilde{X}_{.j.}$, as usual.

Now define the statistics

$$MS\varphi = \frac{1}{(a-1)b} \sum_{i=1}^{a} \sum_{j=1}^{b} (\bar{X}_{ij.} - \tilde{X}_{.j.})^2 \tag{6.2}$$

and

$$MSE = \frac{1}{ab} \sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{k=1}^{n_i} \frac{(X_{ijk} - \bar{X}_{ij.})^2}{n_i(n_i - 1)} \tag{6.3}$$

Proposition 6.1. *Under $H_0(\phi)$, $E(MS\varphi) = E(MSE)$.*

Proof. First, let's rewrite $MS\varphi$ and MSE in terms of residuals.

For $MS\varphi$ note that,

$$\bar{X}_{ij.} = \frac{1}{n_i} \sum_{k=1}^{n_i} X_{ijk} = \frac{1}{n_i} \sum_{k=1}^{n_i} (\mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}) = \mu + \alpha_i + \beta_j + \gamma_{ij} + \bar{\epsilon}_{ij.}$$

and,

$$\begin{aligned}
\tilde{X}_{.j.} &= \frac{1}{a} \sum_{i=1}^{a} \bar{X}_{ij.} = \frac{1}{a} \sum_{i=1}^{a} \frac{1}{n_i} \sum_{k=1}^{n_i} X_{ijk} = \frac{1}{a} \sum_{i=1}^{a} \frac{1}{n_i} \sum_{k=1}^{n_i} (\mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}) \\
&= \mu + \frac{1}{a} \sum_{i=1}^{a} \frac{\sum_{k=1}^{n_i} \alpha_i}{n_i} + \frac{1}{a} \sum_{i=1}^{a} \frac{\sum_{k=1}^{n_i} \beta_j}{n_i} + \frac{1}{a} \sum_{i=1}^{a} \frac{\sum_{k=1}^{n_i} \gamma_{ij}}{n_i} + \frac{1}{a} \sum_{i=1}^{a} \frac{\sum_{k=1}^{n_i} \epsilon_{ijk}}{n_i} \\
&= \mu + \frac{1}{a} \sum_{i=1}^{a} \alpha_i + \frac{1}{a} \sum_{i=1}^{a} \beta_j + \frac{1}{a} \sum_{i=1}^{a} \gamma_{ij} + \tilde{\epsilon}_{.j.} \\
&= \mu + \beta_j + \tilde{\epsilon}_{.j.}
\end{aligned} \tag{6.4}$$

Thus,

$$\bar{X}_{ij.} - \tilde{X}_{.j.} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \bar{\epsilon}_{ij.} - (\mu + \beta_j + \tilde{\epsilon}_{.j.}) = \alpha_i + \gamma_{ij} + \bar{\epsilon}_{ij.} - \tilde{\epsilon}_{.j.}$$

and under the null hypothesis, $H_0 : \alpha_i + \gamma_{ij} = 0$,

$$\bar{X}_{ij.} - \tilde{X}_{.j.} = \bar{\epsilon}_{ij.} - \tilde{\epsilon}_{.j.}$$

Finally, $MS\varphi$ can be rewritten as,

$$\begin{aligned} MS\varphi &= \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}(\bar{\epsilon}_{ij.}-\tilde{\epsilon}_{.j.})^2 = \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}(\bar{\epsilon}_{ij.}^2-2\bar{\epsilon}_{ij.}\tilde{\epsilon}_{.j.}+\tilde{\epsilon}_{.j.}^2) \\ &= \frac{1}{(a-1)b}\left[\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2-2\sum_{j=1}^{b}\tilde{\epsilon}_{.j.}\sum_{i=1}^{a}\bar{\epsilon}_{ij.}+a\sum_{j=1}^{b}\tilde{\epsilon}_{.j.}^2\right] \\ &= \frac{1}{(a-1)b}\left[\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2-2a\sum_{j=1}^{b}\tilde{\epsilon}_{.j.}^2+a\sum_{j=1}^{b}\tilde{\epsilon}_{.j.}^2\right] = \frac{1}{(a-1)b}\left[\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2-a\sum_{j=1}^{b}\tilde{\epsilon}_{.j.}^2\right] \end{aligned}$$

Thus,

$$MS\varphi = \frac{1}{(a-1)b}\left[\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2-a\sum_{j=1}^{b}\tilde{\epsilon}_{.j.}^2\right]. \tag{6.5}$$

Now, for MSE note that,

$$X_{ijk}-\bar{X}_{ij.} = \mu+\alpha_i+\beta_j+\gamma_{ij}+\epsilon_{ijk}-(\mu+\alpha_i+\beta_j+\gamma_{ij}-\bar{\epsilon}_{ij.}) = \epsilon_{ijk}-\bar{\epsilon}_{ij.}$$

and,

$$MSE = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n_i}\frac{(\epsilon_{ijk}-\bar{\epsilon}_{ij.})^2}{n_i(n_i-1)} = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\left[\frac{\sum_{k=1}^{n_i}\epsilon_{ijk}^2-n_i\bar{\epsilon}_{ij.}^2}{n_i(n_i-1)}\right].$$

Thus,

$$MSE = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\left[\frac{\sum_{k=1}^{n_i}\epsilon_{ijk}^2-n_i\bar{\epsilon}_{ij.}^2}{n_i(n_i-1)}\right]. \tag{6.6}$$

Then, it is easy to observe that $E(MS\varphi) = E(MSE)$, since

$$
\begin{aligned}
E(MS\varphi) &= E\left[\frac{1}{(a-1)b}\left[\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 - a\sum_{j=1}^{b}\bar{\epsilon}_{.j.}^2\right]\right] \\
&= \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}E(\bar{\epsilon}_{ij.}^2) - \frac{a}{(a-1)b}\sum_{j=1}^{b}E(\bar{\epsilon}_{.j.}^2) \\
&= \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\sigma_{ij}^2}{n_i} - \frac{a}{(a-1)b}\sum_{j=1}^{b}\sum_{i=1}^{a}E\left[\left(\frac{\bar{\epsilon}_{ij.}}{a}\right)^2\right] \\
&= \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\sigma_{ij}^2}{n_i} - \frac{1}{(a-1)ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\sigma_{ij}^2}{n_i} \\
&= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\sigma_{ij}^2}{n_i} \qquad (6.7)
\end{aligned}
$$

and,

$$
\begin{aligned}
E(MSE) &= E\left[\frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\left[\frac{\sum_{k=1}^{n_i}\epsilon_{ijk}^2 - n_i\bar{\epsilon}_{ij.}^2}{n_i(n_i-1)}\right]\right] = \frac{1}{ab}\left[\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\sum_{k=1}^{n_i}\sigma_{ij}^2 - \sigma_{ij}^2}{n_i(n_i-1)}\right] \\
&= \frac{1}{ab}\left[\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{n_i\sigma_{ij}^2 - \sigma_{ij}^2}{n_i(n_i-1)}\right] = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\sigma_{ij}^2}{n_i} \qquad (6.8)
\end{aligned}
$$

□

From Proposition 6.1 it is reasonable to use a function of $\frac{MS\varphi}{MSE}$ as test statistic for $H_0(\phi)$ in (6.1). The objective is to obtain the asymptotic distribution of a test statistic based on $\frac{MS\varphi}{MSE}$ that allows the testing of the null hypothesis (6.1) of no simple effect under the established conditions.

Quoting van der Vaart [109], "a common method to derive the limit distribution of a sequence of statistics T_n is to show that it is asymptotically equivalent to a sequence S_n of which the limit behavior is known. The basis of this method is Slutsky's Lemma, which shows that the sequence $T_n = T_n - S_n + S_n$ converges in distribution to S if both $T_n - S_n \xrightarrow{p} 0$ and $S_n \xrightarrow{d} S$." Thus, one alternative is to find a suitable sequence S_n that is sufficiently

simple to allow the limit properties to be known, and close enough to T_n. This sequence can be obtained through a projection method. A projection of a random variable is defined as the closest element in a given set of functions, and can be used for the purpose of finding the asymptotic distribution of a sequence of variables through comparison to the projected form. The Hájek projection consists of a set S of all variables of the form $\sum_{i=1}^{n} g_i(X_i)$ for arbitrary measurable functions g_i, where $X_1, \ldots, X_n$ are independent random vectors and $E[g_i^2(X_i)] < \infty$. The interest in this projection form is that the convergence in distribution of the sums can be derived from the Central Limit Theorem. Applying the Hájek projection method to a quadratic form yields a sequence of statistics that is asymptotically equivalent to the sequence of quadratic forms. The next step here is to find the projection of $MS\varphi - MSE$ onto a class S of form $\sum_{i,k} g_{ik}(\mathbf{X}_{ik})$, where $\mathbf{X}_{ik}$ are independent random vectors, and find the asymptotic distribution of a test statistic for $H_o(\phi)$ using the projection of $MS\varphi - MSE$.

The following proposition helps to find the projection of $MS\varphi$.

Proposition 6.2. *If the observations X_{ijk} have finite centered second moment,*

$$T_1 = \frac{\sqrt{ab}}{(a-1)ab} \sum_{i \neq i'}^{a} \sum_{j=1}^{b} \bar{\epsilon}_{ij\cdot} \bar{\epsilon}_{i'j\cdot} \xrightarrow{p} 0 \tag{6.9}$$

as $a \to \infty$.

Proof. We need to show that $E(T_1) = 0$ and $E(T_1^2) \to 0$ as $a \to \infty$. Note that,

$$E(T_1) = E\left[\frac{\sqrt{ab}}{(a-1)ab} \sum_{i \neq i'}^{a} \sum_{j=1}^{b} \bar{\epsilon}_{ij\cdot} \bar{\epsilon}_{i'j\cdot} \right] = 0$$

from the model, and

$$\begin{aligned}
E(T_1^2) &= \frac{ab}{(a-1)^2a^2b^2}\sum_{i\neq i'}^{a}\sum_{i_1\neq i_1'}^{a}\sum_{j=1}^{b}\sum_{j_1=1}^{b} E\left(\bar{\epsilon}_{ij.}\bar{\epsilon}_{i'j.}\bar{\epsilon}_{i_1j_1.}\bar{\epsilon}_{i_1'j_1.}\right) \\
&= \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j=1}^{b}\sum_{j_1=1}^{b} E\left(\bar{\epsilon}_{ij.}\bar{\epsilon}_{i'j.}\bar{\epsilon}_{ij_1.}\bar{\epsilon}_{i'j_1.}\right) \\
&= \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j=1}^{b}\sum_{j_1=1}^{b} E\left(\bar{\epsilon}_{ij.}\bar{\epsilon}_{ij_1.}\right) E\left(\bar{\epsilon}_{i'j.}\bar{\epsilon}_{i'j_1.}\right) \\
&= \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j=1}^{b} E\left(\bar{\epsilon}_{ij.}^2\right) E\left(\bar{\epsilon}_{i'j.}^2\right) + \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j\neq j'}^{b} E\left(\bar{\epsilon}_{ij.}\bar{\epsilon}_{ij'.}\right) E\left(\bar{\epsilon}_{i'j.}\bar{\epsilon}_{i'j'.}\right)
\end{aligned}$$

Let $E(T_1^2) = Q_1 + Q_2$. It is possible to observe that, as $a \to \infty$, with b fixed,

$$Q_1 = \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j=1}^{b} E\left(\bar{\epsilon}_{ij.}^2\right) E\left(\bar{\epsilon}_{i'j.}^2\right) = \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j=1}^{b}\frac{\sigma_{ij}^2}{n_i}\frac{\sigma_{i'j}^2}{n_{i'}} \to 0$$

and

$$\begin{aligned}
Q_2 &= \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j\neq j'}^{b} E\left(\bar{\epsilon}_{ij.}\bar{\epsilon}_{ij'.}\right) E\left(\bar{\epsilon}_{i'j.}\bar{\epsilon}_{i'j'.}\right) \\
&= \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j\neq j'}^{b}\frac{1}{n_i^2}\sum_{k=1}^{n_i}\sigma_{ijj'}\frac{1}{n_{i'}^2}\sum_{k=1}^{n_{i'}}\sigma_{i'jj'} \\
&= \frac{2}{(a-1)^2ab}\sum_{i\neq i'}^{a}\sum_{j\neq j'}^{b}\frac{\sigma_{ijj'}\sigma_{i'jj'}}{n_in_{i'}} \to 0
\end{aligned}$$

□

Proposition 6.3. *The projection of $MS\varphi$ is given by,*

$$PMS\varphi = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 \tag{6.10}$$

Proof.

$$\begin{aligned} PMS\varphi &= \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 - \frac{a}{(a-1)b}\sum_{j=1}^{b}\tilde{\epsilon}_{.j.}^2 \\ &= \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 - \frac{a}{(a-1)b}\sum_{j=1}^{b}\sum_{i=1}^{a}\sum_{i'=1}^{a}\frac{\bar{\epsilon}_{ij.}\bar{\epsilon}_{i'j.}}{a^2} \\ &= \frac{1}{(a-1)b}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 - \frac{1}{(a-1)ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 - \frac{1}{(a-1)ab}\sum_{i\neq i'}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}\bar{\epsilon}_{i'j.} \\ &= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 - \frac{1}{(a-1)ab}\sum_{i\neq i'}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}\bar{\epsilon}_{i'j.} \end{aligned}$$

Thus, using Proposition 6.2, the projection of $MS\varphi$ can be written as $PMS\varphi = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2$.

□

Proposition 6.4. *The projection of MSE is still MSE.*

Proof.

$$\begin{aligned} MSE &= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\left[\frac{\sum_{k=1}^{n_i}\epsilon_{ijk}^2 - n_i\bar{\epsilon}_{ij.}^2}{n_i(n_i-1)}\right] \\ &= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n_i}\frac{\epsilon_{ijk}^2}{n_i(n_i-1)} - \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\bar{\epsilon}_{ij.}^2}{(n_i-1)}. \end{aligned}$$

□

Proposition 6.5. *The projection of $MS\varphi - MSE$ is given by*

$$PMS\varphi - MSE = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{1}{(n_i-1)n_i}\sum_{k\neq k'}^{n_i}\epsilon_{ijk}\epsilon_{ijk'} \tag{6.11}$$

Proof. By Propositions 6.3 and 6.4,

$$
\begin{aligned}
PMS\varphi - MSE &= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2 - \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\left[\frac{\sum_{k=1}^{n_i}\epsilon_{ijk}^2 - n_i\bar{\epsilon}_{ij.}^2}{n_i(n_i-1)}\right] \\
&= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{n_i}{(n_i-1)}\bar{\epsilon}_{ij.}^2 - \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n_i}\frac{\epsilon_{ijk}^2}{n_i(n_i-1)} \\
&= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{n_i}{(n_i-1)n_i^2}\sum_{k\neq k'}^{n_i}\epsilon_{ijk}\epsilon_{ijk'} + \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n_i}\frac{n_i}{(n_i-1)n_i^2}\,\epsilon_{ijk}^2 \\
&\quad -\frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n_i}\frac{\epsilon_{ijk}^2}{n_i(n_i-1)} \\
&= \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{1}{(n_i-1)n_i}\sum_{k\neq k'}^{n_i}\epsilon_{ijk}\epsilon_{ijk'}
\end{aligned}
$$

□

Lyapunov's Central Limit Theorem

Let $\{X_k, k \geq 1\}$ *be an independent sequence of random variables satisfying* $E(X_k) = 0$, $Var(X_k) = \sigma_k^2 < \infty$, $s_n^2 = \sum_{k=1}^{n}\sigma_k^2$. *If for some* $\delta > 0$

$$\sum_{k=1}^{n}\frac{E|X_k|^{2+\delta}}{S_n^{2+\delta}} \to 0$$

Then

$$\frac{S_n}{s_n} \to N(0,1),$$

where $S_n = X_1 + X_2 + \ldots + X_n$.

Proof. See Resnick [84]. □

The following propositions help to verify Lyapounov's condition and to prove Theorem 6.1, that follows. With Proposition 6.6, the use of traditional F-statistic MST/MSE can be modified to consider using $MST - MSE$ only.

Proposition 6.6. $MSE \xrightarrow{p} lim_{a\to\infty}E(MSE)$ *if the observations* X_{ijk} *have finite centered second moment.*

Proof. Note that, for $a \to \infty$ and b fixed, by (6.8) $E(MSE) \to 0$ and the variance of MSE is

$$
\begin{aligned}
Var(MSE) &= Var\left(\frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n_i}\frac{(\epsilon_{ijk}-\bar{\epsilon}_{ij.})^2}{n_i(n_i-1)}\right) \\
&= Var\left(\frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{\sum_{k=1}^{n_i}\epsilon_{ijk}^2 - n_i\bar{\epsilon}_{ij.}^2}{n_i(n_i-1)}\right) \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\frac{1}{n_i^2(n_i-1)^2}Var\left(\sum_{j=1}^{b}\sum_{k=1}^{n_i}\epsilon_{ijk}^2 - n_i\sum_{j=1}^{b}\bar{\epsilon}_{ij.}^2\right) \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\frac{1}{n_i^2(n_i-1)^2}Var\left[\sum_{j=1}^{b}\sum_{k=1}^{n_i}\epsilon_{ijk}^2 - n_i\left(\sum_{j=1}^{b}\sum_{k\neq k'}^{n_i}\frac{\epsilon_{ijk}\epsilon_{ijk'}}{n_i^2} + \sum_{j=1}^{b}\sum_{k=1}^{n_i}\frac{\epsilon_{ijk}^2}{n_i^2}\right)\right] \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\frac{1}{n_i^2(n_i-1)^2}Var\left[\frac{(n_i-1)}{n_i}\sum_{j=1}^{b}\sum_{k=1}^{n_i}\epsilon_{ijk}^2 - \frac{1}{n_i}\sum_{j=1}^{b}\sum_{k\neq k'}^{n_i}\epsilon_{ijk}\epsilon_{ijk'}\right] \\
&\leq \frac{1}{a^2b^2}\sum_{i=1}^{a}\frac{1}{n_i^2(n_i-1)^2}\left[\frac{(n_i-1)^2}{n_i^2}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sum_{k=1}^{n_i}\sum_{k_1=1}^{n_i}E\left(\epsilon_{ijk}\epsilon_{ijk_1}\epsilon_{ij_1k}\epsilon_{ij_1k_1}\right)\right. \\
&\quad \left. -\frac{1}{n_i^2}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sum_{k\neq k'}^{n_i}\sum_{k_1\neq k_1'}^{n_i}E\left(\epsilon_{ijk}\epsilon_{ijk'}\epsilon_{ij_1k_1}\epsilon_{ij_1k_1'}\right)\right] \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\frac{1}{n_i^2(n_i-1)^2}\left[\frac{2(n_i-1)^2}{n_i^2}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sum_{k=1}^{n_i}E\left(\epsilon_{ijk}\epsilon_{ijk_1}\right)E\left(\epsilon_{ij_1k}\epsilon_{ij_1k_1}\right)\right. \\
&\quad \left. +\frac{2}{n_i^2}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sum_{k\neq k'}^{n_i}E\left(\epsilon_{ijk}\epsilon_{ijk'}\right)E\left(\epsilon_{ij_1k}\epsilon_{ij_1k'}\right)\right] \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\frac{1}{n_i^2(n_i-1)^2}\left[\frac{2(n_i-1)^2}{n_i^2}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sum_{k=1}^{n_i}\sigma_{ijj_1}^2 + \frac{2}{n_i^2}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sum_{k\neq k'}^{n_i}\sigma_{ijj_1}^2\right] \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\frac{1}{n_i^2(n_i-1)^2}\left[\frac{2(n_i-1)^2}{n_i}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sigma_{ijj_1}^2 + \frac{2(n_i-1)}{n_i}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sigma_{ijj_1}^2\right] \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\left[\frac{2}{n_i^3}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sigma_{ijj_1}^2 + \frac{2}{n_i^3(n_i-1)}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sigma_{ijj_1}^2\right] \\
&= \frac{1}{a^2b^2}\sum_{i=1}^{a}\left[\frac{2}{n_i^2(n_i-1)}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sigma_{ijj_1}^2\right] = O\left(\frac{1}{a}\right) \to 0
\end{aligned}
$$

Thus, $MSE \xrightarrow{p} lim_{a\to\infty}E(MSE)$. □

Proposition 6.7. *If the observations X_{ijk} have $(2+\delta)$ centered moment finite ($\delta > 0$), the projection $\sqrt{ab}(PMS\varphi - MSE)$ converges in distribution, when n_i are uniformly bounded.*

Proof. From Proposition 6.5,

$$\sqrt{ab}\,(PMS\varphi - MSE) = \frac{1}{\sqrt{ab}} \sum_{i=1}^{a} \sum_{j=1}^{b} \frac{1}{(n_i - 1)n_i} \sum_{k \neq k'}^{n_i} \epsilon_{ijk}\epsilon_{ijk'}$$

Then,

$$E\left(\sqrt{ab}\,(PMS\varphi - MSE)\right) = \frac{1}{\sqrt{ab}} \sum_{i=1}^{a} \sum_{j=1}^{b} \frac{1}{(n_i - 1)n_i} \sum_{k \neq k'}^{n_i} E(\epsilon_{ijk}\epsilon_{ijk'}) = 0 \quad (6.12)$$

from the model. Also,

$$\begin{aligned}
Var\left(\sqrt{ab}\,(PMS\varphi - MSE)\right) &= \frac{1}{ab} \sum_{i=1}^{a} \frac{1}{(n_i - 1)^2\, n_i^2} Var\left(\sum_{j=1}^{b} \sum_{k \neq k'}^{n_i} \epsilon_{ijk}\epsilon_{ijk'}\right) \\
&= \frac{1}{ab} \sum_{i=1}^{a} \frac{1}{(n_i - 1)^2\, n_i^2} \sum_{j=1}^{b} \sum_{j_1=1}^{b} \sum_{k \neq k'}^{n_i} \sum_{k_1 \neq k_1'}^{n_i} E\left(\epsilon_{ijk}\epsilon_{ijk'}\epsilon_{ij_1k_1}\epsilon_{ij_1k_1'}\right) \\
&= \frac{2}{ab} \sum_{i=1}^{a} \frac{1}{(n_i - 1)^2\, n_i^2} \sum_{j=1}^{b} \sum_{j_1=1}^{b} \sum_{k \neq k'}^{n_i} E\left(\epsilon_{ijk}\epsilon_{ijk'}\right) E\left(\epsilon_{ij_1k}\epsilon_{ij_1k'}\right) \\
&= \frac{2}{ab} \sum_{i=1}^{a} \frac{1}{(n_i - 1)^2\, n_i^2} \sum_{j=1}^{b} \sum_{j_1=1}^{b} \sum_{k \neq k'}^{n_i} \sigma^2_{ijj_1} \\
&= \frac{2}{ab} \sum_{i=1}^{a} \frac{1}{(n_i - 1)\, n_i} \sum_{j=1}^{b} \sum_{j_1=1}^{b} \sigma^2_{ijj_1} \qquad (6.13)
\end{aligned}$$

When n_i are uniformly bounded, using the inequality

$$\left|\sum_{i=1}^{n} Z_i\right|^p \leq n^{p-1} \sum_{i=1}^{n} |Z_i|^p, \quad n \geq 1, \quad p \geq 1$$

which for $p > 1$ follows from Hölder's inequality, it is possible to note that,

$$
\begin{aligned}
E\left[\sqrt{ab}\,(PMS\varphi - MSE)\right]^{2+\delta} &= \sum_{i=1}^{a}\sum_{j=1}^{b} E\left[\frac{1}{\sqrt{ab}\;n_i(n_i-1)}\sum_{k\neq k'}^{n_i}\epsilon_{ijk}\epsilon_{ijk'}\right]^{2+\delta} \\
&= \sum_{i=1}^{a}\sum_{j=1}^{b}\frac{1}{\left(\sqrt{ab}\;n_i(n_i-1)\right)^{2+\delta}}\;E\left(\left|\sum_{k\neq k'}^{n_i}\epsilon_{ijk}\epsilon_{ijk'}\right|^{2+\delta}\right) \\
&\leq \sum_{i=1}^{a}\sum_{j=1}^{b}\frac{1}{\left(\sqrt{ab}\;n_i(n_i-1)\right)^{2+\delta}}\;[n_i(n_i-1)]^{1+\delta}\sum_{k\neq k'}^{n_i}E\left|\epsilon_{ijk}\epsilon_{ijk'}\right|^{2+\delta} \\
&= \sum_{i=1}^{a}\sum_{j=1}^{b}\frac{1}{(ab)^{\frac{2+\delta}{2}}\;n_i(n_i-1)}\sum_{k\neq k'}^{n_i}E\left(|\epsilon_{ijk}|^{2+\delta}\right)E\left(|\epsilon_{ijk'}|^{2+\delta}\right) \\
&\leq \sum_{i=1}^{a}\sum_{j=1}^{b}\frac{1}{(ab)^{\frac{2+\delta}{2}}\;n_i(n_i-1)}\sum_{k=1}^{n_i}\left(E\left|\epsilon_{ijk}\right|^{2+\delta}\right)^2 \to 0
\end{aligned}
$$

as $a \to \infty$ and b fixed. □

Using the previous results, the following theorem allows testing for no simple effect of a large number of factor levels (a), observed over a fixed number of time points (b) and with a small number of replications (n_i) under each time point.

Theorem 6.1. *(Test of no simple effect of a group of factor levels.)*

Let $X_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}$, with μ the overall effect, α_i, $i = 1, \ldots, a$, the mean factor effect, β_j, $j = 1, \ldots, b$, the time point effect, γ_{ij} the factor-time interaction effect and ϵ_{ijk} some error term with arbitrary distribution F_{ij}, for all $k = 1, \ldots, n_i$.

Let also, $H_0(\phi)$: all $\phi_{ij} = \alpha_i + \gamma_{ij} = 0$ be satisfied. If the observations X_{ijk} have $(2+\delta)$ centered moment finite ($\delta > 0$), and the number of replications is small, with $n_i \geq 2$ and bounded, observed over a fixed number of time points b,

$$
F_\phi = \sqrt{ab}\,(MS\varphi - MSE) \xrightarrow{d} N\left(0, lim_{a\to\infty}\frac{2}{ab}\sum_{i=1}^{a}\frac{1}{(n_i-1)\;n_i}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sigma^2_{ijj_1}\right) \qquad (6.14)
$$

as $a \to \infty$.

Proof. From Propositions 6.6 and 6.7 the conditions of Liapunov's Theorem holds, and

$$\frac{\sqrt{ab}\,(P - E(P))}{Var(\sqrt{ab}\,P)} \xrightarrow{d} N(0,1)$$

where $P = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\frac{1}{(n_i-1)n_i}\sum_{k\neq k'}^{n_i}\epsilon_{ijk}\epsilon_{ijk'}$. From (6.12), (6.13) in Proposition 6.7, and under $H_0(\phi)$ of no simple effect of group,

$$\sqrt{ab}\,(MS\varphi - MSE) \xrightarrow{d} N\left(0, \frac{2}{ab}\sum_{i=1}^{a}\frac{1}{(n_i-1)\,n_i}\sum_{j=1}^{b}\sum_{j_1=1}^{b}\sigma^2_{ijj_1}\right) \tag{6.15}$$

as $a \to \infty$. □

Thus, using statistic F_ϕ in Theorem 6.1, it is possible to adapt PPCLUST algorithm in Section 3.2 through the replacement of the test statistic $\sqrt{a}(F_r - 1)$ by the new test statistic F_ϕ. The resulting partitional algorithm is called PPCLUSTEL (PPCLUST for Longitudinal data).

6.3 Simulation Study on Performance of The Test

6.3.1 Type I Error Rate

Table 6.2 reports type I error rates compared to nominal levels 0.10, 0.05 and 0.01 when testing (6.1). Three conditions were considered: 2000 factor levels with 3 time points and 4 subjects; 2000 factor levels with 10 time points and 3 replications; and 10000 factor levels with 3 time points and 4 replications. Simulations are based on 2000 replications and the distribution used was a multivariate normal distribution with mean $\cos(\pi \times (j+1))$, $j = 1, \ldots, b$ (b is the number of time points), and variance-covariance matrix Σ with unit variance and decreasing covariance for observations in more distant time points, or equivalently, $\sigma_{ijj_1} = 1 - |j - j_1| \times 0.2$.

Table 6.2. *Estimated levels for test of no single effect.*

Factor Levels	Time Points	Observations	Nominal levels 0.10	0.05	0.01
			Type I errors		
2000	3	4	0.1040	0.0505	0.0145
2000	10	3	0.0940	0.0445	0.0095
10000	3	4	0.1065	0.0510	0.0080

The estimated Type I error rates reported in Table 6.2 are close to the true α levels, indicating that the test statistic F_ϕ performs well for both small and a relatively large number of time points given only a small number of replications. However, in simulations (not shown here) it was observed that the algorithm is sensitive to changes in the variance-covariance matrix. When small covariances are considered, the type I error increases considerably. One possibility for this result is that the sample variance-covariance matrix based on no more than 4 replications is a very poor estimatior of the true variance-covariance matrix (Σ). Since the statistic F_ϕ uses squared values of Σ, unbiased estimation of squared variances would be obtained using the same estimator of σ^4 used in calculation of the test statistic used in PPCLUST. More about the estimation of the variance-covariance matrix is left for future investigations.

6.3.2 Power Curves

To study the power of the test in Theorem 6.1, datasets with 2000 factor levels over 10 time points, with 3 replications for each factor level/time point were generated. The function used to generate data was a multivariate normal with mean $\cos(\pi \times (i+1)) + d$, $i = 1, \ldots, 10$, and variance-covariance matrix with unit variance and decreasing covariances (as before) for more distant time points. The term d in the mean of the multivariate normal is a deviation from H_0 ranging from 0 to 1.6. It is included in some factor levels to examine the power of the test. Power curves were obtained under three situations: (1) 200 factor levels shifted by d; (2) 100 factor levels shifted by d; and (3) 50 factor levels shifted by d.

It is clear from Fig. 6.1 that for 200 shifted factor levels, the power of the test increases very fast and the test also showed to be extremely powerful in detecting small differences even with only 50 factor levels (2.5% of total levels) being different in data.

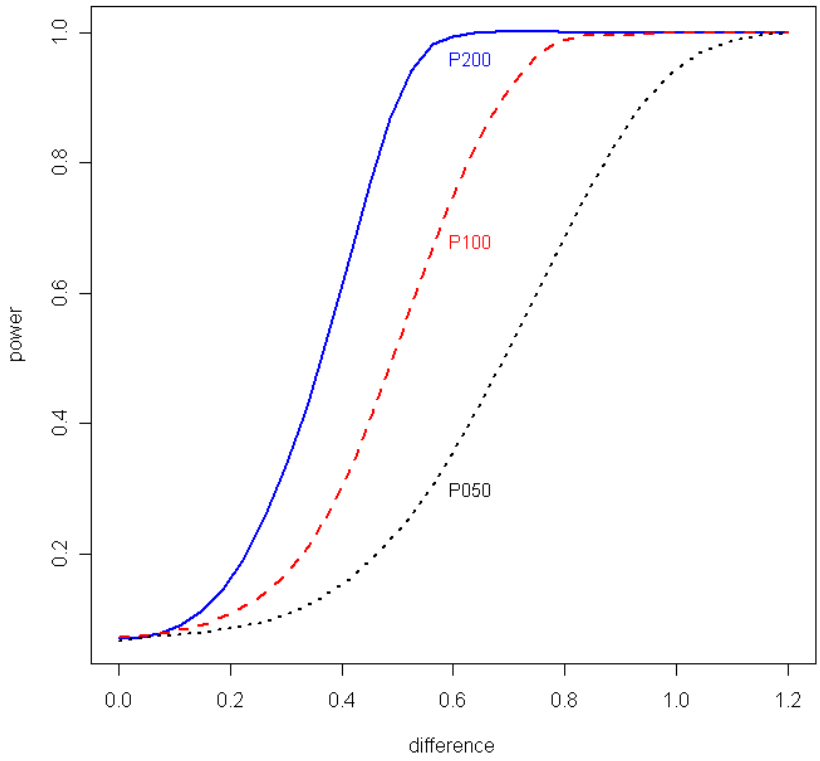

Figure 6.1. *Achieved power for HDLLSS data with* $\alpha = 0.05$*, considering shifted differences in two groups according to three cases: P200 (continuous line in blue) has 200 shifted factor levels; P100 (dashed line in red) has 100 shifted factor levels; and P050 (dotted line in black) has 50 shifted factor levels.*

In the following section a simulation is done to test the quality of PPCLUSTEL in clustering objects.

6.4 Partition Clustering of HDLLSS Data Based on p-Values

Again, it is important to notice that the algorithm follows the same eight steps described in Section 3.2, changing only the test statistic for the new test statistic F_{ϕ}. A SAS© macro called PPCLUSTEL was implemented [3]. To test the efficiency of PPCLUSTEL a dataset with 5 groups was simulated and the quality of clustering was obtained using adjusted rand index (ARI). Details of the efficiency study follows.

6.4.1 Performance on Simulated Data and Comparison with MCLUST

The data structure of HDLLSS data presented in Table 6.1 was used to test the quality of PPCLUSTEL in clustering of factor levels and to compare with MCLUST algorithm described in Chapter 2. The dataset had 2000 factor levels with 3 replications and 10 time points per factor level. The 2000 factor levels were divided into 5 groups. Each group was formed from a multivariate normal distribution with mean vector $\mu_{10\times 1}$ and variance-covariance matrix $\Sigma_{10\times 10}$. The mean of each group is generated according to the following scheme:

- Group 1: 200 factor levels with mean $\mu_{j,1} = \cos(\pi(j+1))$, $j = 1, \ldots, 10$ time points.
- Group 2: 200 factor levels with mean $\mu_{j,1} = \cos(\pi(j+1)/10)$, $j = 1, \ldots, 10$.
- Group 3: 800 factor levels with mean $\mu_{j,1} = \sin(\pi(j+1)/10)$, $j = 1, \ldots, 10$.
- Group 4: 400 factor levels with mean $\mu_{j,1} = i - 4$, $j = 1, \ldots, 10$.
- Group 5: 400 factor levels with mean $\mu_{j,1} = i/4$, $j = 1, \ldots, 10$.

All groups had factor levels generated from the same variance-covariance matrix with unit variance and decreasing covariance for more distant elements in time. Figure 6.2 represents the simulated data considering all factor levels together and separated in each group.

[3] See Appendix B for details of use.

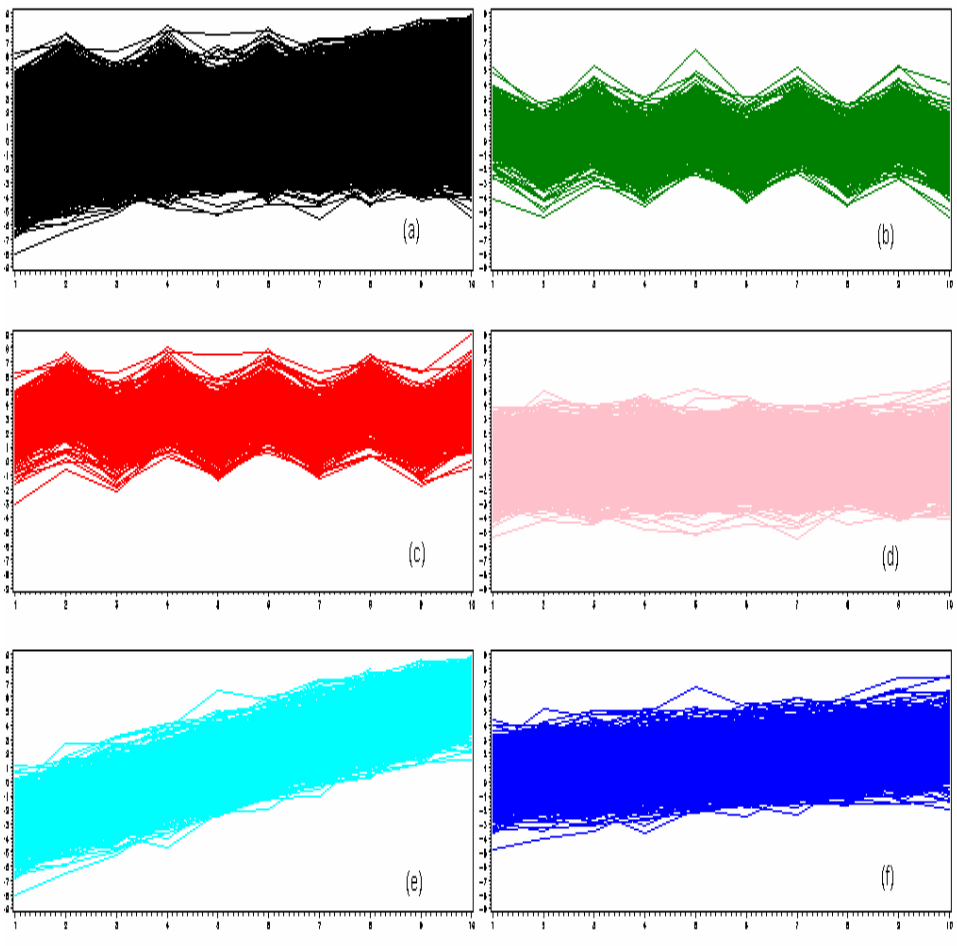

Figure 6.2. *Profile plot for simulated longitudinal data. Gene expression levels in vertical axis and time points in horizontal axis. (a) All replicated factor levels. (b) Factor levels in group 1. (c) Factor levels in group 2. (d) Factor levels in group 3. (e) Factor levels in group 4. (f) Factor levels in group 5.*

A total of 2000 simulations were performed with ARI obtained after applying PPCLUSTEL and MCLUST to cluster factor levels in each simulation. In order to apply MCLUST the mean value of 3 replications for each factor level was used since MCLUST does not allow the use of replications. The results obtained for ARI are summarized in Fig. 6.3 for both PPCLUSTEL and MCLUST.

It can be observed that both PPCLUSTEL and MCLUST were very efficient in clustering

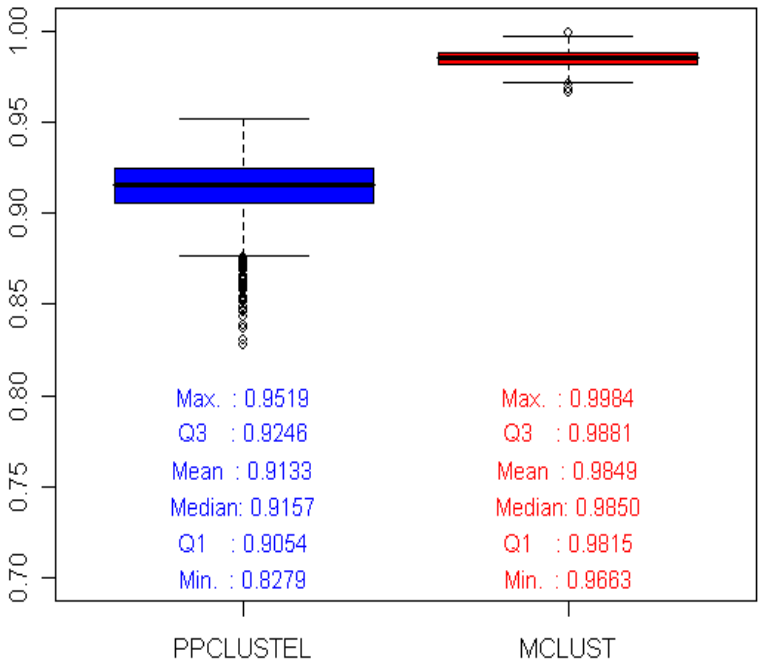

Figure 6.3. *Boxplots for ARI in 2000 simulations using PPCLUSTEL and MCLUST to cluster generated data.*

the factor levels of each group. ARI was very stable on both cases, but MCLUST was more stable and with higher values for ARI in clustering of HDLLSS data. The advantadge of MCLUST over PPCLUSTEL could be justified by the use of multivariate normal distribution to generate different groups in simulated data. However, it was observed that MCLUST is much slower than PPCLUSTEL and when the number of factor levels increases, MCLUST begins experiencing memory allocation problems and does not produce any result when the number of factor levels is about 10000 or more. Table 6.3 presents time of execution of PPCLUSTEL and MCLUST for data sets with different number of factor levels.

In this way, PPCLUSTEL can be used as an alternative algorithm when MCLUST is

Table 6.3. *Execution time of PPCLUSTEL and MCLUST for data with different number of factor levels. MCLUST is considered in two situations: (a) number of groups specified, (b) number of groups not specified.*

Factor Levels	PPCLUSTEL	MCLUST (a)	MCLUST (b)
2000	10 sec	14 sec	1 min, 5 sec
5000	37 sec	2 min, 9 sec	4 min, 38 sec
10000	2 min, 12 sec	does not execute	does not execute
20000	8 min	does not execute	does not execute

not available. When MCLUST has time restrictions or does not work, PPCLUSTEL can be used as an alternative procedure to reduce the number of factor levels in the problem, and to indicate the number of groups that should be used with MCLUST. The reduction of the number of factor levels after use of PPCLUSTEL is possible if one eliminates from the data groups with low variation over time. Those groups will be more easily identified after clustering with PPCLUSTEL. As an example, in Figure 6.2, group 3 could be eliminated from the study before application of MCLUST.

Additionally, most of the advantages observed in PPCLUST are also obtained with PPCLUSTEL, including: automatic specification of number of groups, high quality in grouping variables, no multiple comparison problems, no need of dimension reduction, fast and easy to use. The algorithm is as easy to use as the PPCLUST algorithm, since it was also written in SAS© macro language[4].

The similarity measure used in PPCLUSTEL does not require the data to be equally spaced over time, as most time series procedures do. Another issue with application of time series methods in HDLLSS data is observed by Kim and Kim [55], who note that usually the number of observed time-points in microarray studies is very small, and methods such as auto-regression, moving-average or Fourier analysis are not applicable in this situation.

One drawback of PPCLUSTEL, is that it is not invariant to monotone transformations

[4]Performance of PPCLUSTEL and MCLUST was observed in a machine using Windows XP with Intel Dual-core, 1.79Ghz, and 2Gb of RAM memory. Instructions about macro use in Appendix B.

in data, since the test statistic is obtained using the original observations. Future research should be directed to the use of PPCLUSTEL with different distributions and ranked data instead of original observations.

6.4.2 Comparison to Other Methods

Some of the recent competing methodologies of PPCLUSTEL in the area of microarray gene expression analysis are indicated in the next paragraphs.

The proposed method of Storey et al. [100] focuses particularly on spline-based methods and is used mostly to differentiate two groups of genes in expressed and non expressed genes. An open-source software called EDGE was implemented and is freely available. One of the difficulties with EDGE is its use in simulation procedures. Also, EDGE does not have an output with the list of genes in each group when using large data sets.

Serban and Wasserman [92] describe a technique for nonparametric estimation and cluster of a large number of curves. The technique is called CATS (Clustering After Transformation and Smoothing) and one of the main drawbacks in relation to PPCLUSTEL is that the technique showed to be an effective method only when the number of time points and/or the number of replications is large. In addition, CATS only allows the use of independent observations.

Ma et al. [62] describes another procedure that addresses spline-based methods for clustering of time course gene expression data. Besides the advantages outlined by the authors in their study, simulated curves in each group are very different in shape and easy to cluster. Additionally, the procedure was applied only to a large number of time points and small number of factor levels (150); and it is not clear if the procedure can be applied to HDLLSS data. A software called SSCLUST (smoothing spline clustering) and source code are freely available on the web.

All three previous studies are based on smoothing spline methods that are not recommended for data with small number of time points. This is not an issue with PPCLUSTEL.

A method that is not based on smoothing spline is proposed by Kim and Kim [55] and called DIB-C (difference-based clustering algorithm). DIB-C is an algorithm that works with first- and second-order differences between adjacent time points. The method does not use conventional time-series methods and, according to the authors, works with short and unevenly spaced time-course data. The main drawback is that DIB-C requires more replications and it is not effective with long time-course data. Finally, Angelini et al. [9] developed a functional Bayesian approach designed for time series microarray data. The main criticism of these ANOVA approachs being applied to time-course experiments is the fact that time variables are treated as particular experimental factors. The authors cited studies including Kerr et al. [53], Wu et al. [115], DiCamillo et al. [22], Smyth [96] and others as examples of studies where the temporal biological structure of data is ignored. PPCLUSTEL could also be included in this list. However, the main objective in PPCLUSTEL is not to describe the evolution in gene expression over time, but to cluster genes that are similarly expressed over time. More research about the impact of ignoring temporal structure in clustering of gene expression data should be investigated. A major drawback in Angelini et al. [9] is the fact that the method requires relatively large number of time points.

Overall, PPCLUSTEL was found to be highly efficient in clustering of simulated data with small or large numbers of time points. Comparison with other methods in simulated and real-life datasets should be investigated in future research.

7

Conclusions

Application

Applications using high dimensional low sample size (HDLSS) data and high dimensional longitudinal low sample size (HDLLSS) data are common in current research problems due to the improvements in data collection technologies that allow the obtaining of information from a large number of variables (objects, factor levels, parameters) at once but at a cost or time that restricts the use or observation of many replications. Examples of applications with HDLSS and HDLLSS data include: microarray experimentation, with data collected for thousands of genes with few replications per gene; spectrometry studies, with data collected for thousands of objects in few replications over a short period of time; pattern recognition, with analysis of thousands of voxels for a small number of images; agriculture screening trials, with large number of treatments (cultivars, pesticides) in a complete block design limited to 3 or 4 blocks; and broadband sonars, with detection of high dimensional signals for small, time consuming acquisition, and expensive datasets.

Problem

In the context of microarray gene expression analysis, clustering has been one of the most important statistical learning technologies used to identify groups of differentially expressed genes, [28], [7], [72], [118], [49], [47], [33]. Most of the current implemented clustering algorithms for microarray data are adaptations from standard clustering algorithms used in traditional multivariate analysis, where the number of factor levels is not high and sample sizes are relatively large. As a drawback it is observed that these algorithms cannot deal with the high dimensionality of data, requiring some dimension reduction prior to the analysis or working with comparison of pairs of factor levels in clustering. This dimension reduction is criticized by authors like Yeung and Ruzzo [118]. Further, the use of multiple comparisons results in a drastic reduction of the Type I error rate of the test, and requires the use of alternative solutions as suggested by Benjamini and Hochberg [11]. Another problem from traditional algorithms is the requirement of more replications or time points than usually available in HDLSS and HDLLSS data problems. In longitudinal applications, usually there is an additional requirement of evenly spaced time-points when using traditional time series or smoothing techniques. Some algorithms cannot deal with unbalanced data, requiring the complete elimination of a factor level from the study or the use of some imputation technique. Additionally, many traditional algorithms require a series of assumptions to be met by the data, such as constant variance and distributional assumptions, and most of the algorithms are sensitive to monotone transformations, producing different results when applied to data after some transformation has been made.

Solution to HDLSS Problem

PPCLUST is a novice computational algorithm for partition clustering of large number of factors with small number of observations per factor. The algorithm uses the one-way nonparametric ANOVA model based on ranks developed by Wang and Akritas [112]. The new procedure uses the p-value of the test statistic as a measure of similarity between groups,

i.e., as an indication of degree of alike between factors in a group.

Type I error rates indicate that the test statistic used in PPCLUST performs well in testing the hypothesis of no distributional difference between factor levels, regardless of whether the distribution is symmetric (normal), or skewed (lognormal, exponential), or heavy tailed (Cauchy). The only assumption required by PPCLUST is that observations are independent and from any arbitrary distribution.

Results and Advantages

PPCLUST outperformed 10 other benchmark algorithms described in the microarray literature and was shown to have many interesting properties that make the algorithm appealing for use with HDLSS data, including invariance to monotone transformations, automatic specification of number of groups, high quality in grouping variables, no multiple comparison problems, no need of dimension reduction in data, no requirement of balanced data, fast and easy to use. In simulations, the only competitive algorithm was MCLUST, that is a model-based algorithm which uses a mixture of normal distributions. However, MCLUST showed inferior performance with very small sample sizes or when groups of factor levels are asymmetrically distributed.

Solution to HDLLSS Problem

When HDLSS data are observed over time, PPCLUST is not applicable for clustering of factor levels. For this purpose, the asymptotic distribution of a new test statistic for testing of no simple effect of group of factor levels was obtained, or equivalently, no main effect of group and no group/time interaction effect when data have many factor levels replicated over time, i.e., when data are HDLLSS. Once this statistic was obtained, PPCLUST algorithm was adapted to cluster factor levels using p-values from the new test statistic, as similarity measures. The new algorithm is named PPCLUSTEL.

Results and Advantages

Type I error rates indicated that the test statistic used in PPCLUSTEL algorithm performs well for both small and relatively large number of time points, given only a small number of replications. However, it was observed that the algorithm is very sensitive to changes in the variance-covariance matrix. When small covariances are considered, the Type I error increases considerably. Also, the test was found to be extremely powerful in detecting small differences even with only 50 factor levels (2.5% of total levels) being different in data with 2000 factor levels.

In simulations, PPCLUSTEL was very efficient and stable in clustering factor levels. However, MCLUST was more stable and efficient in simulations with multivariate normal distributions than PPCLUSTEL. Nevertheless, PPCLUSTEL is an excellent alternative when MCLUST is not available or in situations when MCLUST does not produce any result due to memory allocation problems. PPCLUSTEL can also be used as an auxiliary tool to MCLUST in order to reduce the dimension of the problem and to indicate the possible number of groups available in data. Additionally, most of the advantages observed in PPCLUST are also obtained in PPCLUSTEL, including: automatic specification of number of groups, high quality in grouping variables, no multiple comparison problems, no need of dimension reduction, fast and easy to use. In simulations, PPCLUSTEL took less than a minute to complete the clustering of a large dataset with as many as 5000 factor levels. PPCLUSTEL is also very easy to use and the test developed for longitudinal data does not require the data to be equally spaced over time. One drawback with PPCLUSTEL, is that it is not invariant to monotone transformations in data, since it uses the original observations for computation of the test statistic of no simple effect.

7.1 Future Research

This thesis implements two algorithms, PPCLUST and PPCLUSTEL, for the clustering of HDLSS data and HDLLSS data. Although many of the issues with existing algorithms are

not observed with the new algorithms, several points require further research.

Other algorithms have been used with relative success for specific situations in clustering of replicated microarray gene expression data. Examples are gene-shaving (Hastie et al. [42], K-A. Do et al. [23], etc.), density-based hierarchical clustering (Jiang et al. [48]), clustering via iterative feature filtering or CLIFF (Xing and Karp [116]), plaid models (Lazzeroni and Owen [60]), subspace clustering (Parsons et al. [77]), coupled two-way clustering analysis of gene microarray data (Getz et al. [37]) and others. However, comparisons with those methods in simulations were not directly possible due to lack of flexibility in implemented software (no code access), poor documentation or/and incompatibility with different operational system platforms and versions of statistical packages. Efforts in comparing such algorithms with PPCLUST should be made, once the literature about many of these algorithms report excellent results and some advantages related to traditional algorithms.

PPCLUSTEL should be investigated with real-life datasets and also compared with other recent techniques for HDLLSS data, as source codes and user-friendly programs are made available by the authors or by the scientific community. Additionally, since the p-values used in PPCLUSTEL are from a test statistic based on original observations, it is important to compare the performance of PPCLUSTEL in HDLLSS data with different distributions, such as double exponential, Cauchy, lognormal. The asymptotic distribution for testing of no simple effect of group of factor levels using ranks instead of original observations, should also be investigated. This could make PPCLUSTEL invariant to monotone transformations and, might result in better performance under different distributions.

Finally, both algorithms, PPCLUST and PPCLUSTEL, were implemented in SAS© macro language. SAS© was found to be very flexible and capable of performing the clustering of very large datasets in relatively modest machines. However, SAS© is not an open source, or free program, making it difficult to access its capabilities in some cases. One direction of future research should be the implementation of PPCLUST and PPCLUSTEL in other computer languages, such as R software. At this moment R has serious problems

with memory allocation when performing analysis of high dimensional data.

PPCLUST and PPCLUSTEL proved to be powerful and efficient algorithms in clustering of HDLSS and HDLLSS data. This thesis addresses the specific application of microarray gene expression; however there are many important different applications that can take advantage of the features of PPCLUST and PPCLUSTEL.

Bibliography

[1] J. Aach and G. M. Church, *Aligning gene expression time series with time warping algorithms*, Bioinformatics **17** (2001), 495–508.

[2] M. G. Akritas and S. Arnold, *Asymptotics for analysis of variance when the number of levels is large*, Journal of The American Statistical Association **95** (2000), no. 449, 212–226.

[3] M. G. Akritas and N. Papadatos, *Heterocedastic one-way ANOVA and lack-of-fit tests*, Journal of The American Statistical Association **99** (2004), no. 466, 368–382.

[4] C. J. Albers, R. C. Jansen, J. Kok, O. P. Kuipers, and S. A. V. Hijum, *Simage: Simulation of DNA-microarray gene expression data*, BMC Bioinformatics **7** (2006).

[5] Colon Cancer Alliance, *Colorectal cancer: facts + figures*, www.ccalliance.org, 2006.

[6] D. B. Allison, G. P. Page, T. M. Beasley, and J. W. Edwards (eds.), *DNA microarrays and related genomics techniques: design, analysis, and interpretation of experiments*, Chapman and Hall/CRC, 2006.

[7] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays*, Proc. Natl. Acad. Sci. USA **96** (1999), 6745–6750.

[8] O. Alter, P. O. Brown, and D. Botstein, *Singular value decomposition for genome-wide expression data processing and modeling*, PNAS USA **97** (2000), 10101–10106.

[9] C. Angelini, D. De Canditis, M. Mutarelli, and M. Pensky, *A bayesian approach to estimation and testing in time-course microarray experiments*, Statistical Applications in Genetics and Molecular Biology **6** (2007).

[10] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, and J. T. Eppig et. al, *Gene ontology: tool for the unification of biology. the gene ontology consortium.*, Nature Genetics **25** (2000), 25–29.

[11] Y. Benjamini and Y. Hochberg, *Controling the false discovery rate: a practical and powerful approach to multiple testing*, JRSSB **57** (1995), 289–300.

[12] M. J. A. Berry and G. Linoff, *Data mining techniques for marketing, sales and customer support*, John Wiley and Sons, 1997.

[13] D. D. Boos and C. Brownie, *ANOVA and rank tests when the number of treatment is large*, Statistics & Probability Letters **23** (1995), 183–191.

[14] C. Brownie and D. D. Boos, *Type I error robustness of ANOVA and ANOVA on ranks when the number of treatments is large*, Biometrics **50** (1994), 542–549.

[15] A. J. Butte and I. S. Kohane, *Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements*, Pac. Symp. Biocomput. (2000), 418–429.

[16] R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrieian, D. Landsman, D. J. Lockhart, and R. W. Davis, *A genome-wide transcriptional analysis of the mitotic cell cycle*, Molecular Cell **2** (1998), 65–73.

[17] C. Christiansen, *Virtual colonoscopy: the reality of colon polyp and colorectal cancer screening*, Chance Magazine **19** (2006), no. 2, 57–60.

[18] S. Cooper, *Cell cycle analysis and microarrays*, TRENDS in Genetics **18** (2002), 289–290.

[19] A. de la Fuente, N. Bing, I. Hoeschele, and P. Mendes, *Discovery of meaningful associations in genomic data using partial correlation coefficients*, Bioinformatics **20** (2004), 3565–3574.

[20] M. Dettling and P. Bühlmann, *Boosting for tumor classification with gene expression data*, Bioinformatics **19** (2003), no. 9, 1061–1069.

[21] P. D'haeseleer, X. Wen, S. Fuhman, and R. Somogyi, *Mining the gene expression matrix: inferring gene relationships from large scale gene expression data.*, Proceedings of the second international workshop on information processing in cell and tissues (R. Patton and M. Holcombe, eds.), Plenum Publishing, 1998, pp. 203–212.

[22] B. DiCamillo, F. Sanchez-Cabo, G. Toffolo, S. K. Nair, Z. Trajanosky, and C. Cobelli, *A quantization method based on threshold optimization for microarray short time series*, BMC Bioinformatics **6** (2005).

[23] K-A. Do, G. J. McLachlan, R. Bean, and S. Wen, *Application of gene shaving and mixture models to cluster microarray gene expression data*, Cancer Informatics **2** (2007), 25–43.

[24] B. P. Durbin, J. S. Hardin, D. M. Hawkins, and D. M. Rocke, *A variance-stabilizing transformation for gene expression microarray data*, Bioinformatics **18** (2002), S105–S110.

[25] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, *Cluster analysis and display of genome-wide expression patterns*, PNAS-USA **95** (1998), 14863–14868.

[26] K. Fellenberg, N. C. Hauser, B. Bros, A. Neutzner, J. D. Hoheisel, and M. Vingron, *Correspondence analysis applied to microarray data*, PNAS USA **98** (2001), 10781–10786.

[27] D. G. Fisher and P. Hoffman, *The ajusted rand statistic: a SAS macro*, Psychometrika **53** (1988), 417–423.

[28] C. Fraley, *Algorithms for model-based gaussian hierarquical clustering*, SIAM **20** (1998).

[29] C. Fraley and A. Raftery, *Bayesian regularization for normal mixture estimation and model-based clustering*, Tech. report, University of Washington, 2005.

[30] C. Fraley and A. E. Raftery, *Mclust version 3.0: an R package for normal mixture modeling and model-based clustering*, Tech. report, University of Washington, 2006.

[31] J. Fridlyand and S. Dudoit, *Applications of resampling methods to estimate the number of clusters and to improve the accuracy of clustering method*, Tech. Report 600, Berkeley, 2001.

[32] N. Friedman, I. Nachman, and D. Pe'er, *Using bayesian networks to analyze expression data*, Journal of Computational Biology **7** (2000), 601–620.

[33] L. Fu and E. Medico, *Flame, a novel fuzzy clustering method for the analysis of DNA microarray data.*, BMC Bioinformatics **8** (2007).

[34] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms and applications*, SIAM, 2007.

[35] G. E. Garrigues, D. R. Cho, H. E. Rubash, S. R. Goldring, J. H. Herndorn, and A. S. Shanbhag, *Gene expression clustering using self-organizing maps: analysis of the macrophage response to particulate biomaterials*, Biomaterials **26** (2005), 2933–2945.

[36] E. Geraque, *Pequenos grandes arranjos*, Scientific American Brazil **16** (2006), 56–57.

[37] G. Getz, E. Levine, and E. Domany, *Coupled two-way clustering analysis of gene microarray data*, PNAS **97** (2000), 12079–12084.

[38] R. Gottardo, J. A. Pannucci, C. R. Kuske, and T. Brettin, *Statistical analysis of microarrray data: a bayesian approach*, Biostatistics **4** (2003), 597–620.

[39] R. Gottardo, A. E. Raftery, K. Y. Yeung, and R. E. Bumgarner, *Bayesian robust inference for differential gene expression in microarrays with multiple samples*, Biometrics **62** (2006), 10–18.

[40] G. Grant, S. Sokolovsky, and C. Stoechert, *Performance analysis of differential expression prediction algorithms using simulated array data*, Technical Report, University of Pennsylvania.

[41] J. Hardin, *Microarray data from a statistician's point of view*, Stats: the magazine for students of statistics **42** (2005), 4–13.

[42] T. Hastie, R. Tibshirani, M. B. Eisen, A. Alizadeh, R. Levy, L. Staudt, W. C. Chan, D. Botstein, and P. Brown, *'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns*, Genome Biology **1** (2001).

[43] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning (data mining, inference and prediction)*, 2001.

[44] L. J. Heyer, S. Kruglyak, and S. Yooseph, *Exploring expression data: identification and analysis of coexpressed genes*, Genome Research **9** (1999), 1106–11115.

[45] N. S. Holter, M. Mitra, A. Maritan, M. Cieplak, J. R. Banavar, and N. V. Fedoroff, *Fundamental patterns underlying gene expression profiles: simplicity from complexity*, PNAS USA **97** (2000), 8409–8414.

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

[47] C. Huttenhower, A. I. Flamholz, J. N. Landis, S. Sahi, C. L. Myers, K. L. Olszewski, M. A. Hibbs, N. O. Siemens, O. G. Troyanskaya, and H. A. Coller, *Nearest neighbor networks: clustering expression data based on gene neighborhoods*, BMC Bioinformatics **8** (2007).

[48] D. Jiang, J. Pei, and A. Zhang, *Dhc: a density-based hierarchical clustering method for time-series gene expression data*, Proc. BIBE2003: Third IEEE int'l Symp. Bioinformatics and Bioeng., 2003.

[49] D. Jiang, C. Tang, and A. Zhang, *Cluster analysis for gene expression data: a survey*, IEEE Transactions on Knowledge and Data Engineering **16** (2004), 1370–1386.

[50] D. E. Johnson, *Applied multivariate methods for data analysts*, Duxbury, 1998.

[51] R. A. Johnson and D. W. Wichern, *Applied multivariate statistical analysis*, Prentice Hall, 2002.

[52] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: An introduction to cluster analysis*, Wiley Interscience, 1990.

[53] M. K. Kerr, M. Martin, and G. A. Churchill, *Analysis of variance for gene expression microarray data*, J. Computational Biology **7** (2000), 819–837.

[54] R. Khattree and D. N. Naik, *Multivariate data reduction and discrimination with SAS software*, SAS Institute, 2000.

[55] J. Kim and J. H. Kim, *Difference-based clustering of short time-course microarray data with replicates*, BMC Bioinformatics **8** (2007).

[56] H. Kishino and P. J. Waddell, *Correspondence analysis of genes and tissue types and finding genetic links from microarray data*, Genome Informatics **11** (2000), 83–95, In A. Dunker, A. Konagaya, S. Miyano, and T. Takagi editors. Tokyo Universal Academy Press.

[57] R. R. Klevecz, *Dynamic archicheture of the yeast cell cycle uncovered by wavelet decomposition of expression microarray data*, Funct Integr Genomics **1** (2000), 186:192.

[58] S. Knudsen, *Guide to analysis of DNA microarray data*, Wiley-LISS, 2004.

[59] T. Kohonen, *Self-organization and associative memory*, Springer, 1989.

[60] L. Lazzeroni and A. Owen, *Plaid models for gene expression data*, Statistica Sinica **12** (2002), 61–86.

[61] I. Lönnstedt and T. Speed, *Replicated microarray data*, Statistica Sinica **12** (2002), 31–46.

[62] P. Ma, C. I. Castillo-Davis, W. Zhong, and J. S. Liu, *A data-driven clustering method for time course gene expression data*, Nucleic Acids Research **34** (2006), 1261–1269.

[63] P. M. Magwene and J. Kim, *Estimating genomic coexpression networks using first-order conditional independence*, Genome Biology **5** (2004).

[64] K. V. Mardia, J. T. Kent, and J. M. Bibby, *Multivariate analysis*, 1979.

[65] G. McLachlan and D. Peel, *Finite mixture models*, Wiley, 2000.

[66] G. J. McLachlan, R. W. Bean, and L. Ben-Tovim Jones, *A simple implementation of a normal mixture approach to differential gene expression in multiclass microarrays*, Bioinformatics **22** (2006), no. 13, 1608–1615.

[67] G. J. McLachlan, K. A. Do, and C. Ambroise, *Analyzing microarray gene expression data*, Wiley-Interscience, 2004.

[68] J. B. McQueen, *Some methods for classification and analysis of multivariate observations*, Proceedings of fifth Berkeley symposium on mathematical statistics and probability, 1967.

[69] T. Mehta, M. Tanik, and D. B. Allison, *Towards sound epistemiological foundations of statistical methods for high-dimensional biology*, Nature Genetics **36** (2004), 943–947.

[70] P. Mendes, W. Sha, and K. Ye, *Artificial gene networks for objective comparison of analysis algorithms*, Bioinformatics **19 (Supplement)** (2003).

[71] G. A. Milliken, K. A. Garret, and S. E. Travers, *Experimental design for two-color microarrays applied in pre-existing split-plot experiment*, Statistical Applications in Genetics and Molecular Biology **6** (2007).

[72] D. A. Notterman, U. Alon, A. J. Sierk, and A. J. Levine, *Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays*, Cancer Research **61** (2001), 3124–3130.

[73] M. Nykter, T. Aho, M. Ahdesmaki, P. Ruusuvuori, A. Lehmussola, and O. Yli-Harja, *Simulation of microarray data with realistic characteristics*, BMC Bioinformatics **7** (2006).

[74] M. V. Osier, *DNA microarrays and related genomics techniques (design, analysis, and interpretation of experiments)*, ch. Postanalysis interpretation: "What do I do with this gene list?", pp. 321–333, Chapman and Hall, 2006.

[75] W. Pan, *A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments*, Bioinformatics **18** (2002), 546–554.

[76] G. Parmigiani, E. S. Garrett, R. A. Irizarry, and S. L. Zeger (eds.), *The analysis of gene expression data: methods and software*, Springer, 2003.

[77] L. Parsons, E. Haque, and H. Liu, *Subspace clustering for high dimensional data: a review*, Tech. report, Arizona State University, 2004.

[78] P. Pavlidis, *Using ANOVA for gene selection from microarray studies of the nervous system*, Elsevier Science: Methods **31** (2003), 282–289.

[79] Y. Pawitan, *In all likelihood: statistical modeling and inference using likelihood*, Oxford, 2001.

[80] Mediline Plus, *Medical encyclopedia: colon cancer*, www.nlm.nih.gov/medlineplus, 2006.

[81] S. Pomeroy, P. Tamayo, M. Gaasenbeek, L. Sturla, M. Angelo, M. McLaughlin, J. Kim, L. Goumnerova, P. Black, C. Lau, and et al., *Prediction of central nervous system ebryonal tumor outcome based on gene expression*, Nature **415** (2002), 436–442.

[82] E. Purdom and S. P. Holmes, *Error distribution for gene expression data*, Statistical Applications in Genetics and Molecular Biology **4** (2005).

[83] S. Reese, G. Sukthankar, and R. Sukthankar, *An efficient recognition technique for minelike objects using nearest-neighbor classification*, Tech. report, Intel Corportation, 2003.

[84] S. I. Resnick, *A probability path*, Birkhäuser, 2001.

[85] M. L. Rizzo and G. J. Szekely, *The energy package*, 2006, Energy R package under GPL 2.0 license or later.

[86] T. Roy, *The effect of heterocedasticity and outliers on the permutation t-test*, J. Stat. Comput. Simul. **72** (2002), 23–26.

[87] C. Sabatti, *DNA microarrays and related genomics techniques: design, analysis, and interpretation of experiments*, ch. False discovery rate and multiple comparison procedures, pp. 289–304, Chapman & Hall/CRC, 2006.

[88] G. Schwarz, *Estimating the dimension of a model*, Annals of Statistics **6** (1978), 461–464.

[89] H. Schwender, S. Rabstein, and K. Ickstadt, *Do you speak genomish?*, Chance **19** (2006), 3–10.

[90] E. Segal, H. Wang, and D. Koller, *Discovering molecular pathways from protein interaction and gene expression data*, Bioinformatics **19** (2003), 264–272.

[91] E. Segal, R. Yelensky, and D. Koller, *Genome-wide discovery of transcriptional modules from DNA sequence and gene expression.*, Bioinformatics **19** (2003), 273–282.

[92] N. Serban and L. Wasserman, *Cats: clustering after transformation and smoothing*, JASA **100** (2005), 990–999.

[93] J. S. Shaik and M. Yeasin, *A unified framework for finding differentially expressed genes from microarray experiments*, BMC Bioinformatics **8** (2007).

[94] K. Shedden and S. Cooper, *Analysis of cell-cycle gene expression in saccharomyces cerevisae using microarrays and multiple synchronization methods*, Nucleic Acids Research **30** (2002), 2920–2929.

[95] R. M. Simon, E. L. Korn, L. M. McShanne, M. D. Rdmacher, G. W. Wright, and Y. Zhao, *Design and analysis of DNA microarray investigations*, 2004.

[96] G. K. Smyth, *Bioinformatics and computational biology solutions using R and Bioconductor*, ch. Limma: Linear models for microarry data, pp. 397–420, Springer, 2005.

[97] American Cancer Society, *What is colorectal cancer*, www.cancer.org, 2006.

[98] T. Speed (ed.), *Statistical analysis of gene expresssion microarray data*, Chapman and Hall, 2003.

[99] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, *Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarrray hybridization*, Molecular Biology **9** (1998), 3273–3297.

[100] J. D. Storey, W. Xiao, J. T. Leek, R. G. Tompkins, and R. W. Davis, *Significance analysis of time course microarray experiments*, PNAS **102** (2005), 12837–12842.

[101] P. Sykacek, R. A. Furlong, and G. Micklem, *A friendly statistics package for microarray analysis*, Bioinformatics **21** (2005), 4069–4070.

[102] G. J. Székely and M. L. Rizzo, *Hierarchical clustering via joint between-within distances: extending Ward's minimum variance method*, Journal of Classification **22** (2005), 151–183.

[103] P. Tamayo and S. Ramaswamy, *Cancer genomics and molecular pattern recognition*, Tech. report, MIT, Cambridge and Harvard, 2002.

[104] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub, *Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation*, PNAS USA **96** (1999), 2907–2912.

[105] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church, *Systematic determination of genetic network architecture*, Nature Genetics **22** (1999), 281–285.

[106] S. Theodoridis and K. Koutroumbas, *Pattern recognition*, third ed., Academic Press, 2006.

[107] H. Thiele, *Mass spectometry and bioinformatics in proteomics*, Chance **16** (2003), 29–36.

[108] P. Törönen, M. Kolehmainen, G. Wong, and E. Castrén, *Analysis of gene expression data using self-organizing maps*, FEBS Letters **451** (1999), 142–146.

[109] A. W. van der Vaart, *Asymptotic statistics*, Cambridge University Press, 1998.

[110] A. B. van't Wout, G. k. Lehrman, S. A. Mikheeva, G. C. O'Keeffe, M. G. Katze, R. E. Bumgarner, G. K. Geiss, and J. I. Mullins, *Cellular gene expression upon human immunodeficiency virus type 1 infection of cd4$^+$-t-cell lines*, Journal of Virology **77** (2003), no. 2, 1393–1402.

[111] H. Wang, *Testing in multi-factor heteroscedastic ANOVA and repeated measures designs with large number of levels*, Ph.D. thesis, The Pennsylvania State University, 2004.

[112] H. Wang and M. G. Akritas, *Rank tests for ANOVA with large number of factor levels*, Nonparametric Statistics **16** (2004), 563–589.

[113] H. Wang, J. Neill, and F. Miller, *Curve clustering based on nonparametric hypothesis*, Unpublished manuscript.

[114] J. D. Watson and A. Berry, *DNA: The secret of life*, Alfred A. Knopf, 2003.

[115] H. Wu, M. K. Kerr, X. Cui, and G. A. Churchill, *The analysis of gene expression data (methods and software)*, ch. MAANOVA: A software package for the analysis of spotted cDNA microarray experiments, pp. 313–341, Springer, 2003.

[116] E. P. Xing and R. M. Karp, *Cliff: clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts*, Bioinformatics **17** (2001), 306–315.

[117] N. Yano and M. Kotani, *Clustering gene expression data using self-organizing maps and k-means clustering*, SCIE (2003).

[118] K. Y. Yeung and W. L. Ruzzo, *Principal component analysis for clustering gene expression data*, Bioinformatics **9** (2001), 763–774.

[119] S. O. Zakharkin, T. Mehta, M. Tanik, and D. B. Allison, *DNA microarrays and related genomics techniques (design, analysis, and interpretation of experiments)*, ch. Epistemological foundations of statistical methods for high-dimensional biology, pp. 57–75, Chapman and Hall, 2006.

[120] L. P. Zhao, R. Prentice, and L. Breeden, *Statistical modeling of large microarray data sets to identify stimulus-response profiles*, PNAS USA **98** (2001), 5631–5636.

Appendix A

Basics of Microarray Technology

Scientists are studying complex disorders through the identification of components of the interactions of genes. A gene is a specific segment of a deoxyribonucleic acid (DNA) molecule contained within chromosomes in the nucleus of each cell [1] and control the production of proteins and ribonucleic acid (RNA) molecule.

Two huge interwined strands of DNA form a chain that is called a chromosome. The strands of DNA bond together through the base pairing of four nitrogen bases known as adenine (A), thymine (T), cythosine (C), and guanine (G) in a double helix molecule structure as described by James D. Watson and Francis Crick in 1953 [2] and illustrated in Fig. A.1.

The protein production from genes involves the stages of **transcription** and **translation** that is the Central Dogma of Molecular Biology, as represented in Fig. A.2. Transcription is the communication of a genetic code from DNA to RNA through the synthesis of a strand of RNA that has sequences of bases complementary to that of the DNA strand. Translation is the process of synthesizing polypeptide (compound of amino acids) chains from an intermediary form of DNA known as messanger RNA (mRNA). The complementary DNA (cDNA) is a form of DNA that possesses the complementary bases of the mRNA. The amino acids synthesized by one mRNA strand form a chain that folds into a protein.

[1] The nucleus of almost every cell includes the complete human genome that is a representation of our entire gene complement, i.e., about 30,000 genes in human DNA already provided by the Human Genome Project.

[2] An interesting text about the history of discovery of the Double Helix structure is Watson and Berry [114].

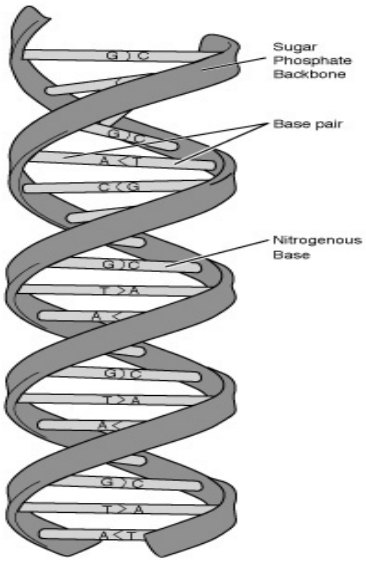

Figure A.1. *DNA double helix molecule base pairing schematic (Access Excellence at the National Health Museum, www.accessexcellence.org).*

Microarray technology utilizes properties of pairing bonding in DNA for a process of joining two complementary strands of DNA, known as the hybridization process, using pieces of labeled DNA or RNA (called probes) to measure the expression of a gene. The molecule of mRNA is relatively fragile and can be broken down by the action of enzimes in laboratory. The creation of cDNA from mRNA is known reverse transcription and is used in microarray technology to create short sequential segments called EST (Expressed Sequence Tag) to represent the coding portion of a gene and as a substitute to a technique that "amplifies" or replicates DNA fragments (PCR - Polymerase Chain Reaction).

To better understand how DNA microarray experiments are performed the following section presents a synthesis of the microarray procedure reproduced from a flash animation developed by Malcolm Campbell at Davidson College. The complete animation can be found in *geat.davidson.edu/Pirelli/index.htm* [3].

[3]Copyright©University of North Carolina at Chapell Hill and A. Malcom Campbell. Slides cut and pasted from flash animation with authorization of the authors.

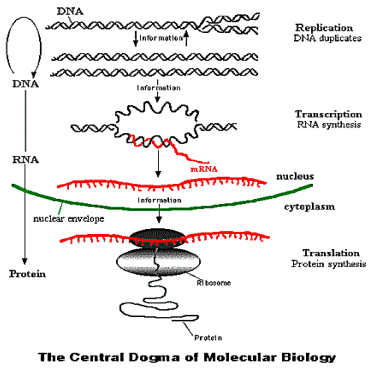

Figure A.2. *The Central Dogma of Molecular Biology (Access Excellence at the National Health Museum, www.accessexcellence.org).*

• Microarray Technology Steps

Microarrays have made it possible to measure genes that are induced or repressed when two populations of cells grow under different conditions. In microarrays the expression of the entire genome of one organism can be analysed in just one experiment. In past, the experiments allowed the observation of just some genes, making the process time consuming and very expensive. With advances in bioinformatics, mathematics, and statistics, DNA microarrays made (and continue to make) a revolution in molecular biology research.

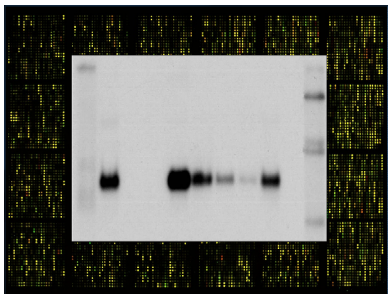

Figure A.3. *Nothern blot slide superposed to microarray slide. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 1** – Suppose two groups of cells are exposed to a control and experimental condition like aerobic and anaerobic, cancer and non cancer, etc. The objective is to compare any two cell types through the measure of expression of their genomes under the different conditions (Fig. A.4).

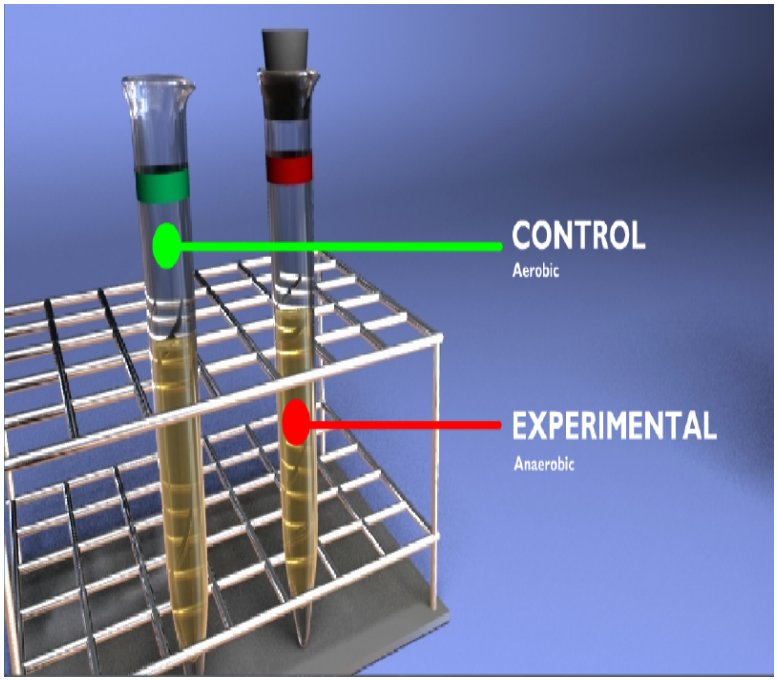

Figure A.4. *Groups of cells exposed to different conditions. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 2** – As the cells grow under the experimental condition, certain genes are repressed, others are induced, and some do not change their expression when compared to the same genes in the cells exposed to the control condition (Fig. A.5).

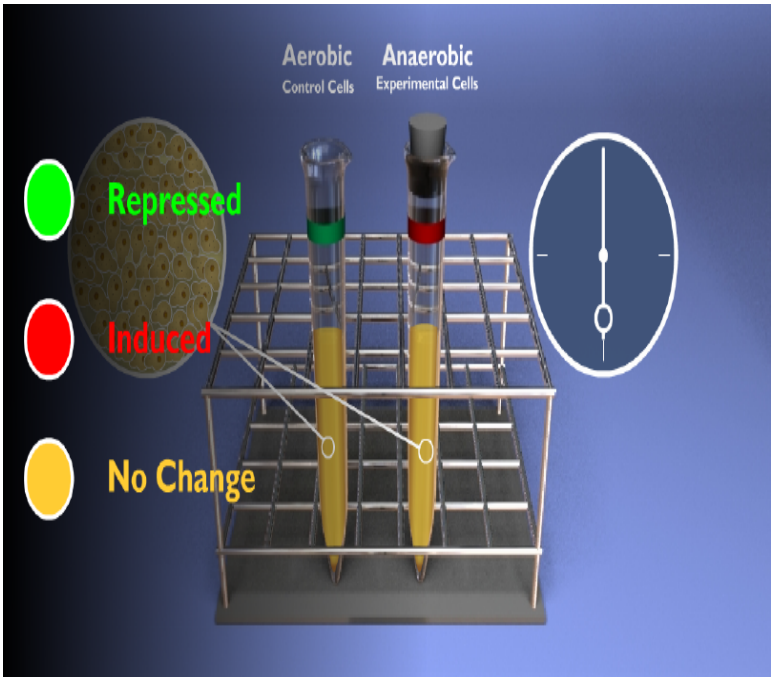

Figure A.5. *Growth of cells under two experimental conditions. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 3** – In order to measure the expression of a gene, a piece of labeled RNA or DNA, called probe, is used. The starting point is the generation of probes in the isolation of mRNA from both conditions, experimental and control. Millions of mRNA molecules are isolated. The population of mRNA molecules is different in each tube (Fig. A.6).

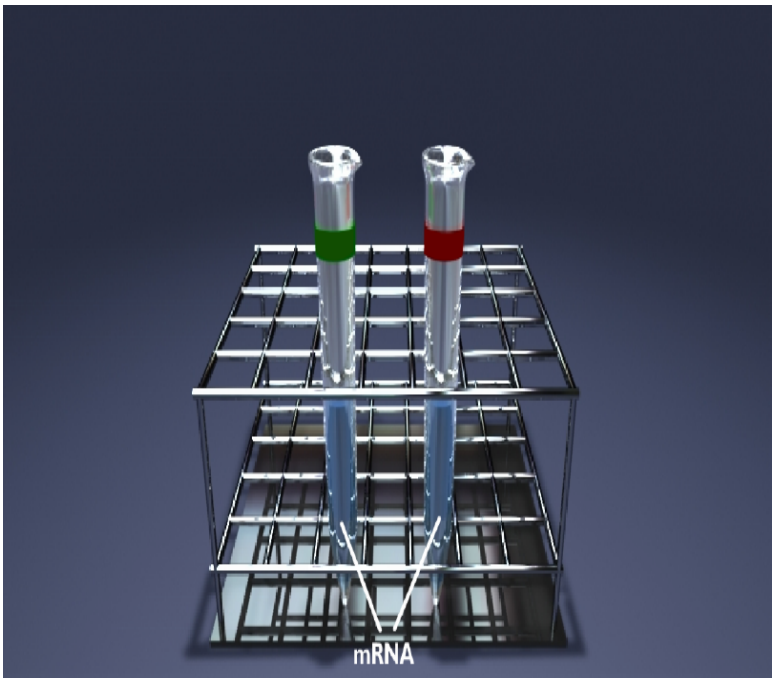

Figure A.6. *Isolation of mRNA. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 4** – Using an enzime, called reverse transcriptase, cDNAs are synthesized from mRNA templates. At this point green fluorescent dyes are used to label other cDNAs from genes transcribed under control. Red fluorescent dyes are used to label genes transcribed under the experimental condition. (Fig. A.7).

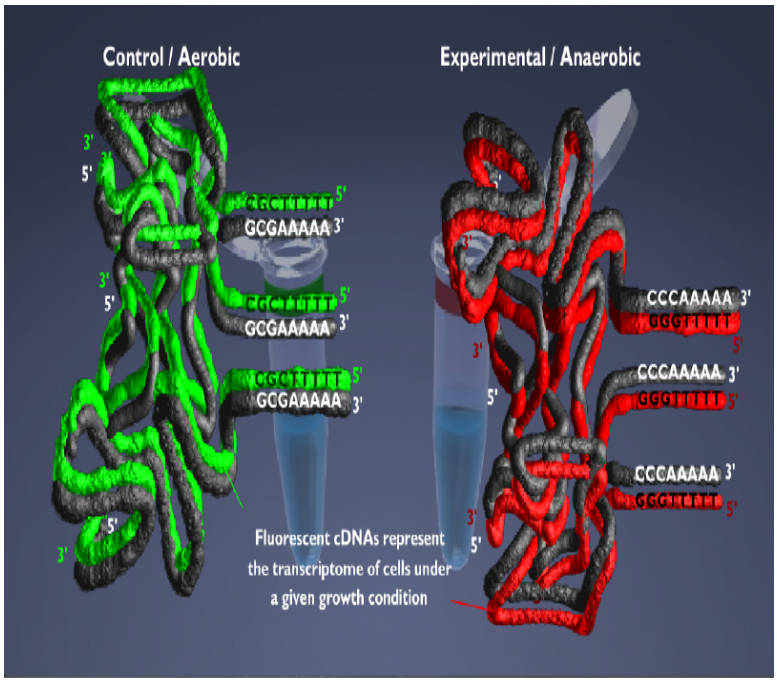

Figure A.7. *Green and red dyes in transcription process. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 5** – The cDNAs are complementary to the mRNA molecules from which they were made and also complementary to the antisense strand of the gene. (Fig. A.8).

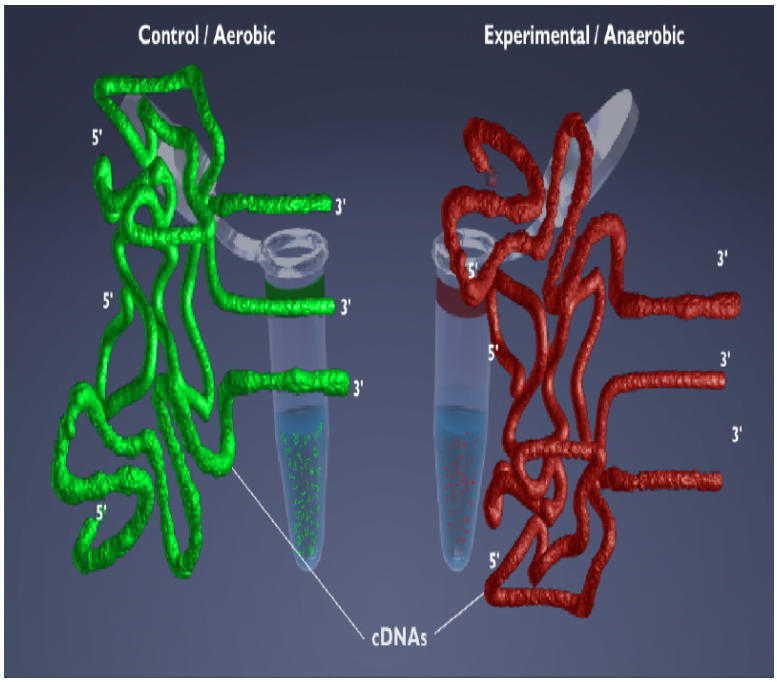

Figure A.8. *Labeled cDNA. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 6** – Combined cDNAs (red and green) are used to probe the microarray (Fig. A.9). In each spot of the microarray there is a portion of the coding sequence from a different gene of the material being analysed. Together all spots could represent an entire genome (human, yeast, etc.).

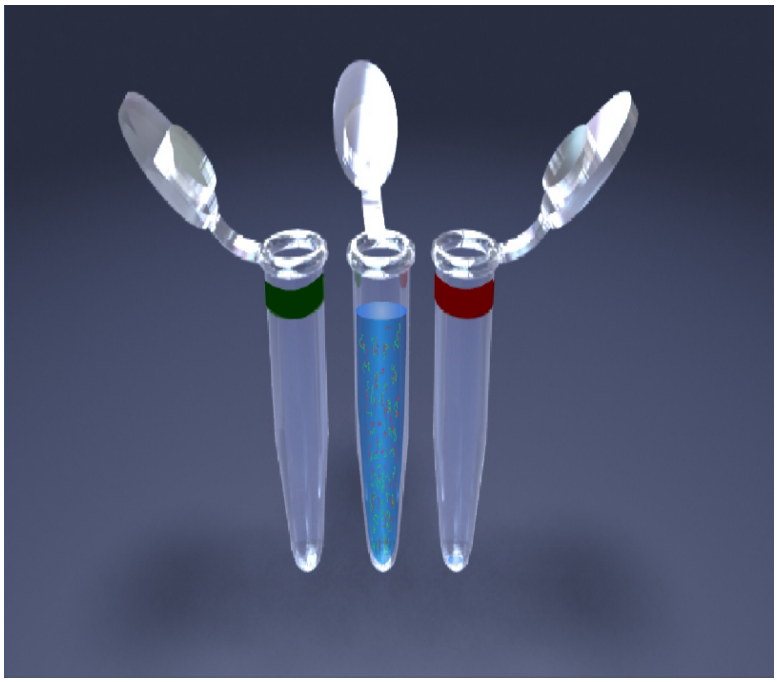

Figure A.9. *Combined cDNAs in preparation for hybridization process. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 7** – In the hybridization process, green and red cDNAs bind to DNA spotted on the microarray. The binding to a spot indicates if the expression of a gene on a specific spot was changed under the condition where cells were grown. The unbound cDNAs are washed off and the remaining will be visualized in the microarray (Fig. A.10).

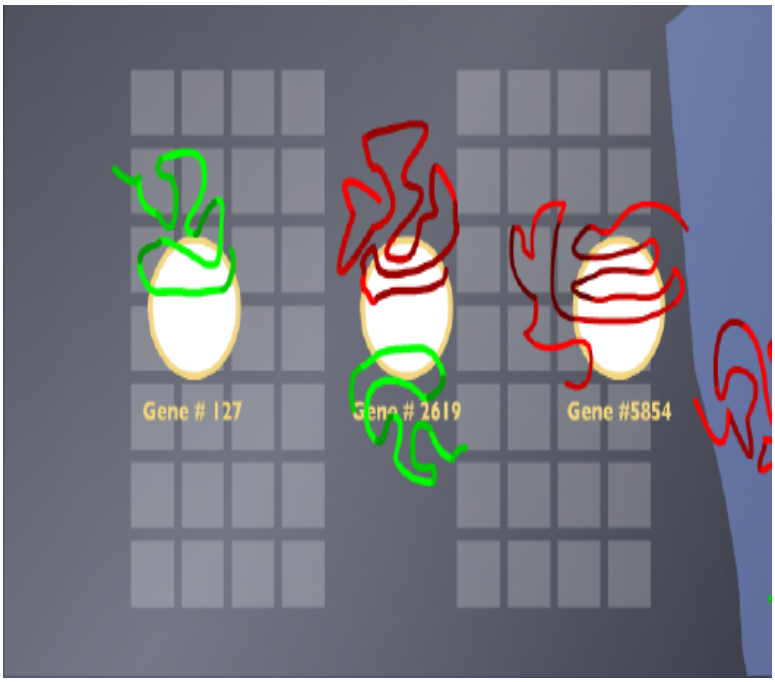

Figure A.10. *Washing off unbound cDNAs. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 8** – The microarray slide is scanned with two different lasers to detect the bound green and red cDNAs. If only green (red) cDNAs bound to a gene this will be indicated by green (red) laser only and means that the gene was only expressed in cells grown under control (experimental) condition. If the gene is expressed under both conditions, than a yellow spot will result from the merging of green and red bounded cDNAs (Fig. A.11).

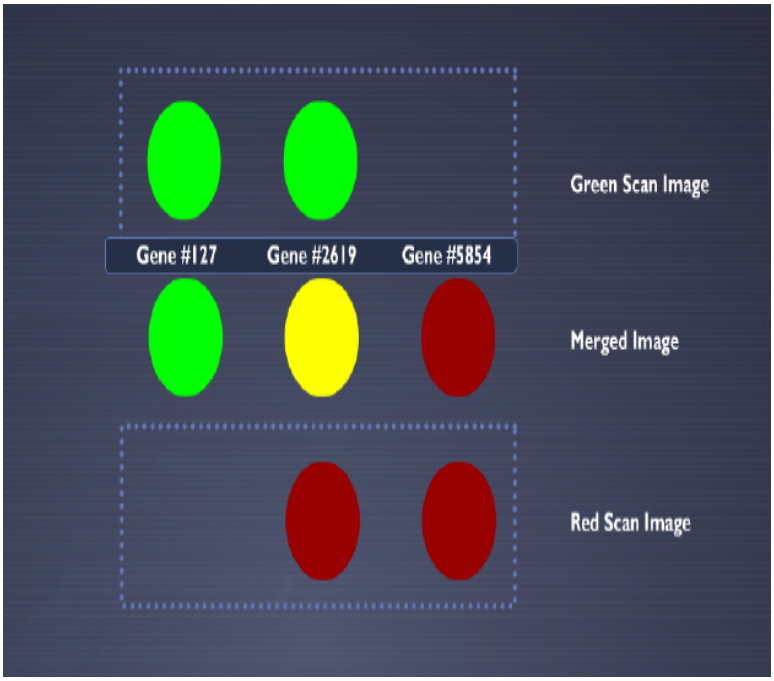

Figure A.11. *Merging of image from 3 scanned genes. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 9** – A final microarray will include different variations of intensity of red and green spots indicating levels of variations of expression of a gene. Black spots indicates that a gene was not expressed (not transcribed) under both conditions (Fig. A.12).

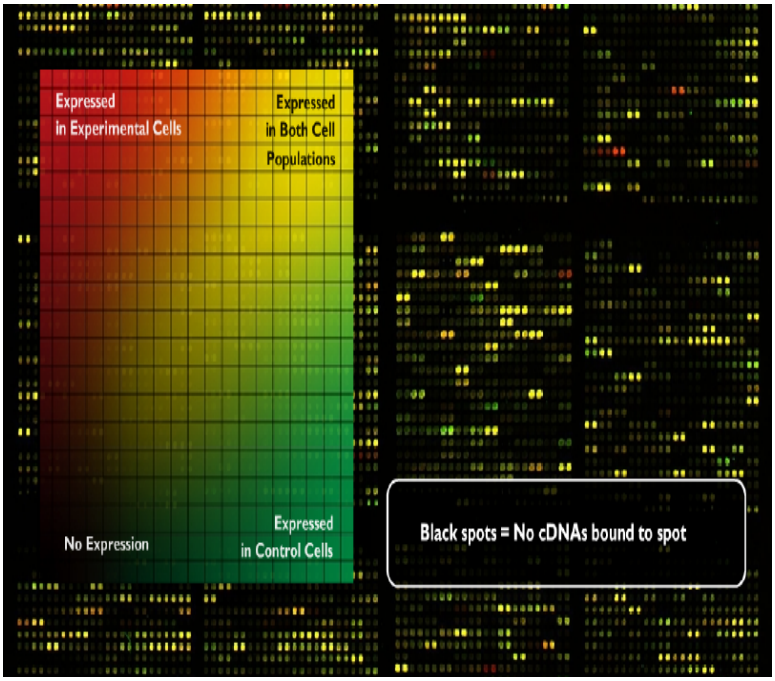

Figure A.12. *Different levels of gene expression in a microarray. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

- **Step 10** – Finally, in a computer, the different intensities of colors in a microarray (Fig. A.13) will be transformed to numbers that represent the final data of gene expression . From this step, procedures like clustering and classification will take place to explore the characteristics of the analysed genome. It will enable the researcher to diagnose genetic diseases, improve medical treatments, predict side effects from treatment, etc.

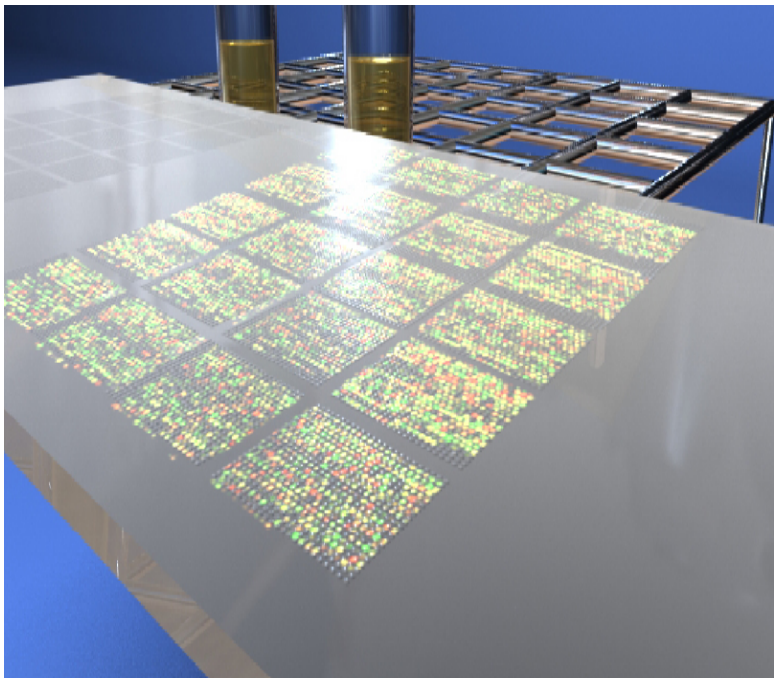

Figure A.13. *Microarray with expression of thousands of genes. Copyright© University of North Carolina at Chapell Hill and A. Malcom Campbell.*

Appendix B

SAS Macros

This appendix provides the SAS©[1] macros introduced in Chapters 3 and 6 to perform partition clustering of high dimensional low sample size (HDLSS) data and high dimensional longitudinal low sample size (HDLLSS) data. In addition, the appendix provides a SAS© macro code for the calculus of Adjusted Rand Index (ARI) from Hubert and Arabie [46]. The ARI macro was adapted from Fisher and Hoffman [27] and corrected for the latest version of SAS©. All macros presented here are compatible with SAS 9.1.3 or later.

The following sections present the SAS© source code and instructions for use of each macro using command lines. Before running each macro a proper SAS data set should be ready for analysis as indicated in each section bellow.

B.1 Macro PPCLUST for HDLSS data

PPCLUST is a SAS© macro for partitioning clustering of HDLSS data as described in Chapter 3 of this monograph.

The SAS data set for PPCLUST should be in the format presented in Table B.1, where Factor Level is an optional variable with the label of each factor. For example, in a microarray data each factor level can correspond to a gene name. The variables $X1, X2, \ldots, Xn$ represent the replications and should be named with a prefix name plus numbers in sequence. For example, in a data set with three replications, they could be denoted by $R1$, $R2$, and $R3$.

Command line: %macro ppclust(dataset,obsmin,obsmax,alpha);

Input commands: In this case **Dataset** is the SAS data name, **obsmin** is the name of the variable indicating the first replication, **obsmax** is the name of the variable indicating the last replication, and **alpha** is the threshold parameter to be compared with p-values in the clustering algorithm, as explained in Chapter 3.

Output: PPCLUST produces three temporary SAS data sets that can be saved or exported using common SAS commands. The data set called **grfr** has the number of factor levels in each created group, while the data set **groupclass** has a numerical identification for each factor

[1]SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. © indicates USA registration.

Table B.1. *High dimensional replicated data set layout. Here* $a \to \infty$ *and* $n_i \geq 2$.

Factor level	$X1$	$X2$	...	Xn
1	X_{11}	X_{12}	...	X_{1n_1}
2	X_{21}	X_{22}	...	X_{2n_2}
⋮	⋮	⋮	⋮	⋮
a	X_{a1}	X_{a2}	...	X_{an_a}

level (variable idobs) and the respective group of each factor level (variable group). The data set **datanew** has the original data with the information in data **groupclass** added to it.

The following command line starts PPCLUST in a data set called **adenoma** with four replications and threshold of 10^{-10}.

%ppclust(adenoma,d1,d4,1e-10);

B.1.1 SAS Code

```
%macro ppclust(dataset,obsmin,obsmax,alpha);

data datanew;
set &dataset;
idobs = _N_;
run;

proc iml;

start rankmiss(matrix1);
matrixm = matrix1 = .;
nmiss = matrixm[,+]; nmiss = nmiss[+,];
miss = -999999999999;
matrix1 = choose(matrix1=.,miss,matrix1);
matrix1 = ranktie(matrix1);
matrix1 = matrix1 - nmiss;
matrix1 = choose(matrix1<=0,.,matrix1);
finish;

start sortmed(matrix1,matrix2);
matrix3 = choose(matrix2=.,max(matrix1)+1000,matrix2);
medians = t(median(t(matrix3)));
call sortndx(matrix1,medians,{1});
```

```
matrix2 = matrix2[matrix1,];
free medians;
finish;

start sortcenter(matrix2,colm5,tst);
nr = nrow(matrix2);
nc = ncol(matrix2);
if tst = 1 then do;
m2 = matrix2;
end;
else if tst > 1 then do;
ed = tst-1;
m1 = matrix2[1:ed,];
m2 = matrix2[tst:nr,];
end;
nr2 = nrow(m2);
j1 = j(nr2,1) * 0;
c1 = int(0.40*nr2);
c2 = int(0.60*nr2);
call sort(m2,colm5);
do i = 1 to nr2;
if i < c1 then j1[i,] = 1;
if i > c2 then j1[i,] = 1;
end;
m2 = m2 || j1;
nc2 = nc + 1;
call sort(m2,nc2);
m2 = m2[,1:nc];
if tst = 1 then matrix2 = m2;
else if tst > 1 then matrix2 = m1 // m2;
finish;

start sig4hat(x,n,s4hat);
mx = sum(x)/n;
s4hat = (ssq(x-mx)/n)**2;
finish;

start sig4jack(x,sigma4);
n = nrow(x); ni = n-1;
if ni > 0 then do;
call sig4hat(x,n,s4hat);
s4jack = n * s4hat;
do  i=1 to n;
xni = t(remove(x,i));
call sig4hat(xni,ni,s4hatni);
```

```
s4jack = s4jack - (n-1)/n * s4hatni;
free s4hatni;
end;
sigma4 = s4jack;
end;
else if ni = 0 then sigma4 = 0;
finish;

start sigma4est(exp1,nrowe1,maxobs,sigma4e);
exp1 = colvec(exp1);
lfac = t(1:nrowe1) @ j(maxobs,1);
expi1 = exp1 ^= .;
lfac = lfac # expi1;
lfac = lfac[loc(lfac),];
exp1 = exp1[loc(exp1 ^= .),];
y = j(nrow(exp1),1);
sigma4e = j(nrowe1,1);
do i = 1 to nrowe1;
temp = choose(lfac=i,y=1,y=0) # exp1;
temp = temp[loc(temp),];
call sig4jack(temp,sigma4);
sigma4e[i,] = sigma4;
end;
free exp1 expi1 lfac y temp;
finish;

start anovarank(resp,nfac,maxobs,sigma4ea,pv);
respi = resp ^= .;
ng = respi[,+];
meang = resp[,+]/ng;
center= resp - repeat(meang,1,maxobs);
ss = center[,##];
varg = ss / (ng-1);
resp = colvec(resp);
j = j(maxobs,1);
free respi;
meant = sum(meang)/nfac;
difmt = meang - meant;
mst = ssq(meang - meant)/(nfac - 1);
mse2= sum(varg/ng) / nfac;
fr2 = mst/mse2;
tau2 = sum(sigma4ea/(ng#(ng-1))) * 2 / nfac;
asyvar = tau2 / mse2**2;
pv = 1 - probnorm(sqrt(nfac)*(fr2-1)/sqrt(asyvar));
finish;
```

```
start indtest(exp,st,limitn,alpha,nrowe,ncole,colgr,colts,cols4,g);
exp[,colts] = 0; count = 0;
do i=1 to nrowe;
if exp[i,colgr]=g then exp[i,colts] = 1;
end;
do i=st to nrowe;
exp[i,colts] = 1;
n = sum(exp[,colts]);
expt = exp[loc(exp[,colts]),];
call anovarank(expt[,1:ncole],n,ncole,expt[,cols4],pv);
if (pv > alpha) then exp[i,colgr] = g;
else if (pv <= alpha) then exp[i,colts] = 0;
end;
call sort(exp,colgr); * Included;
do i=1 to nrowe;
if exp[i,colgr] <= g then count = count + 1;
end;
st = count + 1;
finish;

use datanew;
read all var{idobs} into factor;
read all var("&obsmin":"&obsmax") into expression;
alpha = &alpha;
testpp = 0; g = 1;
ncole = ncol(expression);
nrowe = nrow(expression);
call rankmiss(expression);
call sortmed(factor,expression);
exp1 = expression;
call sigma4est(exp1,nrowe,ncole,sigma4e);
colid = ncole + 1;
colgr = ncole + 2;
colts = ncole + 3;
cols4 = ncole + 4;
colm5 = ncole + 5;
j1 = j(nrowe,1);
expression = expression || j1 # factor;
expression = expression || j1 * 9999;
expression = expression || j1 * 0;
expression = expression || j1 # sigma4e;
expression = expression || j1 # factor;
tstart = 1; tend = nrowe;
call sortcenter(expression,colm5,tstart);
```

```
call anovarank(expression[,1:ncole],nrowe,ncole,expression[,cols4],pv);
pv2 = pv;
if (pv > alpha) then expression[,colgr] = 1;
if (pv <= alpha) then do;
L = (tend - tstart + 1)/2;
tend = tstart + int(L-1);
do while (tstart <= nrowe);
if (pv2 > alpha) then g = g + 1;
expression[,colts] = 0;
do i = 1 to nrowe;
if (tstart <= i) then do;
if ( i <= tend) then expression[i,colts] = 1;
end;
end;
n = sum(expression[,colts]);
if (n = 1) then do;
expression[tstart,colgr] = 0;
tstart = tend + 1;
tend = nrowe;
testpp = -1;
end;
exp = expression[loc(expression[,colts]),];
if testpp = 0 then call anovarank(exp[,1:ncole],n,ncole,exp[,cols4],pv);
pv2 = pv;
if (pv > alpha) then do;
do i = 1 to nrowe;
if (tstart <= i) then do;
if ( i <= tend) then expression[i,colgr] = g;
end;
end;
tstart = tend + 1;
tend = nrowe;
st = tstart;  exp = expression;
call indtest(exp,st,limitn,alpha,nrowe,ncole,colgr,colts,cols4,g);
tstart = st;
expression = exp;
end;
else if (pv <= alpha) then do;
L = (tend - tstart + 1)*0.9;
tend = tstart + int(L-1);
testpp = 0;
end;
end;
end;
exp = expression[,colid];
```

```
exp = exp || expression[,colgr];
create groupclass var{idobs group};
append from exp;
free all;
quit;

proc sort data=datanew; by idobs;
proc sort data=groupclass; by idobs; run;

data groupclass groupclass0;
merge groupclass datanew;
by idobs;
if group = 0 then output groupclass0;
else if group ^= 0 then output groupclass;
run;

proc sort data=groupclass; by group; run;

data t1 (keep = idobs group &obsmin - &obsmax);
set groupclass;

data t1 (drop = &obsmin - &obsmax);
set t1;
array num(*) &obsmin - &obsmax;
length obst $ 8;
do i=1 to dim(num);
expression = num[i];
call vname(num[i],obst);
output;
end;
run;

proc means data=t1 noprint;
var expression;
by group;
output out=t2 median=median;
run;

data t2 (keep=group median);
set t2;
run;

proc sort data=t2; by median; run;

data t2 (keep=group corder);
```

```
set t2; corder = _N_;
run;

proc sort data=t2; by group; run;

data groupclass(drop=corder);
merge groupclass t2;
by group;
group = corder;

data groupclass;
set groupclass;
if idobs=lag(idobs) then delete;
run;

data groupclass;
set groupclass0 groupclass;
run;

proc datasets nolist;
delete t1 t2 groupclass0;
run;

proc freq data=groupclass noprint;
tables group / out=grfr;
run;

proc sort data=datanew;
by idobs;
run;

proc sort data=groupclass(keep=idobs group);
by idobs;

data datanew;
merge datanew groupclass;
by idobs;
run;

proc sort data=groupclass;
by group;
run;

%mend ppclust;
```

B.2 Macro PPCLUSTEL for HDLLSS data

PPCLUSTEL is a SAS© macro for partitioning clustering of HDLLSS data as described in Chapter 6 of this monograph.

The SAS data set for PPCLUSTEL should be in the format presented in Table B.2 where factor level is the variable with gene names, array is the corresponding replication for each factor level, and $X1, X2, \ldots, Xb$ represent the time points each factor level replication is observed.

Table B.2. *High dimensional replicated data layout in longitudinal design.*

Factor Level	Array	Time Points			
		$X1$	$X2$	$\ldots$	Xb
1	1	X_{111}	X_{121}	$\ldots$	X_{1b1}
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	n_1	X_{11n_1}	X_{12n_1}	$\ldots$	X_{1bn_1}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
a	1	X_{a11}	X_{a21}	$\ldots$	X_{ab1}
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	n_a	X_{a1n_a}	X_{a2n_a}	$\ldots$	X_{abn_a}

Command line: %macro ppclustel(dataset,gene,array,time1,timen,alpha);

Input: In this case **Dataset** is the SAS data name, **gene** is the name of the variable indicating the gene names or codes, **array** is the name of the variable indicating each gene replication number, **time1** is the variable indicating the first time point, **timen** is the variable indicating the last time point, and **alpha** is the threshold parameter to be compared with p-values in the clustering procedure, as explained in Chapter 6.

Output: PPCLUSTEL produces three temporary SAS data sets that can be saved or exported using common SAS commands. The data set called **grfr** has the number of factor levels in each created group, and the data set **groupclass** has a numerical identification for each factor level (variable idobs) and the respective group the factor level was allocated (variable group). The data set **datanew** has the original data with the information in data **groupclass** added to it.

The following command line starts PPCLUSTEL in a data set called **simulated** with gene name as **genes**, array name as **rep**, starting time as **t1** and ending time point as **t10**. The threshold is set to 0.01.

%ppclust(simulated, genes, rep, t1, t10, 0.01);

B.2.1 SAS Code

```
%macro ppclustel(dataset,geneid,arrayid,tmin,tmax,alpha);

proc freq data=&dataset noprint;
tables &geneid / out=nigene(keep=&geneid count);
run;

data nigene;
set nigene;
rename &geneid = geneid;
run;

proc means data=nigene sum noprint;
var count;
output out=sumni sum=sumni;
run;

data _null_;
set sumni;
call symput('sumni',sumni);
run;

proc iml;
start rankmiss(matrix1);
matrixm = matrix1 = .;
nmiss = matrixm[,+];
nmiss = nmiss[+,];
miss = -9999999999;
matrix1 = choose(matrix1=.,miss,matrix1);
matrix1 = ranktie(matrix1);
matrix1 = matrix1 - nmiss;
matrix1 = choose(matrix1<=0,.,matrix1);
finish;

use &dataset;
read all var{&geneid &arrayid} into factor;
read all var("&tmin":"&tmax") into expression;
b = ncol(expression);

create nb var{b};
append from b;
quit;

data datasetf1 (keep=geneid arrayid &tmin - &tmax);
```

```
set &dataset;
geneid = &geneid;
arrayid = &arrayid;
run;

data nb;
set nb;
call symput('b',b);
run;

data datasetf2 (drop= &tmin - &tmax);
set datasetf1;
array num(*) &tmin - &tmax;
do time=1 to dim(num);
exp = num[time];
output;
end;
run;

proc means data=datasetf2 median noprint;
var exp;
by geneid;
output out=result1(keep=geneid median) median=median;
run;

proc sort data=datasetf2; by geneid time;
run;

proc means data=datasetf2 mean noprint;
var exp;
by geneid time;
output out=result2(keep=geneid time rbijp) mean=rbijp;
run;

data temp1;
merge datasetf2 result2;
by geneid time;
run;

data temp2;
merge temp1 nigene;
by geneid;
msepartial = (exp - rbijp)**2 / (&b*count*(count-1));
run;
```

```
proc means data=temp2 sum noprint;
var msepartial;
by geneid;
output out=result3(keep=geneid msep) sum=msep;
run;

proc corr data=datasetf1 cov noprint outp=temp3;
var  &tmin - &tmax;
with &tmin - &tmax;
by geneid;
run;

data temp3(keep=geneid time &tmin - &tmax);
set temp3;
if _TYPE_ = 'COV';
rename _NAME_ = time;
run;

data temp4 (drop= &tmin - &tmax);
set temp3;
array num(*) &tmin - &tmax;
do timecov=1 to dim(num);
cov = num[timecov];
cov2 = cov*cov ;
output;
end;
run;

proc means data=temp4 sum noprint;
var cov2;
by geneid;
output out=result4(keep=geneid cov2p) sum=cov2p;
run;

data result4(keep=geneid cov2p);
merge result4 nigene;
by geneid;
cov2p = 2 * cov2p / (&b*count*(count-1));
run;

proc means data=datasetf1 mean noprint;
var &tmin - &tmax;
by geneid;
output out=result5(keep=geneid &tmin - &tmax) mean=&tmin - &tmax;
run;
```

```
data prepmatrix;
merge result1 result3 result4 result5;
by geneid;
run;

proc iml;

start sortmed(matrix);
call sortndx(ms,matrix,{2});
matrix = matrix[ms,];
finish;

start sortcenter(matrix,tst);
nr = nrow(matrix);
nc = ncol(matrix);
if tst = 1 then do;
m2 = matrix;
end;
else if tst > 1 then do;
ed = tst-1;
m1 = matrix[1:ed,];
m2 = matrix[tst:nr,];
end;
nr2 = nrow(m2);
j1 = j(nr2,1) * 0;
c1 = int(0.35*nr2);
c2 = int(0.65*nr2);
call sort(m2,1);
do i = 1 to nr2;
if i < c1 then j1[i,] = 1;
if i > c2 then j1[i,] = 1;
end;
m2 = m2 || j1;
nc2 = nc + 1;
call sort(m2,nc2);
m2 = m2[,1:nc];
if tst = 1 then matrix = m2;
else if tst > 1 then matrix = m1 // m2;
finish;

start anovalongrank(resp,a,b,pv);
cts = 5;
cte = 4 + b;
xijp = resp[,cts:cte];
```

```
xpjp = xijp[+,]/a;
k1 = j(a,1);
k2 = j(b,1);
k = k1 * k2‘;
xpjp = k # xpjp;
mst = (xijp - xpjp) ## 2;
mst = mst[,+];
mst = mst[+,]/((a-1)*b);
mse = resp[,3];
mse = mse[+,] / a;
cov2 = resp[,4];
cov2 = cov2[+,] / a;
pv = 1 - probnorm(sqrt(a*b)*(mst-mse)/sqrt(cov2));
finish;

start indtest(exp,b,st,alpha,a,colgr,colts,g);
exp[,colts] = 0; count = 0;
do i=1 to a;
if exp[i,colgr]=g then exp[i,colts] = 1;
end;
do i=st to a;
if exp[i,colts] = 0 then do;
kwtest = 0;
exp[i,colts] = 1;
n = sum(exp[,colts]);
if kwtest = 0 then do;
expt = exp[loc(exp[,colts]),];
call anovalongrank(expt,n,b,pv);
if (pv > alpha) then exp[i,colgr] = g;
else if (pv <= alpha) then exp[i,colts] = 0;
end;
end;
end;
do i=1 to a;
if exp[i,colgr] <= g then count = count + 1;
end;
st = count + 1;
call sort(exp,colgr);
kwtest = 0;
finish;

use prepmatrix;
read all into mdata;
alpha = &alpha;
g = 1;
```

```
ktest = 0;
a = nrow(mdata);
b = &b;
ncole = ncol(mdata);
call sortmed(mdata);
colid = 1;
colgr = ncole + 1;
colts = ncole + 2;
j1 = j(a,1);
mdata = mdata || j1 * 9999;
mdata = mdata || j1 * 0;
tstart = 1; tend = a;
call sortcenter(mdata,tstart);
call anovalongrank(mdata,a,b,pv);
pv2 = pv;
if (pv > alpha) then do;
mdata[,colgr] = 1;
end;
if (pv <= alpha) then do;
L = (tend - tstart + 1)/2;
tend = tstart + int(L-1);
do while (tstart <= a);
if (pv2 > alpha) then g = g + 1;
call sortcenter(mdata,tstart);
mdata[,colts] = 0;
do i = 1 to a;
if (tstart <= i) then do;
if ( i <= tend) then mdata[i,colts] = 1;
end;
end;
n = sum(mdata[,colts]);
if (n = 1) then do;
mdata[tstart,colgr] = 0;
tstart = tend + 1;
tend = a;
ktest = -1;
end;
exp = mdata[loc(mdata[,colts]),];
if ktest = 0 then call anovalongrank(exp,n,b,pv);
pv2 = pv;
if (pv > alpha) then do;
do i = 1 to a;
if (tstart <= i) then do;
if ( i <= tend) then mdata[i,colgr] = g;
end;
```

```
end;
tstart = tend + 1;
tend = a;
st = tstart;  exp = mdata;
call indtest(exp,b,st,alpha,a,colgr,colts,g);
tstart = st;
mdata = exp;
end;
else if (pv <= alpha) then do;
L = (tend - tstart + 1)*0.9;
tend = tstart + int(L-1);
end;
end;
end;

exp = mdata[,colid];
exp = exp || mdata[,colgr] || mdata[,2];

create groupclass var{geneid group mediange};
append from exp;
free all;
quit;

proc sort data=groupclass;
by group;
run;

data groupclass groupclass0;
set groupclass;
if group = 0 then output groupclass0;
else if group ^= 0 then output groupclass;
run;

proc means data=groupclass median noprint;
var mediange;
by group;
output out=newgroupclass median=med;
run;

proc sort data=newgroupclass;
by med;
run;

data newgroupclass(keep=group sgroup);
set newgroupclass;
```

```
if group=0 then delete;
sgroup = _N_;
run;

proc sort data=newgroupclass;
by group;
run;

data groups;
merge groupclass newgroupclass;
by group;
run;

data groups(keep=geneid group);
set groupclass0 groups;
if group=0 then sgroup=0;
rename group = og;
rename sgroup = group;
run;

proc datasets nolist;
delete datasetf1 datasetf2 groupclass groupclass0
nb newgroupclass nigene prepmatrix result1-result5
sumni temp1-temp5;
run;

data groupclass;
set groups;
rename geneid=idobs;
run;

proc datasets nolist;
delete groups;
run;

proc freq data=groupclass noprint;
tables group / out=grfr;
run;

data datanew;
set &dataset;
idobs = &geneid;
run;

proc sort data=groupclass;
```

```
by idobs;
run;

data datanew;
merge datanew groupclass;
by idobs;
run;

proc sort data=groupclass;
by group;
run;
%mend ppclustel;
```

B.3 Macro ADJRAND for Adjusted Rand Index

ARI is a SAS© macro to obtain the Adjusted Rand Index from Hubert and Arabi [46]. The measure is described in Chapter 2 and the macro is an update from Fisher and Hoffman [27] macro with corrections for the latest version of SAS© language.

The SAS data set needs just two variables indicating the results for the external criteria and the allocated group obtained with some clustering method.

Command line: %ari(name1,name2,name3);

Input: In this case **name1** is the SAS data name, **name2** is the name of the variable containing the results for the external criteria, and **name3** is the name of the variable with the grouping we want to compare with the external criteria.

Output: When ARI is executed it produces two temporary SAS data sets that can be saved or exported using common SAS commands. The data set called **adjrand** has calculated Adjusted Rand Index, and the data set **tabari** has the tabulated results for each pair of grouping in the data set **name1**.

The following command line starts macro ARI in a data set called **comparison** with compared variables named as **original** and **grouped**.

%ari(comparison, original, grouped);

B.3.1 SAS Code

```
%macro tab(data=,missing=);

proc iml;
start a2;
do i=1 to nrow(z);
t[z[i,1],z[i,2]] = z[i,3];
```

```
end;
finish;

start separate;
do k = 1 to nrow(z);
if z0[k,1] = . then if z0[k,2] = . then do;
in = i:i+z0[k,3]-1;
jn = j:j+z0[k,3]-1;
z = z // (in || jn || j(z0[k,3],1,1));
i = i + z0[k,3];
j = j + z0[k,3];
nomiss = 0;
end;
else do;
in = i:i+z0[k,3]-1;
z = z // (in || j(z0[k,3],1,z0[k,2]) || j(z0[k,3],1,1));
i = i + z0[k,3];
nomiss = 0;
end;
else if z[k,2] = . then do;
jn = j:j+z0[k,3]-1;
z = z // (j(z0[k,3],1,z0[k,1]) || jn || j(z0[k,3],1,1));
j = j + z0[k,3];
nomiss = 0;
end;
else keep = keep || k;
end;
if nomiss = 0 then z = z[keep || (nrow(z0)+1:nrow(z)),];
z0 = z;
z[rank(z[,1]),] = z0;
finish;

start combined;
x0 = loc(z[,1] = .);
y0 = loc(z[,2] = .);
if ncol(x0) > 0 then z[x0,1] = i;
if ncol(y0) > 0 then z[y0,2] = j;
if ((ncol(x0) > 0) | (ncol(y0) > 0)) then nomiss = 0;
finish;

use tabari;
read all into z0;
i = max(z0[,1])+1;
j = max(z0[,2])+1;
z = z0;
```

```
nomiss = 1;
if ((&missing = 'separate') | (&missing = '0')) then call separate;
if ((&missing = 'combined') | (&missing = '1')) then call combined;
t = j(max(z[,1]),max(z[,2]),0);
call a2;
cm = t[+,];
sscm = ssq(cm);
rm = t[,+];
ssrm = ssq(rm);
tot = rm[+];
sst = ssq(t);
nn = tot * tot;
n2 = 0.5 * tot * (tot - 1);
n0 = -0.5 * (ssrm + sscm) + n2;
ns = sst + n0;
nc = ssrm * sscm / nn + n0 + (nn - ssrm) * (nn - sscm) / (nn * (tot - 1));
adjrand = (ns - nc) / (n2 - nc);
res = nomiss || sscm || ssrm || tot || sst || adrand;
create adjrand var{adjrand};
append from adjrand;
quit;
%mend tab;

%macro freq(data=,x=,y=) / store des='ARI - FREQ'; ;
proc freq data = &data;
tables &x * &y / out = tabari(rename=(&x = x &y = y) drop=percent) noprint;
run;
%mend freq;

%macro ari(data,x,y) / store des='ARI - MAIN';
%let missing='separate';
data &data;
set &data;
z = &x + 1;
k = &y + 1;
run;

%freq(data=&data,x=z,y=k);
%tab(data=&data,missing=&missing);

data &data(drop=z k);
set &data;
run;
%mend ari;
```