RICE UNIVERSITY

Clustering Time-Course Gene-Expression Array Data

by

Jason Andrew Gershman

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE
Doctor of Philosophy

APPROVED, THESIS COMMITTEE:

Rudy Guerra, Chair
Professor of Statistics

Marek Kimmel
Professor of Statistics

Michael Gustin
Professor of Biochemistry

Gary Rosner
Adjunct Professor of Statistics

Houston, Texas
January, 2008
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.
Abstract

Clustering Time-Course Gene-Expression Array Data

by

Jason Andrew Gershman

This thesis examines methods used to cluster time-course gene expression array data. In the past decade, various model-based methods have been published and advocated for clustering this type of data in place of classic non-parametric techniques like K-means and hierarchical clustering. On simulated data, where the variance between clusters is large, I show that the model-based MCLUST outperforms model-based SSClust and non-model-based K-means clustering. I also show that the number of genes or the number of clusters has no significant effect on the performance of these model-based clustering techniques. On two real data sets, where the variance between clusters is smaller, I show that model-based SSClust outperforms both MCLUST and K-means clustering. Since the "truth" is often not known for real data sets, I use the clustered data as "truth" and then perturb the data by adding pointwise noise to cluster this noisy data. Throughout my analysis of real and simulated expression data, I use the misclassification rate and the overall success rate as measures of success of the clustering algorithm. Overall, the model-based methods appear to cluster the
Later, I examine the role of gene ontology (GO) and using gene ontology data to cluster gene expression data. I find that clustering expression data using a synthesis of gene expression and gene ontology not only provides clustering that has a biologic meaning but also clusters the data well. I also introduce an algorithm for clustering expression profiles on both gene expression and gene ontology data when some of the genes are missing the ontology data. Instead of some other methods which ignore the missing data or lump it all into a miscellaneous cluster, I use classification and inferential techniques to cluster using all of the available data and this method shows promising results. I also examine which ontology, among molecular function, biological process, and cellular component, is best in clustering expression data. This analysis shows that biological process is the preferred ontology for clustering expression data.
Acknowledgments

I would like to thank my advisor, Dr. Rudy Guerra, and my entire thesis committee including Dr. Gary Rosner, Dr. Marek Kimmel, and Dr. Michael Gustin. I would also like to thank the entire statistics department faculty as well as the students past and present. Students for whose advice, input, and moral support for whom I would like to particularly thank include Blair Christian, David Kahle, Dr. Talithia Williams, Galen Papkov, Beth Bower, Sarah Thomas, and my brother Darrin Gershman. I would also like to thank my parents for their love and support always.
Contents

Abstract ii

Acknowledgments iv

List of Illustrations x

List of Tables xviii

1 Introduction to the Problem and the Data 1

1.1 Gene Expression Microarray Data .......................... 1

1.2 Clustering Time-Course Microarray Data .................. 2

1.3 Clustering and Evaluation of Clustering Techniques ...... 4

1.4 Gene Expression Data Sets ................................ 9

1.4.1 Yeast Data Set ......................................... 9

1.4.2 Human Fibroblast Data Set .............................. 10

1.5 Chapter Summary ........................................... 11

2 Non-Parametric Clustering Methods 15

2.1 Introduction Non-Parametric Clustering .................. 15

2.2 K-means Clustering ......................................... 15

2.3 Hierarchical Clustering .................................... 21

2.4 Other Non-Parametric Clustering Methods ................ 23

2.5 Quality of Clustering Techniques .......................... 25
2.6 Shortcomings of Non-Parametric Clustering Methods .................. 27
2.7 Chapter Summary .............................................. 28

3 Parametric Clustering Methods ........................................ 29
  3.1 Splines and Spline Clustering Methods .............................. 29
    3.1.1 Spline Terminology ...................................... 29
    3.1.2 Spline Based Clustering .................................. 30
  3.2 Hidden Markov Model Clustering Methods and Bayesian Clustering
      Methods ....................................................... 34
  3.3 Smoothing Spline Clustering (SSClust) ............................. 36
  3.4 MClust .......................................................... 41
  3.5 Comparisons from the SSclust Methods Paper ....................... 44
  3.6 Chapter Summary ............................................... 49

4 Empirical Comparison of Methods: Simulated Data .................... 51
  4.1 Introduction ................................................... 51
  4.2 Motivation for Simulation Study .................................. 53
  4.3 Simulation Configuration ........................................ 56
  4.4 Checking the Consistency of K-Means on Simulated Data ............ 57
  4.5 Choosing the Methods and Evaluation Techniques .................... 61
  4.6 Comparison of Methods ......................................... 64
5 Empirical Comparison of Methods: Real Data

5.1 Clustering Real Data ........................................... 86
5.2 Clustering Yeast Data ........................................... 86
  5.2.1 Applying K-means to the Yeast Data .................... 89
  5.2.2 Applying SSClust to the Yeast Data .................... 89
  5.2.3 Applying MCLUST to the Yeast Data .................... 92
  5.2.4 Clustering Summary: Yeast Data ....................... 92
5.3 Clustering Human Fibroblast Data .......................... 94
  5.3.1 Applying K-means to the Human Fibroblast Data ........ 95
  5.3.2 Applying SSClust to the Human Fibroblast Data ........ 96
  5.3.3 Applying MCLUST to the Human Fibroblast Data ........ 99
  5.3.4 Clustering Summary: Human Fibroblast Data .......... 99
5.4 Simulations Based on Real Data .............................. 100
5.5 Simulations based on the Yeast Data ........................ 101
  5.5.1 Applying K-means to the Yeast Simulated Real Data .... 102
  5.5.2 Applying SSClust to the Yeast Simulated Real Data .... 104
5.5.3 Applying MCLUST to the Yeast Simulated Real Data .......... 104
5.6 Simulations Based on the Human Fibroblast Data ............... 107
  5.6.1 Applying K-means to the HF Simulated Real Data ........... 109
  5.6.2 Applying SSClust to the HF Simulated Real Data .......... 111
  5.6.3 Applying MCLUST to the HF Simulated Real Data .......... 111
5.7 Examining the Hartigan Rule via Bootstrap Analysis ............. 113
5.8 Chapter Summary ............................................. 120

6 Gene Ontology: Methods and Results ................................ 122

  6.1 Introduction to Gene Ontology ................................. 122
  6.2 Gene Ontology Based Clustering ................................ 125
  6.3 Clustering using GO Data verses Expression Data in the HF Data Set 128
      6.3.1 Clustering Using Expression Data for the HF Data Subset ... 129
      6.3.2 Clustering Using GO Data for the HF Data Subset .......... 134
  6.4 Clustering using GO Data verses Expression Data in the Yeast Data Set 138
      6.4.1 Clustering Using Expression Data for the Yeast Data Subset .. 138
      6.4.2 Clustering Using GO Data for the Yeast Data Subset .......... 141
  6.5 Clustering using both GO and Expression Data ................... 147
      6.5.1 Application to the Human Fibroblast Data Set ............... 154
      6.5.2 Application to the Yeast Data Set ........................ 155
  6.6 Clustering on Expression and GO Data with Missing Values ....... 162
7 Conclusions, Recommendations, and Discussion

7.1 Review of Methods and Conclusions ........................................ 185
7.2 Recommendations .............................................................. 189
  7.2.1 Extensions ................................................................. 197
7.3 Status of this Field ............................................................ 197
  7.3.1 Additional Extensions and Concluding Remarks .................. 199
Illustrations

1.1 K-Means Clustering for the yeast data set with 5 clusters chosen a priori ........................................ 12
1.2 Hierarchical Clustering for the yeast data set with 5 clusters chosen a priori ........................................ 13

2.1 Yeast with seven a priori clusters: Sum of the Within Cluster Sums of Squares vs Run Order .................... 19

3.1 Functional Form of the Four Curves in the Simulation from the SSClust Methods Paper .................................... 46
3.2 Clustering results as outlined in the Ma et al. (2006) Paper ................................................................. 47

4.1 Functional Form of the Four Curves in the Simulation from the SSClust Methods Paper .................................... 52
4.2 Functional Form of the Five Curves for Simulation ................................................................. 54
4.3 Incorrect Clustering by SSClust; The sine curves were divided into two clusters, and the linear and flat curves combined into one cluster. ................................................................. 55
4.4 Simulated Data Sum of WCSS using: (a) one starting chain; (b) five starting chains; The true number of clusters is four. ................................................................. 59
4.5 Sum of the WCSS using: (a) One starting chain; (b) Five starting chains; The true number of clusters is four but clusters are of unequal size. ................................................................. 60
4.6 Underlying functions for the Seven Curves for Simulation Set 2

4.7 Misclassification rates for the three clustering methods for seven true clusters verses four true clusters; Points on this graph correspond to different distance metrics (when applicable), number of time points, curve distribution, and noise

4.8 Rates of finding the correct number of clusters in clustering runs for the three clustering methods for seven true clusters verses four true clusters; Points on this graph correspond to different distance metrics (when applicable), number of time points, curve distribution, and noise

4.9 Overall Success Rate vs Number of Time Course Profiles for two simulated data sets with 4 true clusters and 7 true clusters respectively

5.1 Hierarchical Clustering for the yeast data with 5 clusters chosen a priori

5.2 Plot of Hartigan Rule value vs number of clusters for yeast data; the rule crosses y=10 at x=19

5.3 Plots of the 19 clusters of the yeast data as identified by K-means and Hartigan Rule

5.4 Plots of the 10 clusters for the yeast data in SSCLust

5.5 Plots of the nine clusters in MCLUST for Yeast Data

5.6 Hierarchical Clustering for the Human Fibroblast data with 10 clusters chosen a priori
5.7 Plot of Hartigan Rule Value vs Number of Clusters for Human Fibroblast Data ........................................ 96

5.8 Plots of the 20 clusters of the HF Data by K-means under the Hartigan Rule for choosing the number of clusters .................................................. 97

5.9 Plots of the four clusters found in HF Data by SSClust ................................................................. 98

5.10 Plots of the three clusters in MCLUST for HF Data ........................................................................ 99

5.11 Number of Clusters via Hartigan Rule for K-means Analysis of Yeast Data; The original unperturbed yeast data set was partitioned into 19 clusters by the Hartigan Rule ......................................................... 103

5.12 Number of Clusters via BIC rules for SSClust Analysis of Yeast Data; The original unperturbed yeast data set was partitioned into 10 clusters by SSClust .................................................. 105

5.13 Number of Clusters via BIC rules for MCLUST Analysis of Yeast Data; The original unperturbed yeast data set was partitioned into nine clusters by MCLUST ...................................................... 106

5.14 Number of Clusters via Hartigan Rule for K-Means analysis of Human Fibroblast Data; The original unperturbed yeast data set was partitioned into 20 clusters by the Hartigan Rule .................................................. 109

5.15 Number of Clusters via BIC rules for SSClust Analysis of HF Data; The original unperturbed data set was partitioned into 4 clusters by SSClust ......................................................... 110
5.16 Number of Clusters via BIC rules for MCLUST Analysis of HF Data;

The original unperturbed data set was partitioned into 3 clusters by

MCLUST. .................................................. 112

5.17 Number of clusters found via the Hartigan Rule in K-means for

bootstrap (100 iterations per sample size) samples from the yeast
data set at sample sizes of four different percentages of the overall
sample size ($n=433$) ........................................... 114

5.18 Number of clusters found via the Hartigan Rule in K-means for

bootstrap (100 iterations per sample size) samples from the simulated
data set (with 4 true clusters simulated from 4 true mean curves) at
sample sizes of four different percentages of the overall sample size

($n=400$) .................................................... 116

5.19 Number of clusters found via the Hartigan Rule in K-means for

bootstrap (100 iterations per sample size) samples from the simulated
data set (with 7 true clusters simulated from 7 true mean curves) at
sample sizes of four different percentages of the overall sample size

($n=175$) .................................................... 118

6.1 Two directed graphs for two genes from the Human Fibroblast data

set using GO Molecular Function annotation .......................... 123
6.2  Top- Mean Curves for 3 Clusters found by MCLUST for the Human Fibroblast Data; Bottom: Raw Curves corresponding to the 3 Clusters in the top panel .......................................................... 128

6.3  Hierarchical Clustering Assignment for Expression HF Data; n=211 genes, genes for which GO molecular function annotations are available 130

6.4  MCLUST Clusters for Expression HF Data; n=211 genes, genes for which GO molecular function annotations are available .............. 131

6.5  K-means Clusters for Expression HF Data; n=211 genes, genes for which GO molecular function annotations are available .............. 132

6.6  Density plot of the union-intersection distances for n=211 genes in the HF data set using molecular function annotations ....................... 133

6.7  K-means Clusters of HF gene expression profiles clustered by Gene Ontology molecular function of n=211 genes ......................... 134

6.8  Mean Curves of HF gene expression profiles clustered by Gene Ontology molecular function of n=211 genes ......................... 135

6.9  Density plots of the union-intersection distances within cluster for the six clusters of HF data using K-means clustering ....................... 136

6.10 K-means Clusters for Expression Yeast Data; n=398 genes, genes for which GO biological processes annotations are available) ........ 139

6.11 MCLUST Clusters for Expression Yeast Data; n=398 genes, genes for which GO biological processes annotations are available ........ 140
6.12 SSClust Clusters for Expression Yeast Data; n=398 genes, genes for which GO biological processes annotations are available . . . . . . . . . 141

6.13 Density plot of the union-intersection distances for n=398 genes in the yeast data set using biological processes annotations . . . . . . . . 142

6.14 K-means Clusters of yeast gene expression profiles clustered by Gene Ontology biological processes of n=398 genes . . . . . . . . . . . . 143

6.15 Mean Curves of yeast gene expression profiles clustered by Gene Ontology biological processes of n=398 genes . . . . . . . . . . . . 144

6.16 Density plots of the union-intersection distances within cluster for the nine clusters of yeast data using K-means clustering . . . . . . . . . . 145

6.17 K-means Clusters for the human fibroblast data set clustered by both expression data and gene ontology; n=211 . . . . . . . . . . . . 152

6.18 Mean Curves for the K-Means clusters for the human fibroblast data set clustered by both expression data and gene ontology; n=211 . . 153

6.19 K-means Clusters for yeast expression data clustered by both expression data and gene ontology; n=398 . . . . . . . . . . . . 156

6.20 Mean Curves for the K-Means clusters for yeast expression data clustered by both expression data and gene ontology; n=398 . . . . 157
6.21 13 Clusters of Yeast Data from knowledge guided analysis; mrs
threshold value = 0.35; Each cluster is labeled in the title by its
common Biological Process along with the cluster number, GO tree
level, and mrs value. ................................. 159

6.22 Mean Curves for the 13 clusters of yeast data from knowledge guided
analysis; mrs threshold value = 0.35; Each cluster features is labeled
in the title by its common Biological Process along with the cluster
number, GO tree level, and mrs value. ............................... 160

6.23 K-means Clusters for HF Data: Using all available data (expression
and GO) on the full expression data set (517 genes); Using
interpolation for GO data when that data information is missing ... 165

6.24 Directed Acyclic Graph (DAG) for informative GO nodes of one of
the nine clusters found in the K-means analysis of synthesis of GO
and expression data ................................. 170

6.25 K-means Clusters of yeast gene expression profiles clustered by Gene
Ontology biological processes of n=398 genes .......................... 173

6.26 Mean Curves of yeast gene expression profiles clustered by Gene
Ontology biological processes of n=398 genes .......................... 174

6.27 K-means Clusters of yeast gene expression profiles clustered by Gene
Ontology molecular function of n=398 genes .......................... 176
6.28 Mean Curves of yeast gene expression profiles clustered by Gene

Ontology molecular function of n=398 genes ......................... 177

6.29 K-means Clusters of yeast gene expression profiles clustered by Gene

Ontology cellular component of n=398 genes ......................... 178

6.30 Mean Curves of yeast gene expression profiles clustered by Gene

Ontology cellular component of n=398 genes ......................... 179
Tables

3.1 The various MCLUST models with different covariance structures 43

3.2 Clustering performance over 100 simulation iterations based on the simulated data from and reported in Ma et al. (2006) 48

3.3 Clustering methods I will use in my analysis of time-course data and simulations throughout the remainder of this thesis 49

4.1 Clustering performance over 100 simulation iterations based on the simulated data from and reported in Ma et al. (2006) 51

4.2 Misclassification Rates (MR%) for K-means (KM), MCLUST (MC), and SSClust (SSC) clustering; 4 true clusters; 100 simulated curves total. The misclassification rate is an average over 100 simulation iterations. 65

4.3 Percentage of runs that the clustering technique found the correct number of clusters; 4 true clusters; 100 simulated curves total. The simulation configuration is as in Table 4.1. 66

4.4 Misclassification Rates for K-means, MCLUST, and SSClust clustering methods; seven true clusters, 175 simulated curves total from these 7 true clusters; The misclassification rate is the average over 100 simulation iterations. 72
4.5 Percentage of runs that the clustering technique found the correct number of clusters; seven true clusters; with 175 simulated curves from these 7 true clusters; The simulation configuration is as in Table 4.3. .......................................................... 73

4.6 Misclassification Rates for K-means for 4 true clusters at 4 different total numbers of curves .................................................. 79

4.7 Misclassification Rates for K-means for 7 true clusters at 4 different total number of curves .................................................. 79

4.8 Misclassification Rates for MCLUST for 4 true clusters at 4 different total number of curves .................................................. 80

4.9 Misclassification Rates for MCLUST for 7 true clusters at 4 different total number of curves .................................................. 80

4.10 Misclassification Rates for SSClust for 4 true clusters at 4 different total number of curves .................................................. 81

4.11 Misclassification Rates for SSClust for 7 true clusters at 4 different total number of curves .................................................. 81

5.1 Number of Clusters found via various clustering methods for the yeast data ................................................................. 92

5.2 Number of Clusters found via various clustering methods for the human fibroblast data .................................................. 100
5.3 Clustering performance over 100 simulation iterations based on the Yeast Data ................................................................. 102

5.4 Clustering performance over 100 simulation iterations based on the Human Fibroblast Data ..................................................... 108

6.1 Molecular Function classification for the most informative GO tags obtained from genes within each of the nine clusters obtained by clustering by K-Means by the synthesis of expression and the GO data 171

6.2 Summary statistics for the sum of the within cluster sums of squares for K-means analysis of clustering yeast expression data by a GO distance matrix for the three choices of the GO ontologies; n=398 clustered in each analysis; 100 runs of the clustering algorithm for each ontology ................................................................. 180

7.1 The number of clusters, the sum of squared distances of each element to its cluster centroid, and the penalized sum of squared distances multiplied by the square root of the number of clusters, n; Yeast Data 189

7.2 Biological Process classification for the most informative GO tags obtained from genes from the yeast data set; n=398 genes ........ 191
7.3 Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in K-means where clustering is performed by expression data alone; 18 clusters found for $n=398$ genes .......................................................... 193

7.4 Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in K-means where clustering is performed by a synthesis of added scaled distance matrices for expression data and GO data; 12 clusters found for $n=398$ genes .......................................................... 194

7.5 Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in MCLUST, where genes are clustering using expression data only; 9 clusters found for $n=398$ genes .......................................................... 196

7.6 Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in SSCLust, where genes are clustering using expression data only; 10 clusters found for $n=398$ genes .......................................................... 196
Chapter 1

Introduction to the Problem and the Data

1.1 Gene Expression Microarray Data

The type of data analyzed in this thesis comes from DNA microarrays. DNA microarrays, also known as gene chips, are collections of microscopic DNA spots, often representing single genes, attached to a solid surface. The DNA spots, known as probes, are arranged in known locations throughout the microarray. Microarrays are popular because they contain a large number of genes in a controlled order in a small space making them useful to survey a large number of genes simultaneously. Microarrays can be used to assess gene expression in a single sample or compare gene expression between two different cell types or two different tissue samples, such as in healthy and diseased tissue.

The term gene expression is the used to describe the transcription of information from DNA to messenger RNA and then into proteins which perform the critical functions of cells. Cells in an organism contain a full set of chromosomes and genes but only a small fraction of these genes are turned on, or expressed, and this expression level differs for different cell types. Gene expression is a complex and highly regulated process which responds to environmental stimuli as well as the changing needs of the cell. These changes act as an “on/off” switch, to express or not express genes in the
cell at different times, and as a “volume control” that increases or decreases the level of expression of genes as necessary. These behaviors are further explained by the National Center for Biotechnology Information (2007).

The changing nature of the expression values measured against time leads to this data being referred to as time-course data. The expression values are often expressed as the ratio, on a log base two scale, of the expression under the stimulation in the experiment verses the expression under normal conditions, or the alternative experiment. Scientists hope to use gene-expression microarray data in order to cluster similarly expressed genes together. Proper clustering of similarly behaving genes might be the keys to the early detection or curing of diseases with strong genetic links such as cancer and heart disease. The hope is that a cluster of a collection of over-expressed or under-expressed genes might have some insight into genetic markers that indicate the development of a disease where early intervention is crucial to the recovery of the patient.

1.2 Clustering Time-Course Microarray Data

This thesis examines methods used to cluster time-course gene expression array data. Traditional statistical methods exist which can cluster samples of curves with each curve representing one gene on the gene chip at different times. The curve data can be represented by an \( n \times p \) matrix where each row is one curve representing a single gene and each column is a time point where the expression value is measured. This type of
data is obtained by a carefully controlled microarray experiment. One common use of DNA microarrays is to determine which genes are activated and which genes are suppressed when two populations of cells are compared and, in this comparison, every gene will be analyzed simultaneously. One goal of these experiments is to cluster data by organizing like curves together in order to make conclusions about the role of the genes in the process or the organism being analyzed.

Two common clustering algorithms used to cluster time-course array data are K-means and hierarchical clustering. These methods are non-parametric. In the past decade, various model-based methods have been published and advocated to cluster this type of data in place of non-parametric methods. In the first three chapters, I introduce the arguments in this clustering debate including whether traditional non-parametric methods are sufficient to cluster this type of data. The alternative to these methods are newer, more computationally intensive, model-based methods like MCLUST and SSClust. I also describe and implement these clustering techniques.

In chapter four, I compare the performance of clustering techniques, the non-parametric K-means algorithm and the model-based MCLUST and SSClust methods. These comparisons are done on simulated data designed to reflect shape and amplitude common patterns of gene expression data. In chapter five, I examine the performance of these same clustering techniques on two familiar time-course array data sets (introduced later in this chapter.) This is done by simulating data sets based on real data sets using the real data as a notion of truth. In chapter six, I
introduce gene ontology data and the debate over how to include biological information in clusters expression data. I then compare clustering by ontology to that using expression data and use various methods to combine the data to have biologically sound expression clustering.

The rest of this chapter introduces the evaluation techniques used on clustering methods for gene expression curves and also introduces two data sets I use throughout this thesis.

1.3 Clustering and Evaluation of Clustering Techniques

Attempts to cluster and organize data are nothing new. Recent debates about how to display and organize similar curves were sparked by Jones and Rice (1992). They wrote:

"Naively displaying a large collection of curves by superimposing them one on another all on the same graph is largely uninformative and aesthetically unappealing" (140).

In Jones and Rice (1992), they advocated obtaining a small number of figures which plot a few representative curves from the original collection whereby clearly conveying the major information present from the original data set.

Questions over objective criteria for evaluating clustering methods date back to Rand (1971). Modern techniques of hierarchical clustering and nearest neigh-
bor techniques use evaluation criteria based on this early paper on clustering. Rand (1971) advocated that there are two ways to compare clustering methods; in ease of use and in performance, or how well are like elements joined in the same cluster. Ease of use includes taking into account ease of programming and computation time but Rand (1971) emphasized that the performance of the clustering technique takes precedence over ease of use.

While the power of computing has increased exponentially in the 36 years since the publication of Rand (1971), speed and ease of computation is still an issue to this day and is taken into account in comparisons of new model-based and traditional non-parametric methods. Rand (1971) introduced issues about weighing certain points in the data set or having each point weighted equally. This paper also introduced issues about evaluation when the truth underlying the clustering of a data set is not known, as is the case with real data sets. While the truth is known for simulated data sets and misclassification can be easily counted in these cases, the situation of what to do when truth is unknown was first explored by Rand (1971).

A measure of agreement between the clustering used in the model-based clustering MCLUST software for evaluation purposes is based on a method from Rand (1971) whereby each clustering method has two components:

Criterion assigns each cluster a numerical value which indicates its relative desirability in the context of the given method
Technique selects a particular subset of the set of all partitions of the data which optimize the given criterion.

Based on the evaluation techniques of Rand (1971), statisticians can compare results of a new method to standard results or to another method. In an ideal world, Rand (1971) stated there is a “correct clustering against which we can measure new techniques. These measures of goodness can be as simple as misclassification rates and get increasingly more complicated from there” (847). Three motivating factors in the evaluation of clustering, described in Rand (1971), are (i) every point is assigned to exactly one cluster; (ii) clusters are defined, not just by the points they contain, but also by the points they do not contain; (iii) all points are equally important (equal weights.)

According to Rand (1971), four questions that need to be answered about any clustering method are:

- How well does a method retrieve “natural” clusters?
- How sensitive is a method to perturbations of the data?
- How sensitive is a method to missing individuals?
- Given two methods, do they produce different results when applied to the same data?
The authors of MCLUST in Yeung, Fraley, Murua, Raftery, and Ruzzo (2001) used the index developed by Rand (1971) for the measure of degree of agreement between two clustering methods, which Yeung et al. (2001) called the (Modified) Rand Index. The (Modified) Rand Index is the number of pairs of curves that are either in the same cluster in both clustering partitions or are in different clusters in both partitions, divided by the total numbers of pairs of curves.

A Rand Index of 1 indicates identical clustering between two clustering methods. The Rand Index ranges from 0 through 1 where a Rand Index of 0 indicates a case where one technique produces 1 cluster containing all n curves while the other technique produces n clusters of 1 curve each. Raftery, Yeung, and Fraley (2001) elaborated on of this comparative technique. This method, while it does not give right and wrong clustering as an outcome, does provide a starting point for comparing two clustering results for the same data set.


"success in applications has been reported for many clustering approaches,

but so far no single method has emerged as the method of choice in the
gene expression analysis community. Most of the proposed clustering algorithms are largely heuristically motivated, and the issues of determining the ‘correct’ number of clusters and choosing a ‘good’ clustering algorithm are not yet rigorously solved” (977).

They pointed to the particular shortcoming in the non-parametric methods in that “Eisen et al. (1998) … used visual display to determine the number of clusters” (977).

They believed that clustering algorithms based on probability models offer a principled alternative to heuristic-based algorithms. With the underlying probability model, Yeung et al. (2001) stated that

“the problems of determining the number of clusters and of choosing an appropriate cluster method become statistical model choice problems. This provides a great advantage over heuristic algorithms, for which there is no established method to determine the number of clusters or the best clustering method” (978).

Some comparisons exist between model-based and non-model based methods but most of these are in the context of the development of whether a new clustering method is at least as effective or more effective than other existing methods for particular data sets. While they are more mathematically advanced, one question often brought up for debate is the effectiveness of model-based clustering for time-course
gene expression array data. Are there cases where the non-model based techniques are sufficient? This thesis will compare the clustering methods based on their performance in a variety of situations and under varying operating characteristics. The performance will be measured on data sets that differ in:

**Resolution:** Number of Time Points Measured

**Sample Size:** Number of curves being clustered

**Relative Size of Clusters:** The distribution of curves between each cluster

**Geometry:** The shapes of the curves

**Noise:** Levels of measurement error

### 1.4 Gene Expression Data Sets

#### 1.4.1 Yeast Data Set

Spellman et al. (1998) measured genome-wide mRNA levels on certain yeast cells (*Saccharomyces cerevisiae*.) They measured mRNA on yeast open reading frames (ORFs) simultaneously over approximately two cell-cycle periods in a yeast culture synchronized by a factor relative to reference mRNA from an asynchronous yeast culture. Log ratios of gene expression were taken every seven minutes for 119 minutes. For each of the 18 time points, a total of 6300 ORFs were measured, of which nearly 800 genes were categorized as cell-cycle regulated. Among these genes, a subset of 433
genes containing no missing values is a popular gene expression data set. This data set has been used in numerous methods papers to introduce and evaluate new clustering methods including the Functional Clustering Model clustering method based on the works of James and Sugar (2003) and Luan and Li (2003a). In the initial analysis on this data set from Spellman et al. (1998) and Eisen, Spellman, Brown, and Botstein (1998), hierarchical clustering and $K$-means analysis were used to cluster the yeast data and these results are discussed later in this chapter and in Chapter 5 when I cluster the real data sets. Throughout the duration of this thesis, the terms "yeast data set" and "yeast cell cycle data set" shall refer to this commonly used data set of 433 genes and 18 time points.

1.4.2 Human Fibroblast Data Set

Iyer, Eisen, Ross, Schuler, Moore, Lee, Trent, Staudt, Hudson Jr, and Boguski (1999) measured the gene expression levels during the physiological response of human fibroblasts to serum using cDNA microarrays for over 8000 genes over a 24-hour period. This study was designed to test the growth factors of human fibroblasts in the presence and absence of fetal bovine serum, which normally provides the growth factors for proliferation of these cells in culture. They identified 517 genes whose expression levels changed in response to serum stimulation. A key feature of this data set is that the 12 time points were not equally spaced during the 24-hour period. Sampling times are at 0, 0.25, 0.5, 1, 2, 4, 6, 8, 12, 16, 20, and 24 hours after
serum stimulation. This data set has been used by various authors (e.g., Luan and Li (2003a)) to show the ability of parametric models to handle unevenly spaced time points in their clustering algorithms. Iyer et al. (1999) used hierarchical clustering to cluster this data with biological aspects of the data used to choose the number of clusters. These results are discussed and shown in Chapter 6 when I introduce the use of gene ontology.

In addition to having unequal time spacing and fewer time points than the yeast data set, the variance of this data set is less than half the variance for the entire yeast data set, which adds a slighter degree of difficulty in clustering for some of the clustering algorithms. For the duration of this thesis, the terms “human fibroblast data set” and “hf data set” refer to this data set of 517 genes and 12 time points.

1.5 Chapter Summary

This chapter introduced data sets and evaluation techniques for clustering gene expression microarray data sets. To illustrate the differences between clustering results between different clustering methods, Figures 1.1 and 1.2 show the yeast data set clustered by K-means and hierarchical clustering (average linkage method) respectively, where 5 clusters was chosen a priori for each clustering method (five clusters were initially chosen for hierarchical clustering analysis of this data set by Eisen et al. (1998) because of the five cell-cycle phases.) Looking at these graphs, the shapes of the clusters and the number of genes per cluster are quite different between the two
Figure 1.1: K-Means Clustering for the yeast data set with 5 clusters chosen a priori
Figure 1.2: Hierarchical Clustering for the yeast data set with 5 clusters chosen a priori
methods. Yet, both of these methods are used on this type of data with no general rule as to which method is better and which clustering is “correct”. The addition of numerous other clustering techniques developed over the years increases the difficulty in determining a best method.

The questions over which clustering method is the best clustering method in the case of time-course expression data have existed for many years with little resolution. Recent advancements in the number of available data sets for time-course gene expression data, as well as new statistical techniques for analyzing these data sets, make this a timely and important area to explore. These new methods not only take the time ordering into account but some, like Pan (2006), also took biologic processes into account. These new methods have become more mathematically intrinsic and computationally expensive.

I will explore the traditional techniques and the new model-based techniques, including those which involve Gene Ontology, in the chapters to come. The microarray data sets introduced in this chapter will help provide some insight into how the algorithms clusters and which clustering methods are better than others for this type of data. The specific differences in the methods, and their mathematical foundations, are described in the next two chapters.
Chapter 2
Non-Parametric Clustering Methods

2.1 Introduction Non-Parametric Clustering

This chapter elaborates on two popular non-parametric clustering methods for time-
course gene expression array data, $K$-means clustering and hierarchical clustering.

Both of these methods were developed before the introduction of microarrays but
have been adapted for use on microarray data.

In non-parametric clustering of data curves, the starting point is defining and
applying a measure of dissimilarity between each pair of curves and creating an $n \times n$
distance matrix. The clustering is then based on the distance matrix.

2.2 K-means Clustering

\begin{algorithm}
\caption{$K$-means Algorithm}
Initially, data are assigned at random to $k$ clusters, where $k$ is fixed, chosen \textit{a priori}.
\begin{algorithmic}
\State \textbf{repeat}
\State The centroid is computed for each cluster.
\State Data are reassigned to the cluster whose centroid it is nearest to in Euclidean
\State \hspace{1em} Distance.
\State \textbf{until} The algorithm terminates when all cluster centers are equal to the means of
\State \hspace{1em} their Voronoi sets (the set of data points which are nearest to the cluster center).
\end{algorithmic}
\end{algorithm}

K-means is a relocation method as the number of clusters is pre-determined and
curves can change cluster assignment at each iteration. A commonly used $K$-means
algorithm is based on the algorithm developed in Hartigan and Wong (1979). This is the algorithm implemented in the R software package developed by the R Project for Statistical Computing (2007). In the context of clustering data curves, the $K$-means algorithm is described in Algorithm 1.

In a more mathematical sense, $K$-means finds a clustering of the observations to minimize the total within-cluster sums of squares, defined as:

$$ E = \sum_{c=1}^{K} Error_c $$  \hspace{1cm} (2.1)

where $K$ is the total number of clusters, $Error_c$ is the sum of the squares for cluster $c$ and defined by $Error_c = \sum_{i=1}^{n_c} d_i^2$ and where $n_c$ is the number of data curves in cluster $c$ and $d_i^2$ is the Euclidean distance between data curve $i$ and its designated cluster’s center.

The centroid for for cluster $c$ is defined as

$$ \bar{x}_c = \frac{\sum_{i=1}^{n_c} y_i}{n_c} $$  \hspace{1cm} (2.2)

where $y_i$ is an individual data curve in cluster $c$.

To implement the $K$-means algorithm in the R software packages, I started with the initial data matrix. I then created a square dissimilarity matrix for the distances between each pair of curves. For $n$ rows (genes) in the data set, this square ($nxn$) dissimilarity matrix consisted of Euclidean distances or the Pearson Correlation (used
as a distance) between each pair of curves.

For two vectors of time course data of equal length denoted \((x_1, x_2, ..., x_n)\) and \((y_1, y_2, ..., y_n)\), the Euclidean distance between the curves is defined as:

\[
d_{\text{Eucl}}(\mathbf{x}, \mathbf{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2}
\]  

(2.3)

For two vectors of time course data of equal length denoted \((x_1, x_2, ..., x_n)\) and \((y_1, y_2, ..., y_n)\), the Pearson Correlation distance between the curves is defined as:

\[
d_{\text{Pear}}(\mathbf{x}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^{n} \frac{(x_i - \mu_x)(y_i - \mu_y)}{\sigma_x \sigma_y}
\]  

(2.4)

In addition to the dissimilarity matrix, the algorithm requires either the number of clusters or guesses for the initial cluster centers for the \(k\) clusters. If the initial cluster centers are given, and no element of the data set is closest to one of the initial cluster centers (no curves are assigned to that cluster on the first run through the algorithm,) the algorithm will terminate prematurely. Hence, this choice requires some \textit{a priori} knowledge of the expected means of the clusters in the data. If, instead, the number of clusters is pre-specified, \(k\) elements of the data set are chosen as the cluster centers and the algorithm runs until proper termination.

For these time-course data sets, the performance and hence the final clustering could be influenced by this initial choice of cluster centers. One of the input options for the R implementation of this algorithm is in the number of initial guesses for
the cluster centers. This number is denoted \( r \) in the software. If \( r > 1 \), then the algorithm will run \( r \) times (one choice for the initial cluster centers are elements \( 1 : k \) in the data set; another set of initial cluster centers are elements \( (k + 1) : (2k) \); the final set of initial cluster centers are elements \( (r - 1)(k + 1) : rk \) yielding \( r \) runs total.) The R implementation of K-means will output the run with the smallest sum of within-cluster sum of squares (sums of squared values of each element to the center of the cluster it is assigned to.) Another option is for random initial cluster centers chosen from among the elements in the data set and this is the method that is most commonly used to avoid traps in local minima in the K-means algorithm.

Figure 2.1 shows the sum of the within-cluster sums of squares (WCSS) verses number of starting chains for the clusters centers for 100 runs of the K-means algorithm for the yeast data set for each of two different values of \( r \). This number is found by finding the sum of the squared distances between all pairs of curves in a particular cluster and then adding these results for each cluster. On the left, \( r \) is equal to 1 while on the right \( r \) is equal to 5. The mean of the sum of the WCSS when running 1 starting chain is 59848 while the mean drops to 58531 under 5 starting chains (random initial cluster assignments.) The standard deviation of the sum of the WCSS when running 1 chain is 984 and is reduced to 904 for 5 starting chains. The p-value for a two-sample Wilcoxon rank-sum test for a significant decrease in the within cluster sums of squares from 1 starting chain to 5 starting chains is \(< 0.01 \) and thus the sum of the within cluster sum of squares is significantly smaller with
Figure 2.1: Yeast with seven a priori clusters: Sum of the Within Cluster Sums of Squares vs Run Order
five starting chains.

More initial starting chains for the cluster centers yields more consistent clustering runs and resistance to poor clustering (stuck in a local minimum). This multichain K-means algorithm is not always discussed in papers that attack the K-means method, which tend to run one starting chain.

Two of the most repeated shortcomings of the K-means method are that the number of clusters has to be chosen a priori and that the search is prone to termination at local minima of the Voronoi sets. This second limitation can be avoided by running more initial starting chains for cluster centers but Pelleg and Moore (2000) focused their criticism on the other “major shortcoming” which is that the number of clusters has to be chosen a priori. In their paper, Pelleg and Moore (2000) developed a method and software they call X-means. X-means estimates the number of clusters needed via an optimization method such as the Bayesian Information Criterion, a criterion often used in parametric clustering. But, this software has been cited as being inefficient by Ramoni, Sebastiani, and Cohen (2002a) in its performance and has not been applied in comparisons to model-based method.

While there is no steadfast rule on determining a priori the number of clusters in K-means, a commonly used rule-of-thumb for deciding on the number of clusters is suggested by Hartigan (1975). This method is denoted as the “Hartigan Rule” or the “Hartigan Rule of Thumb” and is advocated by Dudoit and Fridlyand (2002) in their study of methods of clustering time-course array data. The Hartigan Rule states
that for a data set of \( n \) curves (\( n \) genes), let \( k_1 \) denotes the clustering by \( K \)-means with \( k \) groups and \( k_2 \) denotes the clustering by \( K \)-means with \( k + 1 \) groups. If \( E1 \) and \( E2 \) are the sums of the within cluster sums of squares, as defined previously in this chapter in Equation 2.1, for the clustering from \( k_1 \) and \( k_2 \) respectively, then add the extra group if:

\[
\frac{E1(n-k-1)}{E2} > 10
\]  

(2.5)

where the choice of 10 for the cutoff value appears to be arbitrary in this application.

For my work in this thesis, I implemented the \( K \)-means algorithm in R using the `kmeans` command.

I will use the \( K \)-means algorithm to cluster time-course data in later chapters of this thesis.

### 2.3 Hierarchical Clustering

**Algorithm 2 Hierarchical Clustering**

Initially, each curve is assigned to its own cluster.

repeat

The two closest clusters are joined into one branch to create a clustering tree.

(Without stoppage the algorithm would iterate until there was one large cluster.)

until The clustering tree stops joining branches when the algorithm terminates via a stopping rule. These stopping rules are explained in the text.

While \( K \)-means is a relocation method, hierarchical clustering, described in Algorithm 2, is an addition or subtraction method. Hierarchical clustering was first
developed by Johnson (1967). As with K-means, a distance matrix between curves is needed. Clusters are combined to optimize a pre-specified criterion. Three common choices for the criterion are:

*Single Linkage (Nearest Neighbor):* distance between clusters \( r \) and \( s \) is the minimum distance among all pairs of curves, one from each cluster,

\[
D(r, s) = \text{Min}\{d(i, j) : i \in r, j \in s\}
\]

*Complete Linkage (Furthest Neighbor):* distance between clusters \( r \) and \( s \) is the maximum distance among all pairs of elements, one from each cluster,

\[
D(r, s) = \text{Max}\{d(i, j) : i \in r, j \in s\}
\]

*Average Linkage:* distance between clusters \( r \) and \( s \) is the average distance between all pairs of elements between the two clusters:

\[
D(r, s) = \frac{T_{rs}}{N_r N_s}
\]

where \( T_{rs} \) is the sum of all pairwise distances between curves in cluster \( r \) and cluster \( s \) and \( N_r \) and \( N_s \) are the sizes of clusters \( r \) and \( s \) respectively.

The algorithm stops when no new joining of clusters is an improvement over the current clustering based on the particular optimization criteria. Normally, the algorithm stops at a pre-determined number of clusters although it can stop when the distance between the two closest clusters reaches some threshold. But, no universal rule-of-thumb is used to evaluate hierarchical clustering of the same data set while
varying the numbers of clusters found via the algorithm.

These non-parametric clustering methods of $K$-means and hierarchical clustering were implemented for microarray data by Eisen et al. (1998) and in software from Eisen's laboratory at Stanford University. Eisen's laboratory has developed the "tree-view" and "cluster" software packages to implement the $K$-means and hierarchical clustering methods and display the clustering results.

Papers performing hierarchical clustering on gene expression array data generally use one of the three methods described above. In their work, Gibbons and Roth (2002) show that for gene expression array data, complete linkage is the best among these three. For my work in this thesis, I implement the hierarchical clustering algorithm in R using the 'hclust' command.

2.4 Other Non-Parametric Clustering Methods

Using Principal Component Analysis (PCA) was a popular method for analyzing and clustering gene expression data when that data first became available. In a comparison of clustering methods in Yeung and Ruzzo (2001), the authors considered principal component analysis for clustering gene expression data. But Yeung and Ruzzo (2001) concluded that their "empirical study showed that clustering with the PCs instead of the original variables does not necessarily improve, and often degrades, cluster quality. In particular, the first few PCs (which contain most of the variation in the data) do not necessarily capture most of the cluster structure .... Overall,
we would not recommend PCA before clustering" (763). PCA has not been used in methodological papers to compare clustering methods in recent years.

Partitioning around medoids (PAM) developed by Kaufman and Rousseeuw (1990) is another non-parametric method implemented in the "cluster" software package. PAM is similar to K-means but uses medoids instead of centroids. A medoid is defined as having minimal average dissimilarity to all elements in the cluster. This method allows for other measures of dissimilarity than the distance metrics used in K-means. Another non-parametric technique is based on self organizing maps (SOM) introduced by Kohonen (1997). This clustering method attempts to reduce the dimensionality of data by self-organizing neural networks. In their comparison of non-parametric clustering methods on array data that I examine in the next section, Chen, Jaradat, and Banerjee (2002) compared PAM and SOM to K-means and hierarchical clustering.

Other non-parametric clustering methods include fuzzy k-means clustering by Bezdek (1981). Fuzzy k-means is similar to K-means with the exception that in fuzzy k-means elements can belong to many clusters (as opposed to exactly one cluster in K-means.) In fuzzy k-means, elements belonging to multiple clusters have weights for their cluster assignments and these weights for each element sum to 1 in order for all elements to have equal influence in the final clustering. The rest of the fuzzy k-means procedure is similar to k-means. One other method is the quality threshold (QT) technique of Heyer, Kruglyak, and Yooseph (1999) in which the maximum diameter
for clusters is chosen \textit{a priori} and clusters are built. This method is rarely discussed in comparisons made to parametric clustering methods.

\section*{2.5 Quality of Clustering Techniques}

One aspect of clustering introduced by Rand (1971) and expanded upon by Chen et al. (2002) is robustness, the ability to retain a certain clustering results with respect to small changes in the data. Chen et al. (2002) explored a handful of traditional clustering techniques, including \textit{K}-means and hierarchical clustering. This paper also introduced a few indices to evaluate the strength of the clustering techniques as they state that "the hardest problem in comparing different clustering algorithms is to find an algorithm-independent measure to evaluate the quality of the clusters" (242). They introduce indices that evaluate clusters without any \textit{a priori} knowledge about the biological functions of the genes on the microarray in order to avoid trying to force the clusters to match some pre-conceived notion of connectivity based on biologic similarity. They then looked at the biologic qualities to evaluate the quality of the clustering \textit{a posteriori}.

Chen et al. (2002) concluded that \textit{K}-means performed better or as well as the other clustering techniques, which included hierarchical clustering, partitioning around medoids (PAM), and self-organization maps (SOM.) Chen et al. (2002) stated that

"\textit{K}-means generated clusters with slightly better structural quality \ldots \textit{K}-
means appeared more consistent with the biological information" (258).

The data set they used was the ES cell data set, consisting of 3805 genes and six time points for cDNA arrays of undifferentiated mouse R1 embryonic stem cells. Chen et al. (2002) claimed confidence in their results since this was a “difficult” data set, a fairly large yet sparsely sampled data set (the six time-points were unequally spaced from between 4 hours and 7 days from the start of the experiment.) They also perturbed their data in quite a few ways involving missing values and simulations with various variance levels and found K-means to provide better clustering than hierarchical clustering.

Based on their conclusions, I felt comfortable using K-means as a good representative of non-parametric clustering. This method is simple, well-studied, and appears to be of high enough quality to compare to potentially improved clustering provided by newer parametric techniques. Also, there is no generally accepted evaluation technique for hierarchical clustering to determine the correct number of clusters, like the Hartigan Rule for K-means. Thus, evaluation of simulation runs for determining whether hierarchical clustering finds the correct number of clusters cannot be done since this method relies on visual inspection or some other knowledge (like biology) of the data to determine the number of clusters. Hence, since K-means can be properly evaluated, it is the method I use in my simulations.
2.6 Shortcomings of Non-Parametric Clustering Methods

Some of the primary concerns of the traditional non-parametric clustering techniques include that these methods

- Do not take into account the ordering of time-course data

- Require the number of clusters to be chosen \textit{a priori}

- Cluster biologically unrelated genes together

- Ignore data resolution and noise level

- Do not take curve shape into account.

These shortcomings are mentioned in Schliep, Schonhuth, and Steinhoff (2003), Luan and Li (2003a), Yeung et al. (2001), and Ma, Castillo-Davis, Zhong, and Liu (2006). Of these five shortcomings, the first two are most often discussed in methods papers advocating use of a new parametric techniques like in Schliep et al. (2003). In hierarchical clustering and $K$-means, the distance between curves does not change if pairs of points are permuted. Papers which introduce model-based methods, which I describe in the next chapter, often discuss improvements to these shortcomings as reasons behind the improvement in clustering based on their methods.
2.7 Chapter Summary

In this chapter, I introduced the two stalwarts of non-parametric clustering techniques, hierarchical clustering and \( K \)-means analysis. These two methods have been applied to time-course array data since that type of data was first available. They are both well known, easy to implement in software, easy to understand in terms of output, and are still used quite often.

As I explained earlier in this chapter, Chen et al. (2002) showed that \( K \)-means proved to provide better clustering than hierarchical clustering for time-course array data. However, as the previous section explained, there are quite a few shortcomings of both of these non-parametric clustering techniques which have prompted the need and use of the more advanced parametric techniques I describe in the next chapter. I will compare these new methods with \( K \)-means and hierarchical clustering.
Chapter 3
Parametric Clustering Methods

3.1 Splines and Spline Clustering Methods

Splines are popular to fit to time-course curves because of their ease of use. Splines can be used to represent curves by specifying a series of points along a curve and defining functions allowing the curves between those points to be approximated. Clustering methods involving cubic splines, by Bar-Joseph, Gerber, Gifford, Jaakkola, and Simon (2002), and B-splines, by Luan and Li (2003b), have been introduced in recent years for time-course gene expression data.

3.1.1 Spline Terminology

Splines are piecewise polynomial functions that can approximate unknown curves. An order M spline is a piecewise polynomial of order M and has continuous derivatives up to order M-2. The endpoints of each piecewise section of the spline are called knots. A basis is the minimum set of vectors which generates the spline. B-Splines are special kinds of splines consisting of a linear combination of B-Spline basis curves.

Given \( m + 1 \) knots \( t_i \), with \( t_0 \leq t_1 \leq \ldots \leq t_m \), a B-Spline of degree \( n \) is a
parametric curve $S : [t_0, t_m] \rightarrow \mathbb{R}^2$ composed of basis B-splines of degree $n$:

$$S(t) = \sum_{i=0}^{m} P_i b_{i,n}(t), t \in [t_n, t_{m-n}]$$

where the $P_i$ are the control points and the $m - n$ basis B-Splines of degree $n$ are defined as:

$$b_{j,0}(t) = 1 \text{ if } t_j \leq t \leq t_{j+1} \ldots 0 \text{ otherwise}$$

$$b_{j,n}(t) = \frac{t-t_j}{t_{j+n}-t_j} b_{t,n-1}(t) + \frac{t_{j+n+1}-t}{t_{j+n+1}-t_{j+1}} b_{j+1,n-1}(t).$$

### 3.1.2 Spline Based Clustering

Modeling time-course expression profiles as piecewise polynomial cubic splines, estimated from observed data assumed to be coming from some overall smooth expression curve, was first advocated by Bar-Joseph et al. (2002). Their method involves constraining the spline coefficients of genes in the same cluster having similar expression patterns while also allowing for gene-specific parameters. This was one of many papers to use $K$-means as the non-parametric gold standard with which to compare a new model-based technique. The comparison of results of the $K$-means method to the new model-based methods will be discussed throughout this chapter.

Within a few months of the publication from Bar-Joseph et al. (2002), Luan and Li (2003b) published a proposal to cluster time-course gene expression data using mixed-
effects models with B-splines. This method was similar to one developed by James and Sugar (2003), which focused on sparsely sampled data (time-course expression profiles with measurements at between two and six time points.) The methods of James and Sugar (2003) and Luan and Li (2003b) are implemented in James and Sugar's software, which is commonly referred to as Functional Clustering Model (FCM.)

Luan and Li (2003b) developed a method using a mixed-effects model for time-course gene expression. The observed time-courses are received as viewed as samples from underlying continuous smooth curves. Each sample curve is represented as the sum of a smooth population mean spline function (dependent on time and the cluster to which the sample belongs,) a spline function with random coefficients representing individual gene effects, and Gaussian measurement noise. Under this model, Luan and Li (2003b) used the expectation-maximization (EM) algorithm of Dempster, Laird, and Rubin (1977) to cluster genes.

To explicitly define their model, the linear mixed model used was:

\[ y = X\beta + Z\gamma + \epsilon \]  

(3.1)

where

- \( y \) is an observed ordered vector of ordered mRNA responses (over a certain time-course).
- \( X \) and \( Z \) are design matrices associated with a vector of fixed effects \( \beta \) and a
vector of random effects $\gamma$.

- The random effects vector $\gamma$ has mean 0 and a covariance matrix $\Gamma$.

- $\epsilon$ is a vector of residual error terms with mean 0 and covariance matrix $\sigma^2 I$.

Luan and Li (2003a) applied to time-course array data the model developed by Rice and Wu (2001). This model is defined as:

$$ Y_i(t_{ij}) = \sum_{l=1}^{p} \beta_i^{(c)} B_l(t_{ij}) + \sum_{l=1}^{p} B_l(t_{ij}) \gamma_{il} + \epsilon_{ij} \quad (3.2) $$

where $Y_i(t_{ij})$ is the gene expression level for gene $i$ at time $t_{ij}$. Here, assume $n$ genes from $C$ different clusters are indexed $c = 1, \ldots, C$. The first term in Equation 3.2 (term between the equals sign and the first plus sign), models the mean or population average gene expression of the $c$th cluster, where the sum is over a fixed knots sequence. Also, $\bar{B} = \{ B_l(), l = 1, \ldots, p \}$ is a basis for a spline function on $[t_1, t_T]$ which uses B-spline basis with equally spaced knots and uses the same spline basis assumed for all $C$ clusters. $\beta_i^{(c)}$ is the $p$-vector of coefficients corresponding to cluster $c$.

The second term in Equation 3.2 (terms between the two plus signs) models the random effect for the $i$th gene where $B = \{ B_l(), l = 1, \ldots, q \}$ is a basis vector for a possibly different spline function on $[t_1, t_T]$. Also, the $\gamma_{il}$ are normal random coefficients with mean 0 and covariance matrix $Cov(\gamma_{il}) = \Gamma$. This term models the gene-specific deviation from the population average gene expression profiles. The
random effects term \( \gamma \) induces the covariance for a random curve \( Y_i(t) \) at two different time points \( t \) and \( s \):

\[
Cov(Y(s), Y(t)) = \sum_{k=1}^{q} \sum_{l=1}^{p} \Gamma_{kl} B_k(s) B_l(t) + \sigma^2 \delta(s - t)
\]

where \( \delta() \) is the Dirac function.

The residual term in Equation 3.2 (after the final plus sign) is uncorrelated normal measurement error assumed to satisfy \( E(\epsilon_{ij}) = 0, Var(\epsilon_{ij}) = \sigma^2 \).

For the purposes of clustering genes, \( Z_i \) is a cluster indicator for \( i \)th gene, \( Z = (Z_1, Z_2, \ldots, Z_n) \). The clustering is based on the EM algorithm of Dempster et al. (1977). The EM Algorithm calculates the expected value of a complete data log-likelihood and maximization of the expected complete data log-likelihood over model parameters. The EM algorithm estimates the parameters associated with the mixture model treating \( Z \) and the random coefficients \( \gamma \) as missing data. After convergence of the EM iterations, parameter estimates of \( \hat{\beta}_i^{(c)}, \hat{\Gamma}, \hat{\sigma}^2 \) are obtained, as well as posterior probability of gene \( i \) belonging to cluster \( c \):

\[
\hat{\pi}_{ci} = E[I(Z_i = c)].
\]

Then, cluster gene \( i \) into cluster \( c \) if \( \hat{\pi}_{ci} \) is the largest over all \( c = (1, \ldots, C) \). Note that only genes with a unique \( \max_c \hat{\pi}_{ci} \) are classified into one of the \( C \) clusters. The remaining genes remain unclassified.
The mixed-effects model allows the use of approximate Bayes factors (posterior odds for one model against another) to compare models, and hence can be used to determine the optimum number of clusters. When using the EM Algorithm to find maximum likelihood estimates, one rule often used for model selection is to maximize the Bayesian Information Criterion (BIC), defined as:

$$BIC(C) = 2L(C) - m_c \log(n).$$  \hspace{1cm} (3.3)

In equation 3.3, $L(C)$ is the maximized log-likelihood for the model with $C$ clusters and $m_c$ is the number of independent parameters to be estimated in the $C$-clusters model. Luan and Li (2003b) applied this FCM method and found seven clusters for the yeast data set whereas Spellman et al. (1998) found five clusters using hierarchical clustering.

### 3.2 Hidden Markov Model Clustering Methods and Bayesian Clustering Methods

Schliep et al. (2003) introduced a Hidden Markov Model approach to time-course clustering. In this method, each cluster is represented by a different Hidden Markov Model (HMM). Then, using an iterative procedure, this method clusters by an assignment of data points (curves) to these HMMs that maximizes the joint likelihood of clustering and models.
Bayesian techniques are not absent from the many proposed model-based clustering techniques. Ramoni et al. (2002a) introduced a Bayesian algorithm for Clustering by Dynamics (BCD). Their initial aims are “grouping time series clusters so that the elements of each cluster have similar dynamics” (91). BCD transforms each time series into a Markov Chain (MC) and then clusters similar MCs to discover the most probable set of generating processes. They expand some of the details of this process and apply it to gene-expression data in a follow up paper (Ramoni, Sebastiani, and Kohane (2002b).) They named their software CAGED (Cluster Analysis of Gene Expression Dynamics.)

Ramoni et al. (2002b) claimed that “the main contributions of this approach are the ability to take into account the dynamic nature of gene expression time-series during clustering and a principled way to identify the number of distinct clusters” (9121). They applied their technique to the human fibroblast data set. Ramoni et al. (2002b) found four clusters for the human fibroblast data set based on an autoregressive model of order 1 [AR(1)] that provided the best goodness of fit score to their model among all orders of AR models (10 clusters were found by hierarchical clustering in Iyer et al. (1999).) The sizes of these four clusters were 293, 216, 5, and 3 elements per cluster. The small cluster sizes of the last two clusters were a source of contention in papers which discount this method, like Schliep et al. (2003).
3.3 Smoothing Spline Clustering (SSClust)

In March 2006, another spline clustering method was introduced by Ma et al. (2006). This paper discounted the Bayesian methods of Ramoni et al. (2002a) and the Markov Model methods of Schliep et al. (2003) stating “such models often require stationarity and the Markov property, which are unlikely to hold for most time course microarray data” (1261).

They also claimed that the spline-based methods of James and Sugar (2003) and Luan and Li (2003b) may not be robust in that different number of bases and knots could lead to an array of different estimates of the underlying curves. Ma et al. (2006) also believed that computation improvements could be made in the Expectation-Maximization algorithm used in these methods.

Ma et al. (2006) proposed a method they called smoothing spline clustering (SSClust,) which is based on a mixed-effect smoothing spline model using the rejection controlled EM algorithm as used in Dempster et al. (1977) to estimate parameters. Ma et al. (2006) stated that their method accounts for natural differences in gene expression within a cluster of similarly expressed genes. They also claimed to provide goodness-of-fit via a mean curve and confidence bands and have an algorithm that is easily implemented in software. This software (SSClust) is freely available for download as source code in the R programming language.

To define their smoothing spline model, assume that an observed gene expression profile, \( y = (y_1, y_2, \ldots, y_n) \), has an underlying smooth curve \( f \) over time where the
pointwise errors are modeled by a normal distribution with mean 0 and a known constant variance.

\[ y_j = f(t_j) + \epsilon_j \text{ where } j = 1, 2, \ldots, T \]  

(3.4)

Least squares curve fitting minimizes the Residual Sum of Squares (RSS):

\[ RSS = \sum_{j=1}^{T} [y_j - f(t_j)]^2. \]  

(3.5)

Since there are many nonparametric functions that pass through all the data points, making \( RSS = 0 \), they look to avoid over-fitting. A popular constraint to avoid over-parametrization is to parameterize \( f(t) \) by a set of pre-specified basis functions. This can be done by using cubic splines or B-Splines, as advocated by Luan and Li (2003a). Ma et al. (2006) instead called for a more flexible strategy to impose a smoothness condition. This was accomplished by imposing the constraint that \( \int [f''(t)]^2 dt < \eta \) for some specific \( \eta > 0 \). By Lagrange multiplier theory, minimizing the RSS under the smoothness constraint is equivalent to minimizing the combined function:

\[ \sum_{j=1}^{T} [y_j - f(t_j)]^2 + \lambda T \int [f''(t)]^2 dt. \]  

(3.6)

The resulting solution for \( f \) is known as the cubic smoothing spline, a spline that is
a smoothed, piecewise polynomial. The analytic form for the cubic smoothing spline is:

$$\hat{f}(t) = d_0(\lambda) + d_1(\lambda)t + \sum_{j=1}^{T} c_j \int (t_j - u)_+ (t - u)_+ du$$  \hspace{1cm} (3.7)$$

where \((.)_+\) denotes the positive part of the number.

Once Ma et al. (2006) defined the modeling framework, they defined gene expression clusters. First, the mean curve for each cluster of genes is modeled by a smoothing spline. As in the other model-based methods, the time course gene expression for a gene in a given cluster is assumed to follow the shape of the mean curve but with additional gene specific shifts and each measurement at the time points are subject to normal measurement errors. For gene \(i\) in cluster \(k\), the mRNA expression at time \(t_{ij}\) (where the subscript \(j\) is a time-point label) is:

$$y_{ij} = \mu_k(t_{ij}) + b_i + \varepsilon_{ij}$$  \hspace{1cm} (3.8)$$

where \(\mu_k\) is the mean curve for cluster \(k\), \(b_i \sim N(0, \sigma_{bk}^2)\) is the gene specific deviation from the mean curve not due to measurement error, and \(\varepsilon \sim N(0, \sigma^2)\) is normal measurement error.

Under the conditions in our problem, Equation 3.8 is equivalent to:

$$y_i \sim N(\mu_k, \Sigma_k)$$  \hspace{1cm} (3.9)$$
where \( y_i \) and \( \mu_k \) are the vector representations of the expression observations and the mean curve respectively and \( \Sigma_k = \sigma^2_{bk} E_{TxT} + \sigma^2 I_{TxT} \) where \( E_{TxT} \) is a \( T \times T \) matrix of ones and \( I_{TxT} \) is a \( T \times T \) identity matrix.

Thus, the time course expression vector \( y_i \) can be expressed as:

\[
y_i = p_1 N(\mu_1, \Sigma_1) + p_2 N(\mu_2, \Sigma_2) + \ldots + p_k N(\mu_k, \Sigma_k)
\]  

(3.10)

where each of the \( y_i \) time course vectors is a mixture of normals, \( K \) is the total number of clusters, and the \( p \)'s are the relative proportions/sizes of the clusters. Hence, gene \( i \) has probability \( p_k \) of belonging to cluster \( k \) \textit{a priori}.

To take into account the smoothness constraint, this method uses Maximum Penalized Likelihood to estimate parameters. The penalized log likelihood is defined as:

\[
\sum_{i=1}^{n} \log(P_{ij}) - \sum_{i=1}^{n} \sum_{j=1}^{T} \left[ \frac{(y_{ij} - \mu_k(t_{ij}) - b_i)^2}{2\sigma^2} \right] - \sum_{i=1}^{n} \frac{b_i^2}{2\sigma_{bk}^2} - \lambda_k T \int [\mu_k'(t)]^2 dt + C
\]  

(3.11)

where the maximization of the function above, except for the logarithmic term, would yield an estimate of a smoothing spline for \( \mu_k \); also estimated are \( \sigma^2, \sigma_{bk}^2, p_k \). Direct maximization of:
\[ P(\text{gene}_i \in k) = \frac{p_k N(\mu_k, \Sigma_k)}{p_1 N(\mu_1, \Sigma_1) + p_2 N(\mu_2, \Sigma_2) + \ldots + p_K N(\mu_K, \Sigma_K)} \]  

(3.12)

is not analytically possible. Hence, a variation of the EM algorithm was developed.

The expectation step computes the probability that a particular gene belongs to each cluster given the model and current parameter estimates. The Maximization step maximizes the weighted penalized log-likelihood for each cluster:

\[ -\sum_{k=1}^{K} \sum_{i=1}^{n} P(\text{Gene}_i \in k) \left[ \sum_{j=1}^{T} \left( \frac{(y_{ij} - \mu_k(t_{ij}) - b_i)^2}{2\sigma^2} \right) + \frac{b_i^2}{2\sigma^2} \right] - \lambda_k T \int [\mu_k''(t)]^2 dt + C. \]  

(3.13)

In many cases, array data consists of a large number of genes. In these instances, the EM algorithm is too costly since it is maximizing a function that is the sum over all the genes in all clusters. Instead, the Rejection Controlled Expectation Maximization algorithm of Liu, Chen, and Wong (1998), which adds a Monte Carlo step to the EM algorithm, is used.

This is the method used to choose the number of clusters via minimizing the BIC:

\[ BIC = -2 \sum_{i=1}^{n} \log \left( \sum_{k=1}^{K} p_k N(\mu_k, \Sigma_k) \right) + \sum_{k=1}^{k} \nu_k \log(nT) \]  

(3.14)

where \( \nu_k \) is the number of free parameters in the \( k \)th cluster, which is defined as the
trace of its smoothing matrix obtained from Equation 3.8. Notice that this method minimizes the BIC in Equation 3.14, which is equivalent to maximizing $-1$ multiplied by Equation 3.14 (the form of the BIC equation shown in Equation 3.3.) The BIC imposes a penalty on the total number of parameters, scaled by the logarithm of sample size, to strike a balance between the goodness-of-fit and model complexity. The value of $k$ which minimizes the BIC provides the number of clusters in this clustering algorithm.

SSClust is downloadable as source code in the R programming language.

3.4 MClust

One model based method often mentioned in comparisons to SSClust, and other spline, Bayesian, or Markov methods is implemented in MCLUST, the software developed by Fraley and Raftery (1998) to handle time-course gene expression data. This software was later advocated and made freely available by Yeung et al. (2001) making their technique one of the first model-based clustering techniques to have software freely available for public use. This appears to be the benchmark method to which all other parametric methods compare.

The mixture model of Yeung et al. (2001) assumes that each cluster of the data is generated by an underlying probability distribution. Suppose that data $y$ consists of independent multivariate observations $y_1, y_2, ..., y_n$. Let $G$ denote the number of clusters in the data. The likelihood for the mixture model is
\[ L_{MIX}(\theta_1, \ldots, \theta_G | y) = \prod_{i=1}^{n} \sum_{k=1}^{G} \tau_k f_k(y_i | \theta_k) \]  

(3.15)

where \( f_k \) and \( \theta_k \) are the density and vector of parameters of the \( k \)th cluster in the mixture model and \( \tau_k \) is the probability, \( a \ priori \) that an observation belongs to the \( k \)th component \( (\tau_k \geq 0 \text{ and } \sum_{k=1}^{G} \tau_k = 1) \).

In the framework of the Gaussian mixture model, each component \( k \) is modeled by the multivariate normal distribution with mean vector \( \mu_k \) and covariance matrix \( \Sigma_k \):

\[ f_k(y_i | \mu_k, \Sigma_k) = \frac{\exp\{-\frac{1}{2}(y_i - \mu_k)^T \Sigma_k^{-1}(y_i - \mu_k)\}}{\sqrt{\det(2\pi \Sigma_k)}}. \]  

(3.16)

Yeung et al. (2001) represent the covariance matrix in terms of its eigenvalue decomposition given by

\[ \Sigma_k = \lambda_k D_k A_k D_k^T \]  

(3.17)

where \( D_k \) is the orthogonal matrix of eigenvectors, \( A_k \) is a diagonal matrix whose elements are proportional to the eigenvalues of \( \Sigma_k \), and \( \lambda_k \) is a scalar. The matrix \( D_k \) determines the orientation of the component, \( A_k \) determines its shape, and \( \lambda_k \) determines its volume.

Yeung et al. (2001) writes
<table>
<thead>
<tr>
<th>Model Name</th>
<th>Covariance Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal Volume Spherical (EI)</td>
<td>$\Sigma_k = \lambda I$</td>
</tr>
<tr>
<td>Unequal Volume Spherical (VI)</td>
<td>$\Sigma_k = \lambda_i I$</td>
</tr>
<tr>
<td>Unconstrained Model (VVV)</td>
<td>$\Sigma_k = \lambda_k D_k A_k D_k^T$</td>
</tr>
<tr>
<td>Equal Volume Elliptical (EEE)</td>
<td>$\Sigma_k = \lambda D A D^T$</td>
</tr>
<tr>
<td>Diagonal Model</td>
<td>$\Sigma_k = \lambda_k B_k;</td>
</tr>
</tbody>
</table>

Table 3.1: The various MCLUST models with different covariance structures

“allowing some but not all of the parameters in [3.17] to vary results in a set of models within this general framework that is sufficiently flexible to accommodate data with widely varying characteristics” (978).

They outline various techniques for the covariance structure of the models. One of the eight techniques that they examine, the Equal Volume spherical model (denoted EI), is quite similar to $K$-means. In this case, $D_k A_k D_k^T$ is constrained to equal the identity matrix and Equation 3.17 is reduced to

$$\Sigma_k = \lambda I.$$ (3.18)

Yeung et al. (2001) writes

“The classical iterative $K$-means clustering algorithm, first proposed as a heuristic clustering algorithm, has been shown to be very closely related to model-based clustering using the EI model, as computed by the EM algorithm. Celeux and Govaert (1992)” (979).
However, the EI model is just one of multiple models that are implemented in the MClust software and often times this is not the result which is determined as optimal. The different covariance structures of the models analyzed in MCLUST are described in Table 3.1.

Model selection (in terms of the number of clusters and which of the models is optimal) in MClust, is similar to the smoothing spline method in that the Bayesian Information Criterion is optimized based on the maximum likelihood of each model found by the EM algorithm. But, MCLUST selection criteria takes into account model complexity among the various variance structures. Their BIC model "rewards a model that fits the data well...discourages overfitting by penalizing models with more free parameters" (979).

This MCLUST approach by Yeung et al. (2001) is still popular and is available for implementation in the R programming language under the library “mclust”. This is one the methods of the compared to SSClust in Ma et al. (2006).

3.5 Comparisons from the SSClust Methods Paper

Ma et al. (2006) concluded with examples showing the “superiority” of their method (SSClust) over other parametric models as well as K-means on a set of simulated data. They simulated time-course profiles according to four curve types for a total of 150 curves (plus random noise) according to the structure:

\[(1) \ y_{tij} = \frac{(e^{t})}{1000} + \epsilon_{tij}, \text{30 curves}\]
(2) \[ y_{2ij} = \frac{\tan(t_i)}{e_{ij}} + \varepsilon_{2ij}, \quad 40 \text{ curves} \]

(3) \[ y_{3ij} = \frac{5(t_i - 4)^2}{\max((t_i - 4)^2)} + \varepsilon_{3ij}, \quad 50 \text{ curves} \]

(4) \[ y_{4ij} = \cos(t_i) + \varepsilon_{4ij}, \quad 30 \text{ curves}. \]

The time points are at \( t = 1, 2, 3, \ldots, 10 \).

The simulations were generated using Gaussian noise with between curve variances and between time-point covariance structured according to:

\[
\begin{align*}
Var(\varepsilon_{1ij}) &= 1, \quad Cov(\varepsilon_{1ij}, \varepsilon_{1ik}) = 0.2 \\
Var(\varepsilon_{2ij}) &= 2, \quad Cov(\varepsilon_{2ij}, \varepsilon_{2ik}) = 0.3 \\
Var(\varepsilon_{3ij}) &= 1, \quad Cov(\varepsilon_{3ij}, \varepsilon_{3ik}) = 0.2 \\
Var(\varepsilon_{4ij}) &= 2, \quad Cov(\varepsilon_{4ij}, \varepsilon_{4ik}) = 0.2.
\end{align*}
\]

The functional forms of these four curves are shown in Figure 3.1. Notice the difference in scaling on the y-axes between these groups. Ma et al. (2006) applied SSClust to each of 100 simulated data sets to determine how well the algorithm was able to recover the true number of clusters, the mean curve for each function, and the true classification of the expression profiles (curves).

A simulation which yielded perfect clustering (all simulated curves coming from one generating function were assigned to the same cluster for each of the four clusters/four curves) is shown in Figure 3.2. When the clustering was not perfect, Ma et al. (2006) developed comparison criteria. One of these measures was the misclassification rate (MR), which they defined as:
Figure 3.1: Functional Form of the Four Curves in the Simulation from the SSClust Methods Paper
Figure 3.2: Clustering results as outlined in the Ma et al. (2006) Paper

\[ MR = \frac{\text{The number of misclassified curves}}{\text{The total number of curves}} \quad (3.19) \]

For methods where the number of clusters are not needed to be fixed \textit{a priori}, Ma et al. (2006) also defined the Overall Success Rate (OSR) as the the fraction of times an algorithm recovered the correct number of clusters multiplied by 1 minus the MR, where the misclassification rate is the average of the MR values when the correct number of clusters is found.

\[ OSR = (\% \text{ of times the correct numbers of clusters was found})(1 - MR) \quad (3.20) \]
<table>
<thead>
<tr>
<th>Clustering Method</th>
<th>Distance Metric</th>
<th>MR(%)</th>
<th>Correct Number of Clusters (%)</th>
<th>OSR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means</td>
<td>Pearson</td>
<td>2.64</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>K-means</td>
<td>Euclidean</td>
<td>9.73</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>SSClust</td>
<td>NA</td>
<td>0.13</td>
<td>100</td>
<td>98.7</td>
</tr>
<tr>
<td>MCLUST</td>
<td>NA</td>
<td>0.38</td>
<td>77</td>
<td>69.5</td>
</tr>
<tr>
<td>CAGED</td>
<td>NA</td>
<td>11.07</td>
<td>14</td>
<td>2.93</td>
</tr>
</tbody>
</table>

Table 3.2: Clustering performance over 100 simulation iterations based on the simulated data from and reported in Ma et al. (2006)

Ma et al. (2006) compared clustering results from SSClust to clustering results of other methods, namely K-means, MCLUST, CAGED, and FCM. Since K-means and the spline FCM method both require the number of clusters to be specified a priori, they "gave a significant starting advantage to the K-means and the FCM algorithms by letting the number of clusters k be the true number of clusters (four)" (1265.) For K-means, they set the number of starting chains to be 5 with random initial cluster classifications. For MCLUST, the reported method is the one with the optimal BIC among the different models with different covariance structures.

Table 3.2 shows the misclassification rates and overall success rates for the various clustering methods on the simulated data as reported in Ma et al. (2006). They did not use a validation method like the Hartigan Rule to find the number of clusters as suggested for K-means. Instead, for K-means, they only reported the average misclassification rates with the a priori number of clusters set to the true number of clusters (4.) Looking at 3.2, SSClust clustered this data best followed by MCLUST, with CAGED performing poorly. Based on these results, Ma et al. (2006) concluded:
<table>
<thead>
<tr>
<th>Clustering Method</th>
<th>Parametric or Non-Parametric</th>
<th>Model Selection Criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means</td>
<td>Non-Parametric</td>
<td>Hartigan Rule</td>
</tr>
<tr>
<td>H-clust</td>
<td>Non-Parametric</td>
<td>Ad Hoc Methods</td>
</tr>
<tr>
<td>SSclust</td>
<td>Parametric</td>
<td>BIC Criterion</td>
</tr>
<tr>
<td>MCLUST</td>
<td>Parametric</td>
<td>BIC Criterion</td>
</tr>
</tbody>
</table>

Table 3.3: Clustering methods I will use in my analysis of time-course data and simulations throughout the remainder of this thesis.

"These results suggest that SSclust outperforms K-means and FCM (even under the ideal scenario where the correct number of k clusters is provided to K-means and FCM a priori) as well as MCLUST and CAGED" (1265).

The consistency of the misclassification rate and the overall success rate as measures of evaluation will be examined in the next chapter as I look again at this simulation and develop simulations which resemble actual time-course expression data better than in this example.

### 3.6 Chapter Summary

Table 3.3 summarizes the key clustering methods I will be using throughout the remainder of my thesis. The parametric methods have clearly defined selection criteria for letting the model choose the number of clusters while the non-parametric methods are often used with the number of clusters chosen via "visual inspection" of the clustering results or via some notion of how many clusters there should be from some biologic expectation of cell-cycle behavior. K-means has the Hartigan Rule of thumb
to advise how many clusters there are but this rule is not always followed and the same *ad hoc* methods applied to hierarchical clustering are often applied to *K*-means.

This chapter provided an overview of introduces popular and recent model-based clustering methods to analyze time-course array data. The results and the conclusions from Ma et al. (2006) are interesting but they are limited to just one simulated data set. In the next chapter, I report on these methods and their associated software on other simulated data sets to see if the results continue to indicate superior performance from the Ma et al. (2006) SSClust approach.
Chapter 4

Empirical Comparison of Methods: Simulated Data

4.1 Introduction

This chapter uses simulated data to examine the performance of various non-parametric and parametric clustering algorithms described in the previous chapters. Recall the comparison of methods done for a simulated data set in Ma et al. (2006), the results of which I presented in Chapter 3.

The reported results from Ma et al. (2006) for this clustering are displayed again in Figure 4.1. These results prompted Ma et al. (2006) to make some conclusions about the clustering methods. Perhaps, this nearly perfect clustering in this example explains why they chose this data set to show the superiority of SSclust to model-based clustering software such as MCLUST, FCM and CAGED.

<table>
<thead>
<tr>
<th>Clustering Method</th>
<th>Distance Metric</th>
<th>MR(%)</th>
<th>Correct Number of Clusters (%)</th>
<th>OSR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means</td>
<td>Pearson</td>
<td>2.64</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>K-means</td>
<td>Euclidean</td>
<td>9.73</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>SSclust</td>
<td>NA</td>
<td>0.13</td>
<td>100</td>
<td>98.7</td>
</tr>
<tr>
<td>MCLUST</td>
<td>NA</td>
<td>0.38</td>
<td>77</td>
<td>69.5</td>
</tr>
<tr>
<td>CAGED</td>
<td>NA</td>
<td>11.07</td>
<td>14</td>
<td>2.93</td>
</tr>
</tbody>
</table>

Table 4.1: Clustering performance over 100 simulation iterations based on the simulated data from and reported in Ma et al. (2006)
Figure 4.1: Functional Form of the Four Curves in the Simulation from the SSClust Methods Paper
But, look at Figure 4.1 which displays the functional forms from which the simulations were created. The differences in the scale of the dependent variable between the four underlying curves that this clustering is based on makes this situation one which does not reflect the typical time-course expression curves (which often differ in location and spread but not this extreme in scale.) The range of values on the y-axis for standard gene expression data sets is usually in the range of [-3,3] while the range of values on the y-axis for values in this simulation are from approximately [-20,0],[0,20],[0,5], and [-1,1] and the differences in range between clusters is large compared to data sets that are usually seen for gene expression array data. The conclusions from Ma et al. (2006) would need to be validated on other data sets which more accurately resemble actual time-course array data.

4.2 Motivation for Simulation Study

As a motivating case study to illustrate additional performance of SSClust in comparison to other clustering methods, I conducted a small simulation study with 20 time-course profiles for each of 5 true curves for total of 100 curves. The five true curves are:

(1) \( f(t) = 2 \cos \frac{\pi t}{24} \)

(2) \( f(t) = 2 \sin \frac{\pi t}{24} \)

(3) \( f(t) = 2 - 4e^{-25t} \)

(4) \( f(t) = -2 + \frac{t}{12} \)
Figure 4.2: Functional Form of the Five Curves for Simulation

(5) \( f(t) = 0 \)

with time at \( t = 0, 2, 4, 6, 8, \ldots, 48. \)

The functional form of the curves are shown in Figure 4.2 where the scale of the dependent variable is the same in all five graphs. I simulated "observed" curve data by adding pointwise iid normal errors with mean 0 and standard deviation of 0.5. An sd=0.5 was selected because it was nearly one standard deviation of the entire data set of functional values from the five curves.
Figure 4.3: Incorrect Clustering by SSClust; The sine curves were divided into two clusters, and the linear and flat curves combined into one cluster.

Under 100 simulations, the clustering algorithm only found the correct clustering assignment 70 times. When clustering was incorrect, two different true clusters were combined into one while one true cluster was arbitrarily spliced into two clusters yielding clusters of sizes 40, 20, 20, a, and 20-a where $0 < a < 20$. The BIC criteria in some simulations called for four clusters by SSClust when in fact five is the true number of clusters when this arbitrary splicing was made. These numbers indicate that two clusters combined while one split apart.
Figure 4.3 is an example of the incorrect clustering where the sine curve is arbitrarily split while the linear and the flat curves are combined. Hence, SSClust does not appear to be infallible in choosing the correct number of clusters and additional simulations are needed to adequately compare clustering methods. The results from these simulations appear later in this chapter.

4.3 Simulation Configuration

I developed some simple simulated data to analyze the performance of several clustering methods under some various conditions. Simulated data sets are important in this area of applied statistics since real data sets are expensive and hard to gather (most papers use a subset of the same 5-10 data sets) and since you rarely know the absolute "truth" in clustering.

The need for simulation to compare clustering methods is crucial. I created data sets to mimic real-life data yet be flexible enough so as to force the clustering algorithms to have to analyze various real-life data situations. The factors considered in my simulation include:

**Distance Metric** Euclidean distance or Pearson correlation

**Number of Curves** Large \( (n = 3000) \) vs small \( (n = 100) \) sample size representing number of genes in our simulated data

**Number and Resolution of Time-Points** Full Dataset (25 evenly spaced time
points verses 50% of the original data achieved by eliminating every-other point
or randomly eliminating half of the original time points (same time points re-
moved on each curve)

Types of Curves Small number of true curves ($k = 4$ or $k = 5$) verses a large
number of curve types ($k = 7$ or $k = 8$)

Distribution of curves across cluster When $k = 4$, distribute as a 25%-25%-25%
25%-25% equal distribution of curve types verses an alternative 45%-25%-25%-5%
5% unequal distribution of curve types. When $k = 7$, distribute the 7 curves in
equal apportionment, 14.3% per cluster, or use a 20%-20%-20%-20%-12%-4%-4%
4% unequal distribution of curve types.

Noise Level Small level of noise for simulated data within curve type (less than
one-half of the standard deviation of the data set) verses large noise level for
simulated data within curve type (more than one-half of the standard deviation
of the data set.) For these simulated data, a standard deviation equal 1 is
large and a standard deviation equal to 0.5 is small (based on the scaling of the
function forms of the curve centers.)

4.4 Checking the Consistency of K-Means on Simulated Data

Before running through intense simulations, I looked at $K$-means to see if I can
use avoid having to compare results with the number of clusters pre-specified. In
determining the number of clusters, recall that there is the Hartigan rule applied to $K$-means to determine when to go from $k$ clusters to $k + 1$ clusters.

For the yeast data set, running $K$-means repeatedly on the same data set did not providing a stable clustering assignment. Figure 2.1 showed the variation in the sums of the within clustering sums of squares for different clustering results when starting with just one starting chain (for the yeast data set with an a priori seven clusters.) Each plot of sums of within cluster sums of squares verses run number should be a flat line in order for the different clusterings to be "equally correct" based on the initial starting set for the same set of initial conditions.

The choice of initial starting sets has an impact on our clustering. One solution proposed is that instead of running $K$-means with the number of clusters pre-specified, pre-specify the centers of the $k$ clusters. But, this requires a preliminary analysis of the data to determine the centers before fully clustering. Also, if one cluster center is not closest to any element, the cluster contains no element on the first assignment of clusters and the algorithm will fail.

**Algorithm 3 K-Means with Multiple Starting Chains**

Initially, run $K$-means with the number of clusters specified and the “number of random sets” chosen for the initial cluster centers as 1.

**repeat**

Run $K$-means but increase the number of random sets by 1. The algorithm runs slower but is more accurate.

**until** Stop running when the algorithm is too slow or replicates the same clustering results on each run.

I first wanted to confirm that results from the yeast data on the impact of the
Figure 4.4: Simulated Data Sum of WCSS using: (a) one starting chain; (b) five starting chains; The true number of clusters is four.

The number of initial starting chains in the K-means algorithm is not an artifact of the data set. I ran simulated data (four groups of curves (cosine, sine, exponential, and linear as described in curves (1), (2), (3), and (4) in Section 4.2 using the structure of Algorithm 3. Figure 4.4 shows a graph sum of the within cluster sums of squares for each run over 100 simulation runs. When the sum of the WCSS is at the minimum value in these graph, this corresponds to the situation when the clustering finds the correct number of clusters. As can be seen in Figure 4.4, the dependent variable is at its minima is 54 times and 97 times respectively in the two graphs. When one starting chain is used, the overall success rate is 54% and increases to 97% when five starting chains are used. When the clustering was incorrect, and the correct number
Figure 4.5: Sum of the WCSS using: (a) One starting chain; (b) Five starting chains; The true number of clusters is four but clusters are of unequal size.

of clusters was not found, it was often the case that the linear and exponential graphs were combined into one cluster.

When the clusters were in the unequal 45-25-25-5% apportionment (with the smallest cluster being that of the linear graph), again when the curves were clustered incorrectly, the linear curves were again combined with the exponential curves.) The overall success rate increased from 43% with just one starting chain to 92% when five starting chains are used as shown in Figure 4.5.

As a whole over my simulations, as the number of initial random sets increased from one to five to 10 (regardless of the apportionment of curves, the number of curves being simulated (n=100 or n=3000 curves), or number of time points (full data set or
cut in half,)) the OSR increased from approximately 0.4 to 0.9 to 0.95. Hence, this choice of initial random sets has an effect on our clustering. This result is important as this is rarely reported for any method that criticizes the $K$-means in comparison studies.

4.5 Choosing the Methods and Evaluation Techniques

I compared the performance of three clustering methods in my simulations. The non-parametric representative was $K$-means. Recall from earlier that Chen et al. (2002) showed that $K$-means proved to provide better clustering than Hierarchical Clustering for time-course array data. Most parametric clustering methods papers use $K$-means as the convenient non-parametric technique for which to compare to since hierarchical clustering generally requires ad hoc methods to determine the number of clusters.

MCLUST was an easy choice in my simulations as it is the parametric method which is most compared to by new parametric methods. This method outperformed CAGED, the Bayesian method, which performed the worst among all methods in Ma et al. (2006). Recall that Ma et al. (2006) attacked CAGED in its introduction. This paper did not attack MCLUST but did find it to be inferior to SSClust in simulation, but it was the best among other parametric methods where the number of clusters did not have to be chosen a priori.

SSClust was the spline method I used in my simulations. The choice of SSClust over FCM was easy since, as shown in Ma et al. (2006), SSClust outperformed FCM
and FCM requires the number of clusters to be chosen prior to each run. Computational speed proved not to be an issue throughout these simulations and among Rand’s criteria, accuracy of clustering is the focus of this research.

My goal with these simulations is to see if indeed SSClust is superior to both MCLUST and \( K \)-means as reported in Ma et al. (2006). As for the evaluation technique, I use the misclassification rate as defined in the previous chapter. Since I am comparing between results on data sets of different sizes and noise levels, this seems to be the evaluation method advocated in Ma et al. (2006).

\[
MR = \frac{\text{the number of misclassified curves}}{\text{the total number of curves}} \tag{4.1}
\]

The misclassification rate is restated here where the number of misclassified curves are found by clustering the simulated data and comparing the clustering results to the true curves that generated the simulated data.

The algorithm for creating cluster assignments to determine misclassification rate is Algorithm 4. There are advantages and disadvantages to using misclassification rate. Advantages are that it is simple to use and easy to understand for clustering with different cluster sizes, noise levels, etc. It also takes into account when a method finds the incorrect number of clusters. When the number of clusters is not equal to the number of functions, the clustering greatly penalized.

Disadvantages are that the MR is only a number. Two methods can have the
Algorithm 4 Mapping Clusters to Functions to determine Misclassification Rate

Initially, data are generated from $m$ different functional forms.
These data are then clustered into $k$ clusters by a clustering method.
Find the centers of these $k$ clusters.
Find the distances between the centers of the $k$ clusters and each of the $m$ functions.
if $m = k$ then
  Map each cluster to the function that’s closest to it.
else if $m < k$ then
  For the $m$ clusters that have the smallest distances to one of the $m$ functions, map each cluster to the function that’s closest to it. The other $k-m$ clusters are automatically misclassified.
else if $m > k$ then
  For all $k$ clusters, calculate the smallest distances to each of the $m$ functions and map each cluster to the function that’s closest to it. The elements of the clusters that are not in each of the $k$ functions which were closest are hence misclassified.
end if

same misclassification rate if one method finds the correct number of clusters and the other does not. That first method might have some elements permuted between clusters, especially if two or more true clusters are similar. The second one might have two different clusters combined while another true cluster has an arbitrary split. While this second case of finding the wrong number of clusters is the greater mistake, both of these methods yield the same misclassification rate.

The use of the Overall Success Rate clears up this situation. The OSR considers all clusterings where the incorrect number of clusters is found as wrong and only considers the misclassification rate on cases when the correct number of clusters is found. Before using misclassification rate, I will specify whether it is on the whole data set regardless of whether the clustering algorithm found the correct number of clusters (as I used it in this chapter) or only in situations when the clustering found
the correct number of clusters (as in the next chapter when I use the OSR.)

Recall the definition of the OSR as the fraction of times an algorithm recovered the correct number of clusters multiplied by one minus the MR, where the misclassification rate is the average of the MR values when the correct number of clusters is found.

\[
OSR = \left( \% \text{ of times the correct numbers of clusters was found} \right) (1 - MR) \quad (4.2)
\]

Both the MR and the OSR will be used in evaluation of simulations throughout the rest of this thesis.

### 4.6 Comparison of Methods

#### 4.6.1 Simulation Set 1

For the first simulated data set, the mean curves from which simulations were drawn are:

\[
f(t) = 2 \cos \frac{\pi t}{24}
\]

\[
f(t) = 2 \sin \frac{\pi t}{24}
\]

\[
f(t) = 2 - 4e^{-25t}
\]

\[
f(t) = -2 + \frac{1}{12}
\]

with time at \( t = 0, 2, 4, 6, 8, \ldots, 48. \)
<table>
<thead>
<tr>
<th>Time Points (K-Means)</th>
<th>Distance Distribution</th>
<th>Curve Noise SD</th>
<th>Method MR% (KM)</th>
<th>Method MR% (MC)</th>
<th>Method MR% (SSC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Pearson 25-25-25-25%</td>
<td>0.5</td>
<td>1.39</td>
<td>0.00</td>
<td>5.90</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean 25-25-25-25%</td>
<td>0.5</td>
<td>0.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson 25-25-25-25%</td>
<td>1.0</td>
<td>2.26</td>
<td>0.00</td>
<td>9.10</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean 25-25-25-25%</td>
<td>1.0</td>
<td>1.25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson 45-25-25-5%</td>
<td>0.5</td>
<td>1.56</td>
<td>0.00</td>
<td>7.87</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean 45-25-25-5%</td>
<td>0.5</td>
<td>0.93</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson 45-25-25-5%</td>
<td>1.0</td>
<td>3.61</td>
<td>0.00</td>
<td>8.70</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean 45-25-25-5%</td>
<td>1.0</td>
<td>2.10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson 25-25-25-25%</td>
<td>0.5</td>
<td>1.78</td>
<td>0.00</td>
<td>4.33</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean 25-25-25-25%</td>
<td>0.5</td>
<td>0.41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson 25-25-25-25%</td>
<td>1.0</td>
<td>1.87</td>
<td>0.06</td>
<td>8.30</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean 25-25-25-25%</td>
<td>1.0</td>
<td>1.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson 45-25-25-5%</td>
<td>0.5</td>
<td>1.64</td>
<td>0.00</td>
<td>3.60</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean 45-25-25-5%</td>
<td>0.5</td>
<td>1.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson 45-25-25-5%</td>
<td>1.0</td>
<td>4.60</td>
<td>0.82</td>
<td>2.70</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean 45-25-25-5%</td>
<td>1.0</td>
<td>1.80</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Average MR over all simulated data sets: 1.81, 0.11, 6.31

Table 4.2: Misclassification Rates (MR%) for K-means (KM), MCLUST (MC), and SSClust (SSC) clustering; 4 true clusters; 100 simulated curves total. The misclassification rate is an average over 100 simulation iterations.

The simulations with differing curve apportionments and number of time points are described earlier in this chapter. There are 100 total curves from which our simulations were derived and the pointwise noise is Normally distributed with mean 0 and standard deviation either 0.5 (small) or 1.0 (large.) When the apportionment is unequal (45-25-25-5%) they are drawn with those percentages in the order the functions appear above with the cosine curve simulated from 45% and the linear curve simulated from 5%.

Table 4.2 shows the average misclassification rate over 100 runs for 16 different
<table>
<thead>
<tr>
<th>Time Points</th>
<th>Distance</th>
<th>Curve Distribution</th>
<th>Noise SD</th>
<th>Method K-Means</th>
<th>Method MCLUST</th>
<th>Method SSclust</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Pearson</td>
<td>25-25-25-25%</td>
<td>0.5</td>
<td>90</td>
<td>100</td>
<td>77</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>25-25-25-25%</td>
<td>0.5</td>
<td>93</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>25-25-25-25%</td>
<td>1.0</td>
<td>87</td>
<td>100</td>
<td>65</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>25-25-25-25%</td>
<td>1.0</td>
<td>92</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>45-25-25-5%</td>
<td>0.5</td>
<td>85</td>
<td>100</td>
<td>70</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>45-25-25-5%</td>
<td>0.5</td>
<td>88</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>45-25-25-5%</td>
<td>1.0</td>
<td>80</td>
<td>100</td>
<td>66</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>45-25-25-5%</td>
<td>1.0</td>
<td>87</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>25-25-25-25%</td>
<td>0.5</td>
<td>86</td>
<td>100</td>
<td>83</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>25-25-25-25%</td>
<td>0.5</td>
<td>96</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>25-25-25-25%</td>
<td>1.0</td>
<td>85</td>
<td>100</td>
<td>67</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>25-25-25-25%</td>
<td>1.0</td>
<td>92</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>45-25-25-5%</td>
<td>0.5</td>
<td>85</td>
<td>100</td>
<td>86</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>45-25-25-5%</td>
<td>0.5</td>
<td>88</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>45-25-25-5%</td>
<td>1.0</td>
<td>77</td>
<td>100</td>
<td>90</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>45-25-25-5%</td>
<td>1.0</td>
<td>84</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Average over all simulated data sets | 87 | 100 | 76

Table 4.3: Percentage of runs that the clustering technique found the correct number of clusters; 4 true clusters; 100 simulated curves total. The simulation configuration is as in Table 4.1.
simulated data sets. The choice of metric is only applicable in the $K$-means case. It should be noted that the misclassification rates above are in comparison for four clusters over all 100 runs, whether or not the method found the correct number of clusters. This was done since, for example, sometimes curves were misclassified appearing to come from 4 clusters even when the algorithm correctly found 4 clusters by validation criteria and sometimes curves were classified correctly even when the clustering found 3 clusters by validation measure. With curves coming from mean curves which were quite different between the groups, most of the misclassification came from these results came from finding the wrong number of clusters (where many curves would be misclassified) and not from excessive noise alone (where a few curves would be misclassified.)

Looking at the results from Table 4.2, MCLUST performed the best for these simulations, followed by $K$-means, and SSCLust performed the worst based on average misclassification rates across all simulations. The percentage of runs that the clustering technique found the correct number of clusters appears in Table 4.3 and these results confirm the same order of the performance of the three clustering methods.

As seen in Figure 4.3, again here SSCLust combines two clusters while arbitrarily splitting a true cluster regardless of time points, curve apportionment, or noise. This is the newest and the most computationally intensive technique but it performs the worst. For the cases where the arbitrary split was made, the BIC criterion often called for 3 clusters when in fact there were 4 true clusters in table 4.3. The clusters most
prevalently combined were the exponential and the linear curve clusters, although they were not split more prevalently when the linear curve provided just 5% of the total number of curves. For SSClust, I used the input values for the algorithm as an RCEM threshold value of 0.3 and 2 starting chains for the RCEM, which are sufficient for viable output according to the directions for use of the software.

To analyze the source of the highest misclassification rates and lowest percentage of time that the correct number of clusters were found, I ran regressions on the results from all 8 rows in both Table 4.2 and Table 4.3. I regressed on the number of time points, the curve distribution, and the noise level. For the misclassification rate and the correct number of clusters, no variable was significant at the 0.05 significance level in either regression and thus no variable stands out as having the greatest influence on the clustering difficulties (this includes the missing time points as well as increased noise.)

Implementations in K-means (4 starting chains) also had a tendency to combine two clusters while splitting a true cluster. As with SSClust, the clusters most prevalently combined were the exponential and the linear curve clusters, while either of the sine or cosine curves featured an arbitrary split. In this situation, the Hartigan Rule called on occasion for 3 clusters when in fact there were 4 true clusters. Unlike in SSClust, where the BIC criterion only pointed to 3 clusters in clustering cases when 2 clusters were combined and one featured an arbitrary split, in K-means the Hartigan rule sometimes called for 3 clusters but also called for between 5 and 7 clusters
when the clustering was incorrect in Table 4.3. But, at least 75% of the time that
the incorrect number of clusters was called for, the algorithm found 3 clusters and
featured the familiar joining and arbitrary split described above.

To analyze the sources of misclassification and of finding the incorrect number of
clusters, I again ran regressions similar to those described for SSCLust above with the
exception that here 16 rows of data were included in each regression with the addition
of the variable of distance metric. Looking at the regressions on misclassification
rate, an increase in noise level was a significant factor to increase the MR ($p < 0.01$) and in the direction that is expected (more noise leads to higher MR.) The
distance metrics were also significant ($p < 0.01$) as Euclidean distance outperformed
Pearson correlation in all cases. This was due to the fact that when four clusters
were found, more curves were misclassified due to noise under Pearson correlation
than with Euclidean distance (discovered via visual inspection of clustering results.)
Also, the curve distribution was significant ($p = 0.02$) as a move from an equal to
an unequal apportionment of curves increased the misclassification rate. These same
three variables, in the same direction, were found to be significant to decrease the
percentage of runs that the algorithm found the correct number of clusters.

MCLUST appears to cluster this data nearly perfectly and only rarely does it
misclassify the noisy curves for each noise level. This method finds the correct number
of clusters 100% of the simulations in all cases. In regression runs similar to those
for the results in SSCLust, no variable was found to be significant influencing the
misclassification or finding the correct number of clusters. This data set should be an easy data set since the distances between the true clusters in this data set was simulated from was much larger than the distances between centers of clusters from typical sets of real data. It is not a surprise for the well-tested MCLUST clustering algorithm to do well in this instance.

Overall, K-means, and especially SSClust, were disappointing for an “easy” data set to cluster given the differences in the true underlying curves between each cluster and the relatively small amount of noise in the simulated data; see Figure 4.2. While regression revealed factors that affected clustering in K-means, no variables proved significant in regressions on SSClust results. In these simulations, there were only 4 true clusters and 100 total curves. I will examine the effects of more true clusters and more curves in later simulation sets.

4.6.2 Simulation Set 2

For the second simulated data set, the number of mean curves was increased from four to seven to see how the clustering algorithms handled a greater number of clusters. The mean curves from which simulations were drawn are:

\[ f(t) = 2 \cos \frac{\pi t}{4} \]
\[ f(t) = 2 \sin \frac{\pi t}{4} \]
\[ f(t) = 2 - 4e^{-2t} \]
\[ f(t) = -2 + \frac{t}{12} \]
Figure 4.6: Underlying functions for the Seven Curves for Simulation Set 2

\[ f(t) = 0 \]

\[ f(t) = \log\left(\frac{t}{6} + 1\right) \]

\[ f(t) = \sin\frac{\pi t}{24} \]

with time points at \( t = 0, 2, 4, 6, 8, ..., 48 \).

The graphs of the seven true curves are shown in Figure 4.6. Here, there are 175 total curves from which our simulations were derived and the pointwise noise is normally distributed with mean 0 and standard deviation either 0.5 (small) or 1.0
<table>
<thead>
<tr>
<th>Time Points</th>
<th>Distance</th>
<th>Curve Distribution</th>
<th>Noise SD</th>
<th>Method MR% (KM)</th>
<th>Method MR% (MC)</th>
<th>Method MR% (SSC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Pearson</td>
<td>14.3% per clust.</td>
<td>0.5</td>
<td>17.76</td>
<td>0.10</td>
<td>4.76</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>14.3% per clust.</td>
<td>0.5</td>
<td>7.06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>14.3% per clust.</td>
<td>1.0</td>
<td>14.45</td>
<td>6.50</td>
<td>8.77</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>14.3% per clust.</td>
<td>1.0</td>
<td>7.95</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>7.79</td>
<td>0.34</td>
<td>4.98</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>11.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>8.95</td>
<td>4.45</td>
<td>7.73</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>11.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>14.3% per clust.</td>
<td>0.5</td>
<td>22.31</td>
<td>0.61</td>
<td>10.39</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>14.3% per clust.</td>
<td>0.5</td>
<td>7.08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>14.3% per clust.</td>
<td>1.0</td>
<td>16.67</td>
<td>10.55</td>
<td>13.49</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>14.3% per clust.</td>
<td>1.0</td>
<td>10.09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>8.34</td>
<td>0.47</td>
<td>5.67</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>12.39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>9.43</td>
<td>4.93</td>
<td>8.82</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>12.22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average MR over all simulated data sets</td>
<td></td>
<td></td>
<td></td>
<td>11.60</td>
<td>3.49</td>
<td>8.08</td>
</tr>
</tbody>
</table>

Table 4.4: Misclassification Rates for K-means, MCLUST, and SSCLust clustering methods; seven true clusters, 175 simulated curves total from these 7 true clusters; The misclassification rate is the average over 100 simulation iterations.

(large.) When the apportionment is equal, each curve is simulated from 25 times.

When the apportionment of curves is unequal, 20-20-20-20-12-4-4% rather than equal apportionment, the curves were apportioned in the order they appear above and thus the flat curve was represented 12%, while the log and the sine (amplitude=1) were represented 4% each.

Table 4.4 shows the average misclassification rate over 100 runs for 16 different simulated data sets. The misclassification rate is found the same way as for the simulations in the previous subsection. Again, the choice of metric is only applicable
<table>
<thead>
<tr>
<th>Time Points</th>
<th>Distance</th>
<th>Curve Distribution</th>
<th>Noise SD</th>
<th>Method K-Means</th>
<th>Method MCLUST</th>
<th>Method SSClust</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Pearson</td>
<td>25% per clust.</td>
<td>0.5</td>
<td>58</td>
<td>100</td>
<td>90</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>25% per clust.</td>
<td>0.5</td>
<td>66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>25% per clust.</td>
<td>1.0</td>
<td>61</td>
<td>81</td>
<td>75</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>25% per clust.</td>
<td>1.0</td>
<td>67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>64</td>
<td>100</td>
<td>87</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>66</td>
<td>86</td>
<td>77</td>
</tr>
<tr>
<td>25</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>72</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>25% per clust.</td>
<td>0.5</td>
<td>51</td>
<td>100</td>
<td>72</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>25% per clust.</td>
<td>0.5</td>
<td>75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>25% per clust.</td>
<td>1.0</td>
<td>58</td>
<td>70</td>
<td>65</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>25% per clust.</td>
<td>1.0</td>
<td>70</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>72</td>
<td>100</td>
<td>83</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>0.5</td>
<td>63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Pearson</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>72</td>
<td>85</td>
<td>78</td>
</tr>
<tr>
<td>13</td>
<td>Euclidean</td>
<td>20x4,12x1,4x2%</td>
<td>1.0</td>
<td>75</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Average over all simulated data sets: 66, 90, 78

Table 4.5: Percentage of runs that the clustering technique found the correct number of clusters; seven true clusters; with 175 simulated curves from these 7 true clusters; The simulation configuration is as in Table 4.3.
for the $K$-means situation. Table 4.5 shows the percentage of runs that the the correct number of clusters were found in the corresponding simulation runs.

As in Simulation Set 1, the MCLUST algorithm, particularly at the low level of noise, outperformed the other two clustering algorithms. But, at the higher level of noise (SD=1), MCLUST misclassified curves by combining two clusters while arbitrarily splitting a true cluster (the algorithm found the incorrect number of clusters by BIC methods.) In these cases, MCLUST found five or six clusters when there were in fact seven true clusters. The clusters most often combined in the case of large level of noise (SD=1) were the two sine curve groups (with different amplitude.) This did not happen in the case of four true clusters as there, the number of clusters found was correct every time as seen in 4.3. Running regressions as in the previous subset, the only significant variable affecting misclassification rate and finding the correct number of clusters for MCLUST is noise ($p < 0.01$ for both regressions.) As expected, added noise decreases the quality of the clustering results. The apportionment of curves and the number of time points were not significant.

Analyzing $K$-means clustering, Table 4.4 shows that $K$-means misclassified more curves when there were four true clusters. In this case of 7 true clusters, $K$-means performed weakest of the three clustering algorithms (it had outperformed SSClust when there were only four clusters.) None of the variables (metric, number of time points, curve distribution, or noise level) were significant in influencing the misclassification rate but metric (Euclidean) and curve distribution (unequal apportionment)
were significant in improving the finding of the correct number of clusters. Overall, between 5 and 6 and also between 8 and 10 clusters were found when the incorrect number of clusters was determined by the Hartigan Rule.

Unlike MCLUST and K-means, Table 4.4 shows that SSClust actually performed better for the larger number of true clusters than it did when there were just four clusters in 4.2. Here, SSClust outperformed K-means yet still did not perform as well as MCLUST at the smaller level of noise (SD=0.5.) Again, like in the case of four clusters, no predictor variable proved significant in regressions run on the SSClust results.

To compare the clustering results among clustering methods between the simulation set of four true clusters in Simulation Set 1 and the seven true clusters of Simulation Set 2, I plotted misclassification rates from Table 4.4 verses those from Table 4.2 for corresponding entries of distance metric (for K-Means), number of time points, curve distribution, and noise. This plot is shown in Figure 4.7. Except for SSClust, it appears that the clustering methods perform better when there are 4 true clusters than when there are 7 true clusters. I also regressed the results of Tables 4.2 and 4.4 combined. In a regression of misclassification rate on the clustering method, number of time points, curve distribution, noise, and the number of true clusters, only the clustering method and the number of clusters were significant ($p < 0.01$.)

As a whole, the clustering algorithms performed better on the simulations from the last section (four true clusters) than from this section (seven true clusters.) This
Figure 4.7: Misclassification rates for the three clustering methods for seven true clusters versus four true clusters; Points on this graph correspond to different distance metrics (when applicable), number of time points, curve distribution, and noise.
Figure 4.8: Rates of finding the correct number of clusters in clustering runs for the three clustering methods for seven true clusters verses four true clusters; Points on this graph correspond to different distance metrics (when applicable), number of time points, curve distribution, and noise.
could be due to having more true clusters or having true clusters that are more similar to each other and the effect of the number of clusters will be examined later in this chapter. Also, the choice of clustering algorithm was significant as overall, MCLUST performed better for these simulations than either $K$-means and SSClust while $K$-Means and SSClust were not significantly different from each other. The level of noise was marginally significant ($p = 0.06$) for the combined results with a lower level of noise leading to a lower misclassification rate.

I also plotted the rates for finding the correct number of clusters for the clustering runs with seven true clusters and four true clusters in Figure 4.8. I performed a similar regression here as well on the entire combined data set from Tables 4.3 and 4.5. The results were markedly similar to the results of the regression on misclassification rate as both the true number of clusters and the clustering algorithm were significant but here level of noise was significant ($p = 0.03$) as well (lower noise leading to better results as expected.)

Next, I examine the impact of the size of the data set (number of curves to be clustered) on clustering methods.

4.6.3 Simulation Set 3

The goal of this section is to see if the number of curves being clustered has an impact on clustering accuracy (via misclassification rate.) Here, I performed similar simulations as the previous two subsections but with a different total number of
<table>
<thead>
<tr>
<th>Distance</th>
<th>Noise</th>
<th>MR(%)</th>
<th>MR(%)</th>
<th>MR(%)</th>
<th>MR(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>n=100</td>
<td>n=500</td>
<td>n=1500</td>
<td>n=3000</td>
</tr>
<tr>
<td>Pearson</td>
<td>SD=0.5</td>
<td>1.39</td>
<td>0.56</td>
<td>0.97</td>
<td>0.81</td>
</tr>
<tr>
<td>Pearson</td>
<td>SD=1.0</td>
<td>2.26</td>
<td>0.70</td>
<td>1.81</td>
<td>1.33</td>
</tr>
<tr>
<td>Euclidean</td>
<td>SD=0.5</td>
<td>0.75</td>
<td>0.48</td>
<td>0.83</td>
<td>0.56</td>
</tr>
<tr>
<td>Euclidean</td>
<td>SD=1.0</td>
<td>1.24</td>
<td>0.33</td>
<td>0.40</td>
<td>0.21</td>
</tr>
</tbody>
</table>

Table 4.6: Misclassification Rates for K-means for 4 true clusters at 4 different total numbers of curves

<table>
<thead>
<tr>
<th>Distance</th>
<th>Noise</th>
<th>MR(%)</th>
<th>MR(%)</th>
<th>MR(%)</th>
<th>MR(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>n=175</td>
<td>n=700</td>
<td>n=1400</td>
<td>n=3500</td>
</tr>
<tr>
<td>Pearson</td>
<td>SD=0.5</td>
<td>17.76</td>
<td>15.43</td>
<td>13.11</td>
<td>8.41</td>
</tr>
<tr>
<td>Pearson</td>
<td>SD=1.0</td>
<td>14.45</td>
<td>13.13</td>
<td>14.19</td>
<td>9.69</td>
</tr>
<tr>
<td>Euclidean</td>
<td>SD=0.5</td>
<td>7.06</td>
<td>5.96</td>
<td>8.78</td>
<td>4.46</td>
</tr>
<tr>
<td>Euclidean</td>
<td>SD=1.0</td>
<td>7.95</td>
<td>4.60</td>
<td>8.11</td>
<td>5.00</td>
</tr>
</tbody>
</table>

Table 4.7: Misclassification Rates for K-means for 7 true clusters at 4 different total number of curves

curves per simulation run. The data used here are the same as in the two previous subsections where there are 4 and 7 true clusters. Here, however, I looked at cases where the total number of curves increases from 100 to 3000. Previously, the true clusters each consisted of the same number of curves and each curve consisted of the same number of time points (25.) In the case of 4 true clusters, I used overall sample sizes of 100 (same as in the previous simulations), 500, 1500, and 3000 curves while in the case of 7 true clusters, I used overall sample sizes of 175 (same as in the previous simulations), 700, 1400, and 3500 curves.

In K-means analysis with 4 true clusters, Table 4.6 shows as the number of curves being clustered increases, there is a slight downward trend in the misclassification
Table 4.8: Misclassification Rates for MCLUST for 4 true clusters at 4 different total number of curves

<table>
<thead>
<tr>
<th>Noise</th>
<th>MR(%) n=100</th>
<th>MR(%) n=500</th>
<th>MR(%) n=1500</th>
<th>MR(%) n=3000</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD=0.5</td>
<td>0.00</td>
<td>0.02</td>
<td>0.03</td>
<td>0.02</td>
</tr>
<tr>
<td>SD=1.0</td>
<td>0.00</td>
<td>0.04</td>
<td>0.03</td>
<td>0.03</td>
</tr>
</tbody>
</table>

Table 4.9: Misclassification Rates for MCLUST for 7 true clusters at 4 different total number of curves

rate. But, in a regression of misclassification rate on metric, noise, and sample size, the only significant predictor is metric, where the misclassification rate is lower for Euclidean Distance than for Pearson Correlation. Thus, sample size is not significant.

For 7 true clusters in K-means, Table 4.7 shows that again there is a decrease in the misclassification rate as sample size increases. But, unlike in the case of 4 true clusters, in a regression using the same predictors, sample size is a significant predictor in the regression (along with metric as again Euclidean has a lower MR.) In K-means analysis, an increase in the size of the data set does not adversely affect the clustering results and, for the case of a higher number of true clusters, the algorithm performs better on larger data sets.

The misclassification rates in MCLUST for 4 true clusters shown in Table 4.8 do not change when there are more curves in each cluster at either level of noise.
Table 4.10: Misclassification Rates for SSClust for 4 true clusters at 4 different total number of curves

<table>
<thead>
<tr>
<th>Noise</th>
<th>MR(%) n=100</th>
<th>MR(%) n=500</th>
<th>MR(%) n=1500</th>
<th>MR(%) n=3000</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD=0.5</td>
<td>5.90</td>
<td>10.50</td>
<td>8.89</td>
<td>6.77</td>
</tr>
<tr>
<td>SD=1.0</td>
<td>9.10</td>
<td>12.52</td>
<td>8.56</td>
<td>7.98</td>
</tr>
</tbody>
</table>

Table 4.11: Misclassification Rates for SSClust for 7 true clusters at 4 different total number of curves

<table>
<thead>
<tr>
<th>Noise</th>
<th>MR(%) n=175</th>
<th>MR(%) n=700</th>
<th>MR(%) n=1400</th>
<th>MR(%) n=3500</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD=0.5</td>
<td>4.76%</td>
<td>9.87%</td>
<td>8.34%</td>
<td>9.43%</td>
</tr>
<tr>
<td>SD=1.0</td>
<td>8.77%</td>
<td>10.76%</td>
<td>11.61%</td>
<td>11.48%</td>
</tr>
</tbody>
</table>

MCLUST easily handles this simulated data set as the size of the overall data set increases and regressions run on this data set reveal that noise and sample size are not significant predictors of misclassification rate.

When there are 7 true clusters as shown in Table 4.9, MCLUST again is nearly flawless at the low level of noise and improves at the higher noise level as the number of curves increases. But, regression analysis reveals that neither noise level nor sample size is a significant predictor of misclassification rate. Thus, statistically, MCLUST performs as well for small data sets as it does for large data sets regardless of the number of true clusters.

Using SSClust, the results for 4 true clusters are shown in Table 4.10. Using the same initial conditions as described earlier (2 RCEM starting chains and threshold value of 0.3,) this algorithm appears to perform similarly in MR for different sample sizes. Regression analysis shows that neither noise nor sample size in significant in
predicting misclassification rates.

For SSClust with 7 true clusters, the results are shown in Table 4.11. Again, neither variable was significant although noise level was marginally significant \( p = 0.07 \). Thus, like MCLUST, the model-based clustering method SSClust does not appear to have the misclassification rate affected by the number of curves being clustered.

The analysis above comparing the clustering results for different sample sizes are all separate for different numbers of true clusters. But, examining the impact of sample size and the number of true clusters simultaneously will help examine the consistency of the misclassification rate, and hence the overall success rate, as measures of clustering quality of each clustering method.

For each of the three clustering methods, I regressed the misclassification rate as a function of overall sample size, the noise levels, the number of clusters, and, in the case of \( K \)-means, distance metric. For both parametric clustering methods, SSClust and MClust, neither the overall sample size of the data set nor the number of clusters was significant in this regression. This is encouraging in that these methods should not only yielded consistent results for large and small data sets but also for a large and small number of clusters. For \( K \)-means, the number of clusters was significant as the algorithm performed significantly better for the smaller number of clusters. This is one of the weaknesses of the \( K \)-means method regardless of the size of the data set.

Overall, the size of the data set (number of curves being clustered) did not change
the clustering algorithms performances in terms of misclassification rates in all cases except for $K$-means. This is encouraging as the data sets in this field can be massive and the clustering algorithms need to be able to handle these large sample size in order to be useable. This is also encouraging in examining the consistency of the overall success rate, which is a function of the misclassification rate.

The OSR was defined in Equation 4.2. If the OSR changes as the sample size increases or the number of clusters increases, then that would indicate that the OSR would not be a consistent measure when examining clustering on a data set with more curves or more clusters.

Figure 4.9 shows the OSR plotted against the number of curves for both sets of simulated data from this section (4 and 7 true clusters) for each of the three clustering algorithms. The noise level is SD=0.5 and the distance metric from $K$-means analysis is Euclidean distance. Recall, using the OSR, the misclassification rate is only for the situation when the correct number of clusters is found.

As can be seen in Figure 4.9, for SSClust and MCLUST the OSR is consistent across the size of the data set while for $K$-means, the size of the data set appears to have an impact on the overall success rate. But, in a regression of the OSR verses the number of curves, the number of curves is not significant ($p > 0.05$) in an overall regression nor in separate regressions for each clustering method. Thus, indeed the OSR appears to be a consistent measure for analyzing the success of a clustering method on a data set.
Figure 4.9: Overall Success Rate vs Number of Time Course Profiles for two simulated data sets with 4 true clusters and 7 true clusters respectively
4.7 Chapter Summary

Overall, for my simulated data cases, MCLUST appeared to be the superior method at finding the correct number of clusters in the simulated data from this chapter. Too often, in all subsets, regardless of the number of clusters or number of time points, both $K$-means and SSCLust found the incorrect number of clusters and yielded the higher misclassification rate. Hence, the conclusions of Ma et al. (2006) that stated that “SSCLust outperforms $K$-means...as well as MCLUST” (1265) are in doubt as a general rule and perhaps might be just an artifact of their particular simulated data sets.

The number of true clusters did have an impact on clustering but the overall size of the data set, when the number of clusters was fixed, did not have an adverse effect on the performance of the clustering algorithms. The next chapter reports on comparisons based on real data and simulations based on real data. These comparisons should provide some more insight into the comparison of the clustering methods under consideration.
Chapter 5

Empirical Comparison of Methods: Real Data

5.1 Clustering Real Data

In this chapter, I use the yeast and human fibroblast data sets to compare the non-parametric $K$-means and the parametric models MCLUST and SSCLust. An advantage to using real data sets is that since they come from a real experiment, they detail actual biologic events and thus have some meaning to biologists. Another advantage is that some microarray data sets are familiar and the same data set may be used in dozens of papers, thus making comparisons of methods easier. In comparison to simulated data sets where clusters are generated from true curves, the disadvantage of real data sets is that the “truth” of the clustering of data set is often unknown. But, as I show later in this chapter, real data can be perturbed and used to generate simulated data from which the clustering of the original data set can be used as truth to avoid this pitfall.

5.2 Clustering Yeast Data

One data set that has been analyzed using most of the new model-based and compared to the traditional non-parametric methods is the yeast data set generated by Spellman et al. (1998). Spellman et al. (1998) cited the five cell-cycles phases for
Figure 5.1: Hierarchical Clustering for the yeast data with 5 clusters chosen a priori
the reasoning behind choosing a priori five clusters in their hierarchical clustering analysis. Hierarchical clustering results are shown in Figure 5.1.

Luan and Li (2003a) used a mixed-effects model with B-Splines and found seven clusters for these data. Schliep et al. (2003) used a Hidden Markov Model to obtain five clusters corresponding to five different cell-cycle phases (M/G1, G1, S, S/G2, and G2/M) corresponding to the number of clusters obtained by Spellman et al. (1998). These five clusters found by Schliep et al. (2003) were different from the five clusters found by Spellman et al. (1998). Schliep et al. (2003) used the Ramoni et al. (2002a) Bayesian AR model and found only two clusters for the yeast data.

The differences in the clustering results described above show a lack of clustering consensus of the yeast data. The Bayesian AR model appears to split the data into two clusters, with the amplitude of the curve from 0 as the deciding factor for cluster assignment. The Hidden Markov Model appears to split the clusters by the presence of absence of periodicity in the data and then by amplitude and shape. The seven clusters from the mixed-effects model with B-Splines appear to take on seven distinct shapes.

I used non-parametric K-means and the newer parametric models MCLUST and SSClust to analyze the yeast data as I did with the simulated data.
Figure 5.2: Plot of Hartigan Rule value vs number of clusters for yeast data; the rule crosses y=10 at x=19

5.2.1 Applying K-means to the Yeast Data

Applying K-means (Euclidean Distance metric) to the yeast data, the Hartigan Rule of thumb found 19 clusters as shown in Figure 5.2. Figure 5.3 shows a K-means run of 19 clusters. Upon visual inspection, some of these clusters look quite similar to one another (like clusters 5 and 9 or clusters 4 and 10) or appear to have dissimilar curves clustered together (like clusters 6 and 13.)

5.2.2 Applying SSClust to the Yeast Data

Applying SSClust to the yeast data set yields 10 clusters by BIC optimization as seen in Figure 5.4. The cluster sizes ranged from 6 to 85 with only one cluster having
Figure 5.3: Plots of the 19 clusters of the yeast data as identified by K-means and Hartigan Rule
Figure 5.4: Plots of the 10 clusters for the yeast data in SSClust
<table>
<thead>
<tr>
<th>Clustering Algorithm</th>
<th>Number of Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical Clustering</td>
<td>5</td>
</tr>
<tr>
<td>K-Means</td>
<td>19</td>
</tr>
<tr>
<td>SSClust</td>
<td>10</td>
</tr>
<tr>
<td>MCLUST</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 5.1: Number of Clusters found via various clustering methods for the yeast data

fewer than 15 elements. Upon visual inspection, the 10 clusters appear to be distinct from each other.

5.2.3 Applying MCLUST to the Yeast Data

Running the original yeast data set through MCLUST, this method yields 9 clusters as seen in Figure 5.5 by BIC optimization. Upon visual inspection, the nine clusters appear to be distinct from each other. No cluster has fewer than 10 elements from the 433 element data set.

5.2.4 Clustering Summary: Yeast Data

Table 5.1 summarizes the number of clusters found via four different clustering algorithms for the yeast data set. The non-parametric clustering algorithms yield markedly different results as K-means found 19 clusters while the data set was clustered in Spellman et al. (1998) via hierarchical clustering and five clusters. Both parametric clustering algorithms yielded a similar number of clusters although the results visually are different from each other. Later in this chapter, I will examine the
Figure 5.5 : Plots of the nine clusters in MCLUST for Yeast Data
Figure 5.6: Hierarchical Clustering for the Human Fibroblast data with 10 clusters chosen a priori.

stability of these clustering results to noise added to these data sets in an attempt to determine which algorithm best clusters this data set.

5.3 Clustering Human Fibroblast Data

In addition to the yeast data set, the human fibroblast data set as described in chapter one and in Iyer et al. (1999) is also a popular data set used to compare clustering methods. Iyer et al. (1999) used hierarchical clustering to find 10 clusters for this data.
Hierarchical clustering on this data with 10 clusters chosen *a priori* is shown in Figure 5.6. The variance of this data set is smaller than that in the yeast data set and there is less periodicity than in the yeast data set.

Luan and Li (2003a) used a mixed-effects model using B-Splines and found seven clusters for the human fibroblast data. The seven clusters found in the human fibroblast data by Luan and Li had a smallest cluster having just six elements. Ramoni et al. (2002a) used their Bayesian AR model and found four clusters including clusters having three and five elements. These sparse clusters were one of the bases for the concerns about this method as stated in Luan and Li (2003a).

I again used *K*-means, MCLUST, and SSeClust to analyze the human fibroblast data as I did with the yeast data.

### 5.3.1 Applying *K*-means to the Human Fibroblast Data

Applying *K*-means under the Hartigan Rule for the human fibroblast data set found 20 clusters as shown in Figure 5.7. Recall, that *K*-means using the Hartigan Rule found 19 clusters for the yeast data, while other parametric and non-parametric methods called for between 5 and 10 clusters. Again, it appears that the Hartigan Rule is relatively liberal in determining the number of clusters. This rule of thumb is not helpful if the clustering is unhelpful or nonsensical.

Looking at Figure 5.7, the curve looks to flatten out at around 14 clusters. Figure 5.8 shows the 20 clusters of the Human Fibroblast data using *K*-means clustering.
Figure 5.7: Plot of Hartigan Rule Value vs Number of Clusters for Human Fibroblast Data

Many of these clusters look similar (like clusters 6, 9, and 12). Recall that the regressions in the last chapter showed that $K$-means performed worse for a higher number of clusters and perhaps this is having an impact in the clustering results here.

5.3.2 Applying SSClust to the Human Fibroblast Data

Figure 5.9 shows the four clusters found for the HF data in SSClust. This clustering appears to be an improvement from $K$-means since some of these clusters here have distinctive qualities (increasing patterns, decreasing patterns, and sinusoidal patterns.)
Figure 5.8: Plots of the 20 clusters of the HF Data by K-means under the Hartigan Rule for choosing the number of clusters.
Figure 5.9: Plots of the four clusters found in HF Data by SSClust
5.3.3 Applying MCLUST to the Human Fibroblast Data

Clustering the human fibroblast data set with MCLUST found only three clusters as shown in Figure 5.10. Overall, this clustering appears similar to that from clustering via SSClust, as shown in Figure 5.9, with the exception that SSClust found one additional cluster and appears to have more distinctly similar patterns within clusters.

5.3.4 Clustering Summary: Human Fibroblast Data

Table 5.2 shows the number of clusters found for the human fibroblast data set from various clustering algorithms. MCLUST found one fewer cluster (3 verses 4) for the human fibroblast data set compared to SSClust and, as seen in Table 5.1, MCLUST
<table>
<thead>
<tr>
<th>Clustering Algorithm</th>
<th>Number of Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical Clustering</td>
<td>10</td>
</tr>
<tr>
<td>K-Means</td>
<td>20</td>
</tr>
<tr>
<td>SSClust</td>
<td>4</td>
</tr>
<tr>
<td>MCLUST</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 5.2: Number of Clusters found via various clustering methods for the human fibroblast data

...found one fewer cluster (9 verses 10) for the yeast data set as compared to SSClust.

Again, as in the case of the yeast data, *K*-means found by far the greatest number of clusters but some of these cluster appears to be quite similar.

But, what do these clustering results say about underlying biological groupings? This issue is explored in Chapter 6. In the next sections of this chapter, I consider robustness of the clustering algorithms using the original clusterings as “truth” within each clustering method.

### 5.4 Simulations Based on Real Data

Simulated data sets are nice but do not always capture the essence of real-life situations. Simulations based on real data sets are one way to improve the authenticity of simulated data sets. I started with real data and added small uncorrelated errors to each term (each time point in each gene.) These sort of simulated data sets are important and can be an improvement over other simulate data sets since these data sets are less likely to be carefully crafted to show superiority of one method.

This simulation method extends real data in a way that is inexpensive and efficient
and can show the strength of a clustering method in terms of their resistance to small changes in the data set (displayed in simulation via noise.) To get these results:

1. Obtain a real data set, such as the yeast data

2. Cluster the results based on a given method

3. Perturb the original data set (for example, by adding noise at each time point)

4. Evaluate how different the new clustering result is from that based on the original data.

5.5 Simulations based on the Yeast Data

I compared the number of clusters and the misclassification rates (and Overall Success Rates) for different levels of noise for three different clustering methods (K-means, SSClust, MCLUST.) To perturb the yeast data, I added normal independent noise $N(\mu = 0, \sigma = 0.1, 0.25, \text{or} \ 0.5)$ at each of 18 time points for n=433 time-course profiles. I used the Euclidean Distance Metric and 5 starting chains for the K-means algorithm (the same as I used for the yeast data earlier in this chapter.)

A standard deviation of $\sigma = 0.1$ is small compared to the standard deviation of $\sigma = 0.6$ for the entire true data set. I increased the noise to 0.25 (nearly half of an SD) and 0.5 (nearly one SD) to see how these changes affected the clustering.

Table 5.3 shows the fraction of curves that were misclassified from the "truth" for simulations based on real data sets (when the correct number of clusters was found)
<table>
<thead>
<tr>
<th>Clustering Method</th>
<th>Assessment</th>
<th>Noise(0.1)</th>
<th>Noise(0.25)</th>
<th>Noise(0.5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means</td>
<td>Misclassification Rate (%)</td>
<td>5.05</td>
<td>5.56</td>
<td>8.00</td>
</tr>
<tr>
<td>K-means</td>
<td>Correct Number of Clusters (%)</td>
<td>27</td>
<td>14</td>
<td>3</td>
</tr>
<tr>
<td>K-means</td>
<td>Overall Success Rate (%)</td>
<td>25.64</td>
<td>13.22</td>
<td>2.76</td>
</tr>
<tr>
<td>SSClust</td>
<td>Misclassification Rate (%)</td>
<td>4.62</td>
<td>5.64</td>
<td>5.41</td>
</tr>
<tr>
<td>SSClust</td>
<td>Correct Number of Clusters (%)</td>
<td>81</td>
<td>74</td>
<td>51</td>
</tr>
<tr>
<td>SSClust</td>
<td>Overall Success Rate (%)</td>
<td>77.26</td>
<td>69.83</td>
<td>48.24</td>
</tr>
<tr>
<td>MCLUST</td>
<td>Misclassification Rate (%)</td>
<td>3.69</td>
<td>5.54</td>
<td>11.50</td>
</tr>
<tr>
<td>MCLUST</td>
<td>Correct Number of Clusters (%)</td>
<td>51</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>MCLUST</td>
<td>Overall Success Rate (%)</td>
<td>49.01</td>
<td>6.61</td>
<td>7.08</td>
</tr>
</tbody>
</table>

Table 5.3: Clustering performance over 100 simulation iterations based on the Yeast Data

at different levels of noise as well as the fraction of times that the algorithm found the correct number of clusters. These values are then combined to give the overall success rate (via Equation 4.2). Thus, the misclassification rate here is different than in the simulations from last chapter as this misclassification rate is only for the cases when the correct number of clusters was found.

5.5.1 Applying K-means to the Yeast Simulated Real Data

Under K-means clustering, the overall success rate as seen in Table 5.3 is no higher than 26% and fell to just 3% for the highest level of noise. These results do not bode well for the stability of the K-means algorithm in practice. Histograms of the number of clusters found by the Hartigan Rule for each level of noise is shown in Figure 5.11. The number of clusters found ranged from 8 to 20 depending on the level of noise.

As the noise level increases, the number of clusters found decreases as it seems
Figure 5.11: Number of Clusters via Hartigan Rule for K-means Analysis of Yeast Data; The original unperturbed yeast data set was partitioned into 19 clusters by the Hartigan Rule.
that the distinctions between groups get blurred as more pointwise noise is added. While 19 clusters, the number of clusters found when no noise is added, is the mode for the number of clusters found at the lowest level of noise, the mode decreases to 14 clusters and 13 clusters respectively for the higher levels of noise. I performed a similar analysis on the human fibroblast data to see if these poor results are an artifact of the yeast data set, or are indicative of \( K \)-means as a whole when using the Hartigan Rule, and these results are shown later in this chapter.

### 5.5.2 Applying SSClust to the Yeast Simulated Real Data

Table 5.3 also shows the misclassification rate and overall success rate for simulations of the yeast data set clustering with SSClust. SSClust appears to be the best clustering method as it has the highest overall success rate (48% to 77%) at all three levels of noise among the three clustering algorithms. The advantage of SSClust over the other methods increases at higher levels of noise. When the incorrect number of clusters was found, normally the number of clusters was underestimated as can be seen in Figure 5.12. But, the mode at all three levels of noise was 10, which was the number of clusters in the unperturbed data.

### 5.5.3 Applying MCLUST to the Yeast Simulated Real Data

Table 5.3 also shows the MR and OSR of simulations on the yeast data clustered using MCLUST. Here, the results for MCLUST are disappointing, especially at the
Figure 5.12: Number of Clusters via BIC rules for SSClust Analysis of Yeast Data; The original unperturbed yeast data set was partitioned into 10 clusters by SSClust.
Figure 5.13: Number of Clusters via BIC rules for MCLUST Analysis of Yeast Data. The original unperturbed yeast data set was partitioned into nine clusters by MCLUST.
two higher levels of noise where the OSR is no higher than 7%. Looking at the simulation runs in MCLUST, the BIC curve is fairly flat and the model has a hard time finding the correct number of clusters.

As seen in Figure 5.13, between 6 and 10 clusters are found for SD=0.1 and this range widened as the noise increased. As the noise level increases, a loss in distinction between several clusters occurs and these clusters are combined in the model. The mode jumps from 9 clusters, the number of clusters of the unperturbed data and at the lowest level of noise, to 5 clusters and then to 6 clusters as the noise level increases.

MCLUST appears to be quite sensitive to the level of noise in the case of simulations on the real yeast data. Unlike the simulations in Chapter 4, which saw larger distances between average curves in pairs of clusters and MCLUST perform well, MCLUST does not cluster this data well. Overall, the clustering for the yeast data set with pointwise noise was most stable when using for SSClust rather than MCLUST or K-means. These results are shown in the Overall Success Rates at varying levels of noise.

5.6 Simulations Based on the Human Fibroblast Data

As with the yeast data, I compared the misclassification rate and OSR for the human fibroblast data for different levels of noise for three different clustering methods (K-means, SSClust, and MCLUST.) To perturb the human fibroblast data, I added
<table>
<thead>
<tr>
<th>Clustering Method</th>
<th>Assessment</th>
<th>Noise(0.1)</th>
<th>Noise(0.25)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means</td>
<td>Misclassification Rate (%)</td>
<td>6.12</td>
<td>7.50</td>
</tr>
<tr>
<td>K-means</td>
<td>Correct Number of Clusters (%)</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>K-means</td>
<td>Overall Success Rate (%)</td>
<td>14.08</td>
<td>1.85</td>
</tr>
<tr>
<td>SSClust</td>
<td>Misclassification Rate (%)</td>
<td>2.51</td>
<td>17.6</td>
</tr>
<tr>
<td>SSClust</td>
<td>Correct Number of Clusters (%)</td>
<td>79</td>
<td>67</td>
</tr>
<tr>
<td>SSClust</td>
<td>Overall Success Rate (%)</td>
<td>77.02</td>
<td>55.21</td>
</tr>
<tr>
<td>MCLUST</td>
<td>Misclassification Rate (%)</td>
<td>11.18</td>
<td>17.44</td>
</tr>
<tr>
<td>MCLUST</td>
<td>Correct Number of Clusters (%)</td>
<td>14</td>
<td>7</td>
</tr>
<tr>
<td>MCLUST</td>
<td>Overall Success Rate (%)</td>
<td>12.43</td>
<td>5.78</td>
</tr>
</tbody>
</table>

Table 5.4: Clustering performance over 100 simulation iterations based on the Human Fibroblast Data

normal independent noise $N(\mu = 0, \sigma = 0.1 \text{ or } 0.25)$ to the HF dataset (517 rows, 12 columns) I used the Euclidean Distance Metric and 5 starting chains for the K-means algorithm. The standard deviation of the human fibroblast data set ($\sigma = 0.29$) was less than half that of the yeast data set ($\sigma = 0.60$). Again, I increased the noise in the simulations up to 0.25 (nearly one SD) to see how these changes affected our clustering.

Table 5.4 shows the fraction of curves that were misclassified from the “truth” for simulations based on real data sets at different levels of noise as well. The fraction of times that the algorithm found the correct number of clusters was also found and these values are then combined to give the overall success rate (via Equation 4.2). Again, as in the previous section, the misclassification rate here is different than in the simulations from last chapter as this misclassification rate here is only for the clusterings when the correct number of clusters was found.
Figure 5.14: Number of Clusters via Hartigan Rule for K-Means analysis of Human Fibroblast Data; The original unperturbed yeast data set was partitioned into 20 clusters by the Hartigan Rule.

5.6.1 Applying K-means to the HF Simulated Real Data

Table 5.4 shows the MR and OSR for K-means clustering of the HF data. The overall success rate here is no higher than 14% and falls to 2% at the higher level of noise. These results do not bode well for the stability of the K-means algorithm in practice for data sets for which the amplitude of the curves is small. As seen in Figure 5.14, in K-means analysis the number of clusters found ranged from 10 to 21 clusters, with
Figure 5.15: Number of Clusters via BIC rules for SSClust Analysis of HF Data; The original unperturbed data set was partitioned into 4 clusters by SSClust.

modes at 15 clusters and 14 clusters respectively for the two levels of noise. Again, simulated data based on real data as truth, when the number of clusters found by the Hartigan Rule is high, yields poor clustering results all levels of noise using the K-means algorithm.
5.6.2 Applying SSClust to the HF Simulated Real Data

Table 5.4 also shows the MR and OSR for SSClust for the HF data based on the algorithm finding four true clusters. Again, as with the yeast data, SSClust had the highest OSR at both levels of noise and appears to be the most stable clustering algorithm. Figure 5.15 shows histograms of the number of clusters found at both levels of noise and the mode is clearly 4 clusters for each level of noise.

5.6.3 Applying MCLUST to the HF Simulated Real Data

Table 5.4 also shows the MR and OSR for clustering results using MCLUST. Again, looking at the simulations as I did with the yeast data, BIC curves are fairly flat and small perturbations in the data can lead to finding an incorrect number of clusters (the OSR is no higher than 13%). Between 3 and 9 clusters were found for SD=0.1 and the sharp distinctions between the 3 clusters found initially were blurred by the addition of noise as seen in Figure 5.16. The mode at both levels of noise was five clusters. Compared to the yeast data set, when the noise was small, MCLUST performed better for yeast than for the HF data and this might be due to the small variance between the clusters in the HF data set. MCLUST did not perform well for either data set at the higher levels of noise.

MCLUST appears to be quite sensitive to the level of noise in the case of simulations on the real HF data unlike the simulations in Chapter 4 which saw larger differences between any two clusters. This is because the differences between the
Figure 5.16: Number of Clusters via BIC rules for MCLUST Analysis of HF Data; The original unperturbed data set was partitioned into 3 clusters by MCLUST.
“true clusters” was small for the yeast and even smaller for the HF data. These results show the difficulty with finding a definitive number of clusters in a data set in MCLUST for real data sets. Again, as in the case of the yeast data, SSClust appears to provide the best, most stable clustering assignments in simulations based on the human fibroblast data set.

5.7 Examining the Hartigan Rule via Bootstrap Analysis

To examine the bias and variability of the Hartigan Rule and whether it is a consistent measure to find the number of clusters via $K$-means analysis, I used the yeast data set from this chapter and ran bootstrap analysis, Efron (1983), at four different percentages of overall sample size (25%, 50%, 75%, and 100%).

Figure 5.17 shows histograms of the number of clusters found in $K$-means, using the Euclidean distance metric, via the Hartigan Rule for bootstrap samples. Bootstrap samples were found with replacement from the yeast data with sample sizes of $n = (433, 325, 217, 108)$, with 100 bootstrap iterations per sample size. As the sample size decreases, the number of clusters found falls as expected since some of the original clusters had sample sizes less than 15 and thus a resampling of size 25% of the size of the original data set is expected to lose most elements of some of these smaller clusters. The median of number of clusters is 18 for resampling at 100% of the original sample size and falls to 16.5, 14, and 10.5 when resampling is done at 75%, 50%, and 25% of the original sample size respectively.
Figure 5.17: Number of clusters found via the Hartigan Rule in K-means for bootstrap (100 iterations per sample size) samples from the yeast data set at sample sizes of four different percentages of the overall sample size (n=433)
Figure 5.17 shows that the variability of the number of clusters found increases as sample size decreases and the bias in the number of clusters found increases as the sample size decreases. The number of clusters found via the Hartigan Rule appears somewhat normally distributed at 100% and 75% of the original sample size, but the distribution becomes skewed with a longer tail as the sample size decreases to 25%.

Next, I performed the same Hartigan Rule bootstrap analysis using simulated data from Chapter 4. The sizes of the clusters are equal and the "truth" is known for the simulated data sets and thus the hope is that the Hartigan Rule yields the correct number of clusters regardless of the percentage of original sample size used in the bootstrap analysis. Recall, Chapter 4 showed that the size of the data set had no significant effect on the misclassification rate in clustering. Thus, if the Hartigan Rule shows a bias based on sample size in this data set, then it is not a good rule of thumb to use to determine the number of clusters.

I first looked at the simulated data set from Section 4.6.1 with four true clusters and \( n = 400 \) curves, 100 in each cluster. I ran bootstrap analysis at four different percentages of overall sample size (25%, 50%, 75%, and 100%) using the Euclidean distance metric in \( K \)-means clustering with 25 time points and pointwise noise of \( sd=1.0 \). This situation corresponds to the fourth row in Tables 4.2 and 4.3.

Figure 5.18 shows the number of clusters found via the Hartigan rule for the bootstrap iterations of the simulated data with four true clusters at four different percentages of overall sample size. Four, the true number of clusters, was found by
Figure 5.18: Number of clusters found via the Hartigan Rule in K-means for bootstrap (100 iterations per sample size) samples from the simulated data set (with 4 true clusters simulated from 4 true mean curves) at sample sizes of four different percentages of the overall sample size (n=400)
the Hartigan Rule to be the number of clusters, at least 80% of the time at each sample size and four is the mode and the median of each graph. Thus, the Hartigan Rule handled this simulated data set quite well and did not show bias based on sample size. Perhaps the varied results from using the Hartigan Rule in the bootstrap iterations of the yeast data set is due to the small sample sizes of some of the clusters as well as the large number of clusters found in that data set, and not an inherent deficiency in using the Hartigan Rule.

To test this statistic once more, I performed the same bootstrap analysis using the same percentages of sample size for the simulated data set of Section 4.6.2 with seven true clusters. This data corresponds to the analysis in the fourth row of Tables 4.4 and 4.5. I performed this bootstrap analysis to see if more clusters, 7 here verses 4 in the last simulated data set, causes poorer clustering. If so, then this would indicate that the Hartigan Rule performs worse when clustering with a larger number of true clusters.

I performed this bootstrap analysis for overall sample size n=175 to see if the small sample sizes of the clusters (100 curves per true cluster before verses 25 curves per true cluster now) causes clustering to be poorer as the percentages of sample size taken for the sample decreases. If this is true, this would indicate that the Hartigan Rule performs worse when clustering data sets with a small cluster sample sizes (more sparse clusters.)

Figure 5.19 shows the number of clusters found in this simulation. Notice how
Figure 5.19 : Number of clusters found via the Hartigan Rule in K-means for bootstrap (100 iterations per sample size) samples from the simulated data set (with 7 true clusters simulated from 7 true mean curves) at sample sizes of four different percentages of the overall sample size (n=175)
7, the true number of clusters, was the mode and the median for bootstrap samples of 100% and 75% of the sample size of the full data set. Thus, these results confirm that Hartigan Rule does not perform worse for a higher number of true clusters and this is encouraging for our results on both the yeast and human fibroblast data sets.

But, looking at Figure 5.19, the appearances of these graphs resemble those in Figure 5.17 showing the Hartigan bootstrap simulations for the yeast data. As the fraction of resampling from the overall sample size decreases, the Hartigan Rule finds fewer clusters, here fewer than the actual number of clusters. Performing a Kruskal-Wallis rank-sum test on the elements of the histograms in Figure 5.19 shows that there is a significant difference between the number of clusters found in the four groups that differ by sample size ($p < 0.01$).

Since the original cluster sizes were 25, some of these cluster sizes are small in these resampling situations and like the yeast data these small cluster sample sizes might be causing the bias in the Hartigan Rule. This indicates that small cluster sizes have an effect on the Hartigan Rule and thus might have an effect on the Overall Success Rate of $K$-means clustering. Small cluster sample sizes have also been cited as issues in some parametric clustering methods as stated in Schliep et al. (2003). But, the number of clusters found in the data set is not a significant factor in effecting the Hartigan Rule. Also, when the cluster sizes are sufficiently large, the overall size of the data set is not a significant factor effecting the Hartigan Rule. Thus it appears that the Hartigan Rule is an acceptable measure for choosing the number of clusters
in $K$-means clustering.

5.8 Chapter Summary

Clustering these real data and simulations based on real data sets showed how the same data set can yield remarkably different clustering based on the clustering algorithm that was used. These real data set clustering assignments, within method, provided a truth for which to compare simulations. Based on these simulations, some conclusions can be drawn on clustering real data.

$K$-means performed poorly for both data sets. The number of clusters decreased to between one-half and three-quarters of the number of clusters of the original clustering. The Hartigan Rule appeared to be unstable as the noise increased in the data set, as the differences between clusters decreased due to the noise introduced. In a regression analysis of the overall success rates from each data set in Table 5.3 and Table 5.4 run with predictors of noise level and clustering method, $K$-means was found to have a significantly lower OSR ($p < 0.01$) than SSclust for both the yeast and the human fibroblast data sets. The noise level was also significant, as expected, in clustering the yeast data ($p = 0.02$) indicating more noise led to a lower OSR.

MCLUST performed poorly with the perturbed real data for both yeast and HF. Relative to the original clustering of nine groups of the yeast data, as the level of noise increased the algorithm did not find nine clusters, often-times underestimating the number of clusters (9) found in the original data. The opposite pattern proved true
for the HF data which had three "true" clusters. MCLUST appears sensitive to noise perturbations in the data. While MCLUST performed well for the simulated data in Chapter 4, for the real data clusters which were relatively less distinct from each other than the simulated data, the algorithm had difficulty finding the correct clustering. Again, performing a regression described above, MCLUST had a significantly lower OSR than SSClust for both data sets.

SSClust provided the most stable clustering between runs of the algorithm for finding the correct number of clusters. As stated by Ma et al. (2006), SSClust outperforms other clustering techniques for real data. While MCLUST performed nearly flawlessly for the simulations Chapter 4, in this chapter SSClust outperforms both MCLUST and K-means ($p < 0.01$) (but MCLUST and K-means did not have an OSR significantly different from each other for the second-best method.) Even though SSClust is the most computationally intensive method, perhaps the method is worthwhile if the clustering is much improved. The next chapter moves into the field of examine the role of gene ontology in clustering time-course expression data.
Chapter 6
Gene Ontology: Methods and Results

6.1 Introduction to Gene Ontology

The Gene Ontology (often denoted GO) project (Ashburner, Ball, Blake, Botstein, Butler, Cherry, Davis, Dolinski, Dwight, et al. (2000)) provides a controlled vocabulary to describe genes and gene product attributes in any organism. The effort is collaborative among biological researchers around the world. The coordinators of the project developed three vocabularies, called ontologies, that describe gene products, in a species-independent manner. The ontologies are defined in terms of molecular function (MF) of gene products, their role in multi-step biological processes (BP), or their cellular components (CC). Ontology files are freely available from the GO website, http://www.geneontology.org/. The ontologies provide a vocabulary for representing and communicating knowledge about the set of genes at hand. The controlled vocabularies are structured so that a search can be conducted for a few specific genes or an entire genome for an organism.

Terms from the gene ontology can be used in the annotation of gene products in a biological database. Gene Ontology annotations are the associations between made between gene products and the GO terms describing them. A gene product can be an RNA or protein product encoded by a gene. A gene product can be annotated
Figure 6.1: Two directed graphs for two genes from the Human Fibroblast data set using GO Molecular Function annotation

to 0, 1, or multiple nodes of an ontology and annotations of a gene product in one ontology are independent to its annotation in other ontologies.

Annotation data can be described through a series of directed acyclic graphs (DAGs). Mathematically, this is a graph which is represented by a series of nodes and edges $G = \{N, E\}$. Figure 6.1 shows two such graphs for the MF annotation for two genes from the human fibroblast data set. The node at the head of the arrow is the “parent” of the node at the tail of the arrow, while the node at the tail is the
“child” of the node at the head of the arrow. Looking at Figure 6.1, the head at the
bottom of each graph is “all”, the universe of all ontologies. The child of this term is
the term with GO ID “GO:0003674”, which corresponds to the universe of “molecular
function,” one of the three ontologies. If the graph is read from bottom to top, the
node at the top of the graph on the left in Figure 6.1 is the GO node associated with
gene “AA045003” (as denoted by GenBank,) a gene from the human fibroblast data
set. The graph on the right in Figure 6.1 has two GO child nodes for the annotation
of gene “AA044605,” also from the HF data set, where the two paths in the graph
meet at the term annotated to “molecular function.”

In previous chapters, different non-parametric and parametric clustering tech-
niques were used to cluster time-course expression profiles including some newer
model-based methods that have been advocated in recent years. However, these
model-based methods themselves are criticized by Pan (2006) in that “most existing
methods, including model-based clustering, ignore known gene functions in cluster-
ing” (795.)

This ignorance of the gene functions, when clustering expression data, is seen as a
major flaw in clustering expression data. The issue as to how best to incorporate such
gene functions in the clustering of RNA transcripts is an open question and the main
subject of this chapter. It is expected that incorporating additional biological data
into clustering of gene expression data will provide more biologically interpretable
clustering as well as perhaps improved clustering. Questions of how to quantify a
biologic distance between genes and how to incorporate this biologic information in the clustering models for the expression data will be explored in this chapter, with some new ideas developed and enforced. In this chapter, I will first cluster by expression value, then cluster by gene ontology value, and later cluster on a synthesis of the data sets.

6.2 Gene Ontology Based Clustering

Before clustering gene ontology data, a notion of distance needs to be defined between annotated transcripts in order to cluster them. Distances between the RNA transcripts as time-course profiles have been well studied using distance metrics such as Euclidean distance, Pearson correlation between curves, or some model based techniques; but, distance as defined in a gene ontology setting is a fairly new area.

Chapter 3 of the unpublished Ph.D thesis “Incorporating Annotation Data in Quantitative Trait Loci Mapping with mRNA Transcripts” by Christian (2007) addresses GO based distances and I applied some of the techniques from this paper. Two methods to compute the distance between annotated transcripts are based on the “union-intersection” method and on the “longest path” method. This paper developed distance metrics based on those described in Gentleman (2005).

Assume that there are subgraphs $G_1 = \{N_1, E_1\}$ and $G_2 = \{N_2, E_2\}$ both from a larger graph $G = \{N, E\}$, where the subgraphs represent two genes coming from an organism in question (the larger graph.) The distance between $G_1$ and $G_2$ can be
defined in many ways. Following Christian (2007), the Union-Intersection distance between two subgraphs is:

\[ D_{ui}[G_1, G_2] = 1 - \frac{\text{card}(N_1 \cap N_2)}{\text{card}(N_1 \cup N_2)} \]  

where \( \text{card} \) denotes the cardinality, or the number of terms in the set. To illustrate this definition, recall our example from the previous section in Figure 6.1. Three nodes appear in common to both graphs while nine different nodes appear on either graph. Thus, if these two graphs were labeled \( H_1, H_2 \), then \( D_{ui}[H_1, H_2] = 1 - \frac{3}{9} = \frac{2}{3} \). The Union-Intersection distance becomes larger as number of nodes which appear in both graphs decrease.

In equation 6.1, \( D_{ui}[G_1, G_2] = 1 - \text{Sim}_{ui} \) where \( \text{Sim}_{ui} \) is a GO induced distance developed in Gentleman (2005) but with \( D_{ui}[G_1, G_2] \) defined for proper clustering of genes. As it is defined, \( D_{ui}[G_1, G_2] \) has a range of \([0,1]\) where it is 0 when two subgraphs are identical and 1 when two subgraphs have no common term. This definition leads to more intuitive clustering based following the distances matrices from earlier chapters of this thesis based on Pearson correlation distances.

The other distance metric defined by Gentleman (2005) is the longest path distance between two subgraphs. The idea is based on the notion of the longest path (LP), which is the longest path of nodes, in order, among two graphs.
\[
D_{ip}[G_1, G_2] = 1 - \frac{LP(G_1, G_2)}{\max(LP(G, G))}
\] (6.2)

To illustrate this definition, the longest path shared by the two graphs in Figure 6.1 is, from bottom to top, from node ‘all’ to node “GO:0003674”, to node “GO:0005488.” and is of length 2 edges. For illustrative purposes, assume that \( G \) is the union of the two graphs. The longest path of \( G_1 \) is length 3 edges and the longest path of \( G_2 \) is length 5 edges. Hence, \( D_{ip}[G_1, G_2] = 1 - \frac{2}{\max(3, 5)} = 1 - \frac{2}{5} = \frac{3}{5}. \)

In Equation 6.2, \( D_{ip}[G_1, G_2] = 1 - \frac{Sim_{ip}}{\max(LP(G, G))} \) where \( Sim_{ip} \) is defined in Gentleman (2005) as the longest shared path in two subgraphs. Again, as it is defined, \( D_{ui}[G_1, G_2] \) has a range of \([0,1]\) where it is 0 when two subgraphs are identical and 1 when two subgraphs have no common term.

Both of these distances have been constructed such that 0 is the minimum and 1 is the maximum distance between two subgraphs. The advantages of using the longest path distance is that it incorporates the whole graph of the species or genome looked at and references distance in this comparison. Plus, the order these gene products appear in the graphs is important. The disadvantage is that the term on the denominator of Equation 6.2 is often normalized by different factors to ease in computation, especially in large genomes. The union-intersection distance is easier to interpret as it takes just the two subgraphs into account and it is consistent in its use. The union-intersection method is the method I incorporate to find distances
Figure 6.2: Top- Mean Curves for 3 Clusters found by MCLUST for the Human Fibroblast Data; Bottom: Raw Curves corresponding to the 3 Clusters in the top panel

throughout the remainder of this chapter.

6.3 Clustering using GO Data verses Expression Data in the HF Data Set

In Chapter 5, I used model-based clustering techniques SSClust and MCLUST to cluster real data such as the gene expression values of the human fibroblast data set (517 genes and 12 time points.) Here, I compare how well the data sets are clustered by their gene ontology data (ignoring expression value) to clustering based on their expression value (ignoring GO data.)
Figure 6.2 shows the original clusters as well as mean clusters for each of the three clusters found on the expression data for the human fibroblast data from the MCLUST. Of the 517 genes in the data set, 211 genes had GO molecular function (MF) annotation data available; the other 306 elements were marked as NA in the R-software annotation information.) The original analysis on this data set by Iyer et al. (1999) clustered the data (517 genes) based on the non-parametric hierarchical clustering. This clustering found 10 clusters (chosen a priori using biologic properties of the clusters obtained by the expression values) using the average linkage method.

Based on the clustering assignments on expression alone, Iyer et al. (1999) analyzed the genes within each cluster for similar biologic roles and function throughout the cell cycle. They found patterns from 12 subsets of the genes total within the 10 clusters. Some clusters contains subsets with more than one functional group and others contained no subsets based on molecular function. These subsets include functional groups involved in cell cycle and proliferation, coagulation and hemostasis, tissue remodeling, angiogenesis, and cholesterol biosynthesis. This analysis was done after the clustering was performed on the expression data.

6.3.1 Clustering Using Expression Data for the HF Data Subset

For the 211 genes that had gene annotation information available, I needed to re-run my expression analysis for just this subset of human fibroblast data to make proper comparisons. Figure 6.3 shows the clustered expression data for which gene
Figure 6.3: Hierarchical Clustering Assignment for Expression HF Data; n=211 genes, genes for which GO molecular function annotations are available
Figure 6.4: MCLUST Clusters for Expression HF Data; n=211 genes, genes for which GO molecular function annotations are available

ontology information was available using the same method (hierarchical clustering using the average linkage method) as the original paper with 10 clusters. The choice of 10 clusters was somewhat arbitrary based on the original clustering from Iyer et al. (1999).

Parametric clustering using MCLUST on this subset of expression data yielded 5 clusters based on the BIC model selection criteria. These clusters are shown in Figure 6.4. Recall that only 3 clusters were found for the full data set in the previous chapter. Non-Parametric Clustering (K-means) clustering based only on expression data yielded 10 clusters based on the Hartigan Rule in Figure 6.5. Recall that 20 clusters were found using the full data set. The clustering results here on only around
Figure 6.5: K-means Clusters for Expression HF Data; n=211 genes, genes for which GO molecular function annotations are available
Figure 6.6: Density plot of the union-intersection distances for n=211 genes in the HF data set using molecular function annotations.

40% of the expression data are different than on the full data set. The differences are in terms of the cluster structures as well as the number of clusters found. Based on the effect that small cluster sizes has on the number of clusters found in the K-means method from Chapter 5, this clustering of 10 clusters is around what is expected when 60% of the data is missing.
Figure 6.7: K-means Clusters of HF gene expression profiles clustered by Gene Ontology molecular function of n=211 genes

6.3.2 Clustering Using GO Data for the HF Data Subset

As explained above, I used the Union Intersection method to determine the GO distance for my distance matrix. I used this distance metric to obtain the biologic distance between all pairs of curves (genes) for which this information was available (211 genes.) Figure 6.6 shows a density plot of the union-intersection distances between each pair of genes. Terms on the diagonal of the matrix (the distance between a gene and itself is always 0) are omitted. The slight bump in the tail at 0 comes from two genes that have the same GO information. The mean of all of the UI values is 0.79 and the median of all of the values is 0.82. While there is a long tail on just
Figure 6.8: Mean Curves of HF gene expression profiles clustered by Gene Ontology molecular function of n=211 genes

one side due to the truncation at the maximum value of 1, there does not appear to be a transformation which would improve the normality of this data. Thus, I will use the union-intersection distance to create the GO distance matrix to be used in clustering.

I applied K-means clustering to these GO distance matrices (the clustering method which best allows the input of the distance matrix as I have created; again hierarchical clustering would lead to the issue of lack of evaluation for choosing the number of clusters.) This analysis yielded six clusters shown in Figure 6.7. Plots are of the gene expression profiles in these six clusters for elements that were clustered by the GO data. Recall that clustering based on gene expression profiles yielded 10 clusters.
Figure 6.9: Density plots of the union-intersection distances within cluster for the six clusters of HF data using K-means clustering
Mean expression curves of these six clusters are shown in Figure 6.8. Visual inspection of Figures 6.7 and 6.8 do not appear to provide much insight into the expression data as clustered by GO since some the six clusters do not appear to be distinct from each other (like clusters 3 and 6.) A density plot of the union-intersection GO distances for the elements in each cluster is shown in Figure 6.9. The smallest distances, and hence closest related genes by GO, are in cluster 3, a cluster which shows some homogeneity in its expression profiles as seen in Figure 6.7.

Statistically, I can compare the six clustered groups by multivariate analysis of variance techniques to see if in fact there is no significant difference among the six clusters found by GO clustering. Performing a multivariate analysis of variance (MANOVA) over the 12 time points under the GO clustering assignments yields $p < 0.01$. Thus, despite the visual impression, there does appear to be differences between the clusters (clusters 1, 2, 4 appear different from 3 and 6 which appear different from 5.) In a MANOVA performed on both clustering by GO and clustering by expression tested together, both clustering assignments are significant ($p < 0.01$) for the human fibroblast expression data.

Thus, it appears that clustering by each of expression and gene ontology data provide insight into clustering the expression profiles. Later in this chapter, I will examine techniques to combine the expression and the gene ontology data to cluster the data set on this combined information.
6.4 Clustering using GO Data verses Expression Data in the Yeast Data Set

Using the Saccharomyces Genome Database developed by Hong, Balakrishnan, Christie, et al. (2007), I obtained the gene ontology mappings from the biological processes (BP) ontology of the GO database for the yeast data set. Of the 433 curves from the data set I have been using, 35 (8.1%) of the curves do not have annotations mapped to those genes and thus GO analysis can be performed on the 398 curves for which annotations are available.

Thus, in comparison to the human fibroblast analysis in the previous section, this analysis of GO data considers a different ontology (BP verse MF) and has a larger set of curves for which annotations are available.

6.4.1 Clustering Using Expression Data for the Yeast Data Subset

Since this data set is smaller by around 8.1%, the results of $K$-means on the expression data should be similar to that from the last chapter for the full data set. Here, the Hartigan Rule of thumb finds 18 clusters via the $K$-means algorithm, as compared to 19 for the full data set. These clusters are seen in Figure 6.10. These clusters appear quite similar to the clustering of the entire data set as seen in Figure 5.3.

Again, MClust finds nine clusters as shown in Figure 6.11. This clustering appears similar to clustering the full yeast data set as seen in Figure 5.5. Compared to the
Figure 6.10: K-means Clusters for Expression Yeast Data; n=398 genes, genes for which GO biological processes annotations are available.
Figure 6.11: MCLUST Clusters for Expression Yeast Data; n=398 genes, genes for which GO biological processes annotations are available.
Figure 6.12: SSClust Clusters for Expression Yeast Data; n=398 genes, genes for which GO biological processes annotations are available.

results for all of the yeast data from the last chapter, these results for clustering expression on the 91.9% of curves that have annotation data available are quite similar. Next, I will look at clustering on the GO data.

6.4.2 Clustering Using GO Data for the Yeast Data Subset

Again, as with the human fibroblast data, I used the Union Intersection method to determine the GO distance for my distance matrix. I used this distance metric to
Figure 6.13: Density plot of the union-intersection distances for n=398 genes in the yeast data set using biological processes annotations.
Figure 6.14: K-means Clusters of yeast gene expression profiles clustered by Gene Ontology biological processes of n=398 genes

obtain the biologic distance between all pairs of curves for which this information was available (398 genes.) A density plot of these the values in this distance matrix (with diagonal terms omitted) is shown in Figure 6.13. Again, there is a bump in the tail at 0 where pairs of genes are both annotated to the same GO information. The mean of all UI distances is 0.77 and the median is 0.80. Again, as in the case of the HF data set, no transformation is needed to make this data normally distributed and thus I shall use this values in my distance matrix.
Figure 6.15: Mean Curves of yeast gene expression profiles clustered by Gene Ontology biological processes of n=398 genes
Figure 6.16: Density plots of the union-intersection distances within cluster for the nine clusters of yeast data using K-means clustering.
Using the GO distance matrix, K-means using the Hartigan Rule found nine clusters in the expression data as shown in Figure 6.14. Nine clusters is half as many clusters as found when clustered by the distance matrix on expression data. Visual inspection of these clusters for expression data using a GO distance matrix shows better clustering of expression values than when clustering the human fibroblast expression data by a GO distance matrix. This might be due to the different ontology, as this data set was of biologic processes and the human fibroblast data set was of molecular function. This issue of choice of ontology on clustering results will be further examined later in this chapter.

Looking at the density plots of the union-intersection distances within cluster in Figure 6.16, some of the clusters which show similar expression profiles also have the smallest union-intersection distance. Cluster 5 has the smallest union-intersection distances and it appears that the expression profiles and the GO data for these genes are each very similar within this cluster. Clusters 1, 3, 6, and 7, all of which have some appearance of expression clustering in Figure 6.14, also have the smallest union-intersection distances. The relationship between the GO data and the expression data appears to be closer in the yeast data using the biological processes ontology than in the human fibroblast data using molecular function ontology. As I stated above, issues about the choice of ontology will be examined later in this chapter.

I analyzed these clusters via similar MANOVA techniques to those applied to the human fibroblast data. Analyzing just clustering results on the yeast GO data over
the 18 time points yielded $p < 0.01$ and thus the clustering by the GO data set is significant. I also performed a MANOVA on both the GO data cluster assignments and the expression data cluster assignments over the yeast expression data set. Again, the cluster assignments on both methods were significant ($p < 0.01$) for prediction of expression value over the time-course.

Thus, both the expression values and the gene ontology values have a role in clustering gene expression profiles for both the human fibroblast and the yeast data sets. The addition of gene ontology information brings biologic meaning to the expression clustering. In the next section, I examine ways to combine the expression and the GO data in order to best cluster the expression data into biologically meaningful clusters.

### 6.5 Clustering using both GO and Expression Data

A few methods have been proposed to combine gene ontology and gene expression data to obtain good quality, and perhaps improved, expression clustering that has biologically meaningful characteristics. The paper by Pan (2006), while criticizing methods that cluster expression data without taking GO data into account, describes a model which uses GO data as prior information in model-based clustering of the time-course expression data. First, clustering is done on the GO data (with expression profiles where the GO information is unknown lumped into a cluster of genes of unknown function.) Then, the expression data set is stratified by the GO clustering and the expression data set is clustered within each stratum. But, some of the
prior clusters are so small that it makes this clustering difficult to interpret. Also, stratifying by the GO clustering limits the effectiveness of clustering the data by the expression values.

Within some of the model-based clustering techniques, GO data can be used as initial cluster assignments for the expression data being clustered. In MCLUST, I specified the assignments by GO data as the priors in the clustering based on expression data for the human fibroblast and yeast data sets. But, this yielded identical clustering assignments as the situation where clustering was based on expression alone. Thus, these initial values based on the clustering of GO data do not have an impact on the clustering of expression data in MCLUST. Looking at incorporating GO data into clustering expression data by model based techniques is something that I discuss later in this chapter.

One method which I focused my analysis on was first proposed by Boratyn, Datta, and Datta (2007). This is not a new clustering method but just a modification of the distance matrices. This technique takes distance matrices (on the same scale) for the expression data and the GO data and simply adds them together to obtain a “new distance matrix corresponding to distances used by semi-supervised clustering techniques in Machine Learning” (396.) These scaled distance matrices are expected to capture both the differences in expression profiles and in GO information between the genes.

These new distance matrices can then be used as input in clustering techniques
like hierarchical clustering and $K$-means to cluster the expression values. Again, since evaluation techniques exist for the determination of number of clusters in $K$-means, I focused my analysis on this clustering method. To do the scaling of the matrices, I scaled the elements of the expression distance matrix so as this matrix has the same range $[0,1]$ as the GO distance matrix entries and shifted all elements of the expression matrix so that the mean of all the entries in each matrix is the same.

Another method which I used to cluster the expression and the GO data for the yeast data set was using knowledge guided analysis of microarray data as proposed in Fang, Yang, Li, Luo, and Liu (2006). Here, gene ontology is the guide to cluster expression profiles which capture both expression pattern similarities and biological function similarities. In this method, initially a GO clustering tree is constructed from GO data from the biological process ontology. Biological process was chosen as the GO knowledge mapping base because, according to Fang et al. (2006), “experimental studies show that among these three categories of GO [biological process, molecular function, and cellular component], biological process agrees best with the hypothesis that similar expressions indicate a functional relationship” (402.) I will examine the role of the choice of ontology in clustering expression data later in this chapter.

These constructed GO clustering trees have more branches and nodes than than my example annotation DAGs for one terminal node shown in Figure 6.1. But, using the gene on the right side of Figure 6.1 as an example DAG, there are six levels rising vertically from level 0 containing “all” to level 5 containing “GO:0003755,” with two
elements in levels two and three.

For the gene expression matrix, Fang et al. (2006) defined the mean squared residual score (mrsr) to assess the expression correlation of genes within a particular cluster from the clustering by GO. In the definition of the mrsr for the expression matrix \((G,C)\), \(G\) is the set of genes and \(C\) is the set of conditions (which is time in the case of time-course data.) For a subset of genes \(I \subset G\), the mrsr of the submatrix specified by \((I,C)\) is:

\[
H(I,C) = \frac{1}{|I||C|} \sum_{i \in I, j \in C} (a_{ij} - a_{iC} - a_{Ij} + a_{IC})^2
\]  

(6.3)

where \(a_{ij}\) is the corresponding element of the expression matrix, and

\[
a_{iC} = \frac{1}{|C|} \sum_{j \in C} a_{ij},
\]

\[
a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij},
\]

and

\[
a_{IC} = \frac{1}{|I||C|} \sum_{i \in I, j \in C} a_{ij} = \frac{1}{|I|} \sum_{i \in I} a_{iC} = a_{iC} = \frac{1}{|C|} \sum_{j \in C} a_{Ij}.
\]

Here, \(a_{iC}\) is the average expression level of gene \(i\) across all conditions while \(a_{Ij}\) is the average expression of time \(j\) for all genes in \(I\), and \(a_{IC}\) is the average expression of all genes in \(I\) across all time points in \(C\). A low mrsr indicates strong coherence among elements within a cluster. The threshold value \(\delta\) of the mrsr to qualify a cluster for output from this algorithm is one of the user input values in the C++ implementation of this software.

For a given \(\delta\), for every level or generation of the GO tree, if the node \(j\) is clustered
then continue. If the node is not clustered, all descendant nodes (lower levels on the tree) are found and the union of all of these genes corresponding to the descendants of node $j$ are described as gene set $I$. Denote the expression profiles of $I$ as matrix $B$ specifying $(I, C)$ (genes and time-points.) The msrs of $B$ is calculated and if the msrs is below the threshold, then this cluster is output and the algorithm goes on to the next level in the GO tree. If the msrs exceeds the threshold $\delta$, go to the next node in that level of the GO tree. Further details of this clustering algorithm are outlined on page 403 of Fang et al. (2006).

The final step in this method is to filter any outlier expression patterns within a cluster via average trend constraint filtering (atcf). For the cluster output from the clustering algorithm described above, the average trend for each adjacent pair of time-points are calculated producing a trend vector. After a threshold for the maximum tolerable number of inconsistent trends is defined, the trend vector of each gene is compared with the average trend vector. If the difference between the gene's average trend and the average trend vector exceeds the threshold, then the gene is removed from that cluster.

I clustered the expression and GO data from the yeast data set through this algorithm later in this section.
Figure 6.17: K-means Clusters for the human fibroblast data set clustered by both expression data and gene ontology; n=211
Figure 6.18: Mean Curves for the K-Means clusters for the human fibroblast data set clustered by both expression data and gene ontology; n=211
6.5.1 Application to the Human Fibroblast Data Set

I clustered the human fibroblast data set, where GO information was available, by adding the scaled distance matrices of expression and GO data together. The nine $K$-means clusters found for the clustering of this new data matrix are shown in Figure 6.17. Mean curves for these nine clusters are shown in Figure 6.18.

To obtain the scaled distance matrices, I began with the distance matrices for the expression data (using Euclidean Distance) and for the gene ontology data (using the union-intersection method) between all pairs of curves. I then scaled the expression distance matrix to have the same range, [0, 1]) and mean as the GO distance matrix. These two matrices, which are then of equal weight, are then added together to get the combined distance matrix from which the clustering is performed.

The nine clusters found here are more than in clustering this data set using GO data alone (6 clusters as shown in Figure 6.7) and fewer than clustering this data set by expression data alone (10 clusters as shown in Figure 6.5. This clustering in Figure 6.17 appears to yield markedly distinct clusters and includes biologic information. Recall that the six expression clusters found by clustering on the GO data appeared to have some clusters that were similar while the 10 clusters found by clustering expression data appeared well defined but lacked a biologic base from gene ontology.

A multivariate ANOVA analysis on the clustering by adding the GO and expression distance matrices yielded $p < 0.01$ for differences between the clusters and an
analysis of the scaled expression and the GO distance matrices on the combined clustering assignment yielded that both terms were significant. This indicates that both the expression and the GO distance were significant factors in determining final cluster assignment. This is an encouraging result to have a biologically based sensible expression clustering. The average within clusters sums of squares was also lower than it was by clustering by expression data alone.

6.5.2 Application to the Yeast Data Set

I scaled the expression and the GO data distance matrices for the yeast data in the same manner I did for the human fibroblast data. K-means analysis of this combined data matrix yielded 12 clusters as shown in Figure 6.19. Mean curves for these 12 clusters are shown in Figure 6.20. This combined analysis yielded fewer clusters than the 18 found by clustering the expression data alone as shown in Figure 6.10 and more clusters than the 9 clusters found by clustering the expression data using only GO data as shown in Figure 6.14. Visually, these clusters appear to have some of the scale and shape characteristics, including periodicity, of both clustering by expression data alone and clustering by GO data alone. A multivariate ANOVA analysis on the clustering by adding the scaled GO and expression distance matrices yielded \( p < 0.01 \) and both the expression and GO distances were significant in this clustering assignment. The sum of within clusters sums of squares, when penalizing the addition of new clusters when multiplying by the number of clusters \( n \), was lower in clustering
Figure 6.19: K-means Clusters for yeast expression data clustered by both expression data and gene ontology; n=398
Figure 6.20: Mean Curves for the K-Means clusters for yeast expression data clustered by both expression data and gene ontology; n=398
on this combined distance than when clustering by expression alone.

Thus, the results of the clustering of the yeast expression data through the method outlined by Boratyn et al. (2007) confirm the conclusions of the clustering of the human fibroblast data set. The addition of the expression and the gene ontology distance curves, properly scaled, yields a clustering of data curves so as to cluster expression profiles with a biological base for the clustering results.

I also clustered the yeast GO and expression data via a C++ implementation of the clustering method from Fang et al. (2006). Since, for this clustering technique, not every gene is clustered, and some genes are placed into more than one cluster, direct comparisons to one-to-one mapping techniques, like $K$-means clustering on the addition of scaled data matrices, are not applicable. But, clustering from knowledge guided analysis can be used to find particular subsets from the data set which are clustered by biological process and can be used to verify subsets of clustering results from other methods.

The full clustering of the yeast data set implementation of the method of Fang et al. (2006) contained 86 clusters from 11 levels of the GO tree when the algorithm was allowed to run unconstrained. Unlike the clustering obtained from adding distance matrices, some of the expression profiles may not appear in any of the clusters and some expression profiles appear in multiple expression profiles in multiple levels of the GO tree. Raising the threshold mrs value to 0.35 along while requiring each a posteriori cluster to contain at least 10 genes reduces this number of clusters from
Figure 6.21: 13 Clusters of Yeast Data from knowledge guided analysis; msrc threshold value=0.35; Each cluster is labeled in the title by its common Biological Process along with the cluster number, GO tree level, and msrc value.
Figure 6.22: Mean Curves for the 13 clusters of yeast data from knowledge guided analysis; msrs threshold value = 0.35; Each cluster features is labeled in the title by its common Biological Process along with the cluster number, GO tree level, and msrs value.
86 to 13 and these clusters are shown in Figure 6.21.

The labels of the graphs of Figure 6.21 correspond to shared biological process of the genes in this cluster (enrichment of the GO data, for other techniques, is detailed later in this chapter) and these titles also indicate the cluster number from the analysis, the level of the GO tree (lower numbers are at the root of the GO tree and higher numbers are closer to terminal nodes) as well as the mers for that cluster. The largest cluster was "metabolism" with 215 genes while no other cluster had more than 65 genes.

Visual comparison of the clusters from Figure 6.21 and the clusters from Figure 6.19, as well as the corresponding mean clusters from knowledge guided analysis (Figure 6.22) and addition of scaled distance matrices (Figure 6.20), show that the clusters obtained by scaled distance matrices appear to be more homogenous within each cluster. A Wilcoxon rank-sum test comparing the average within cluster sums of squares for the clusters from each clustering method yields $p < 0.01$ with the clusters from this new method having statistically higher within cluster sums of squares (with a one-sided alternative hypothesis.) This result confirms the visual conclusions that the clusters obtained by $K$-means analysis of adding scaled distance matrices yields more homogenous expression clusters than those clusters obtained by knowledge guided analysis.

This clustering method finds detailed subsets of expression data which have similar profiles and share the same biological process. The clustering obtained by adding
scaled distance matrices places equal weight on the GO and the expression information. But, this knowledge guided analysis is designed so that more weight is placed within clusters on similar GO information (biological process in this example) than on the expression profile (where increasing or decreasing trends at each time-point is a filter for cluster assignment.)

The next section looks at this synthesis of information for expression data when some of the corresponding GO information is not available.

6.6 Clustering on Expression and GO Data with Missing Values

Recall, throughout this chapter, when analyzing the human fibroblast data set, I have been using the subset of data for which GO information is available, nearly 40% of the data. But, these clustering results ignore the over 60% of expression data for which no GO information was available. This is a common issue in gene ontology analysis, in which different amounts of data are available for different annotations of genes (only 7% of the annotations were missing in the yeast data set from this chapter.) My goal in this section is to use all possible data to get the best classification of genes using both expression and gene ontology data.

In the case of the human fibroblast data set, the full data set featured 517 genes for which gene expression data are available. I used various clustering techniques
including $K$-Means clustering. In chapter 5, Figure 5.7 showed the 20 clusters found via $K$-Means clustering using the Hartigan stopping rule for the expression data for the 517 genes.

Earlier in this section, Figure 6.17 showed the $K$-Means clustering result on the data set of the expression data added to the gene ontology data for the 211 genes for which the gene ontology information was available. While the nine clusters appear distinct, this clustering result is somewhat unsatisfying since I ignored the 306 genes for which the GO data were unavailable.

**Algorithm 5** Inferring expression and GO clusters with missing GO data

Cluster the entire expression data set using the distance matrix based on expression data (denoted $K1$ containing $k1$ clusters.)

Cluster the subset of expression data for which GO information is available on the distance matrix of the addition of scaled expression and GO data matrices (denoted $K2$ containing $k2$ clusters.)

For each cluster from $K1$, enumerate the cluster assignment (from 1 to $k2$) when that information is available, from $K2$ for each element of each cluster of $K1$.

if For a cluster in $K1$, a single cluster from $K2$ corresponds to 75% or more of the data set for which GO information is available that cluster in $K1$ then

Let that cluster in $K1$ equal the corresponding cluster value (number) from $K2$.

if Two more or more clusters have the same $K2$ value then

Combine those elements into one large cluster containing the union of all elements in those clusters.

end if

else if For those clusters that do not meet the 75% threshold, the two most prevalent values from $K2$ are the same for two clusters and combine for 75% or more of the data set for which GO information is available in each cluster then

Combine these clusters into one larger cluster.

else

Keep the remaining clusters separate.

end if

Hence, I developed and applied an inferential technique to find the optimal average
clustering to take into account all 517 genes and the GO information and this is shown in Algorithm 5. If for two or more clusters of the expression values of the full data set, the data for which GO are available are clustered into the same "Expression+GO" cluster 75% of the time or more, then these clusters are combined into one large cluster (interpolating the missing GO data as coming from this cluster.) Clusters can also be combined if instead the "Expression+GO" cluster assignments come from the same pair of clusters 75% or more of the time. The choice of 75% as a cutoff value was somewhat arbitrary and can be adjusted for various clustering situations. Lowering the threshold to 60% could increase the number of clusters that combine hence decreasing the number of clusters found by the algorithm. I use the more conservative value of 75% since some of my simulations will remove a large portion of the GO data and leave expression clusters with few elements having GO data. Some of this sparse data might be incorrectly combined at a lower threshold value.

Applying this technique to the $K$-means analysis of the human fibroblast data set, 14 of the 20 clusters from $K1$ came from a single cluster in $K2$ 75% of the time or more, including 5 pairs of clusters where each pair is combined into one cluster. The remaining six clusters each remained their own separate cluster. Thus, this synthesis of information yielded 15 clusters, 5 of which are pairs of clusters from the original clustering for which GO information prompted their addition. These new clusters, which take into account expression and GO data for the entire data set can be seen in Figure 6.23.
Figure 6.23: K-means Clusters for HF Data: Using all available data (expression and GO) on the full expression data set (517 genes); Using interpolation for GO data when that data information is missing.
Looking at the clusters, these 15 clusters retain much of the clustering identity from the clustering by the synthesis of expression and GO data as seen in Figure 6.17. But, now I can cluster the entire data set of 517 genes as opposed to just a subset of 211 genes. Comparing this new clustering of the entire expression data set to the clustering of the entire data set by expression alone in Figure 5.7, the clustering by expression alone appeared to have some very similar clusters. But, with some GO information added, the clustering not only appears improved but is more biologically viable.

Under this clustering measure, the "worst case scenario" is that no cluster reaches the 75% threshold and thus all of the original clusters do not have a corresponding cluster from "GO+Expression" for it to be matched with. In this case, the clustering on expression alone is the "optimal" clustering for the entire data set of expression and GO data. Under the "best case scenario," every cluster has a corresponding cluster from the clustering of "GO+Expression" for it to be matched to. In this case, the "optimal" clustering consists of clusters which are just the synthesis of clusters from the clustering by expression value alone (with like-clusters combined) with the number of clusters less than or equal to the number of clusters by clustering by "GO+Expression" for the subset of data.

I compared the accuracy of my predictions based on these clusters from the method of Algorithm 5 by running this algorithm on the full data set with no GO data removed (n=398.) To test this method, I will use the clustering of the synthesis of the
expression and the GO data, as shown in Figure 6.19 as the "truth" in the clustering algorithm. I lowered the threshold of Algorithm 5 so as to force this algorithm to find 12 clusters, which is the number of clusters for the data set treated as "truth." Although this might increase the misclassification in the clustering, this was the best way to compare to the truth by using the same number of clusters. I also clustered the expression data alone into 12 clusters and compared this clustering to the "truth" as this clustering is used as the baseline for the improvement of clustering using Algorithm 5. The misclassification rate decreased from 16%, when the expression data were clustered and compared to "truth" to 6% when this new clustering algorithm's clustering was compared to truth. Thus, 63% of the curves that were misclassified, when 12 clusters were found in K-means clustering, in absence of GO information, were clustered correctly by Algorithm 5.

To examine the quality and the stability of the clustering technique from Algorithm 5, I used resampling (without replacement) of my yeast data set to test this method for accuracy. I removed a portion (a quarter, half or three-quarters) of the GO data and attempted to cluster the remaining data (398 expression profiles for which only 398k have GO data available for k = 0.25, 0.5, 0.75) using the inferential method of Algorithm 5 to cluster all 398 profiles on the expression and GO data. The clustering was done using K-means where the "true" number of clusters is 12. I compared these predictions to the actual clustering assignments as shown in Figure 6.19 to see how the algorithm performs as the fraction of expression data for which
GO information is available decreases.

Again, I lower the threshold of Algorithm 5 to force 12 clusters to be found so as to compare clustering via misclassification rates. For each value of \( k \), I resampled 100 times and clustered these results. The average misclassification rate was 7.8\% for \( k = 0.75 \), 11.3\% for \( k = 0.5 \) and 14.4\% for \( k = 0.25 \). Thus, as the amount of GO information that was removed increases, the average misclassification rate nears 16\%, which is misclassification rate when no GO data is used. But, there was an increase in clustering accuracy even when the most information was removed and thus the algorithm appears to perform well. Also, since 12 clusters were found for comparison purposes, increasing the threshold to make combining clusters more difficult might avoid combining some clusters that are less similar, whereby reducing misclassification.

6.7 Enrichment of GO Data

While visual comparison and MANOVA tests and p-values can tell whether there is an improvement in clustering expression data based on expression and GO data verses clustering expression data on GO data alone, that alone does not say that the clustering has some improved biological impact. One way to see if the clustering using both expression and ontology data did provide some biological improvement is to look at the enrichment of the gene ontology data for the elements of the clusters. Enrichment of gene ontology data can be used to examine the differences in molecular
functions between groups of clustered expression data.

The enrichment information is available in the R software package for the human fibroblast data set. Recall Figure 6.17 showed nine clusters obtained using K-means when the distance matrices from expression data and ontology data were scaled and added together before clustering. For each gene, annotation information is available including the gene ontology tags based on Locus Link IDs (maintained by the NCBI) for each element of the data set. These were based on the functional class of annotations. Using the 'ontotools' library package, developed in Gentleman (2005) within the R software environment, the most informative tags for each element of each cluster are found. Using these tags, the most informative GO nodes for each cluster are available. These most informative tags can be used to obtain directed graphs for all nodes in a group when taken together to find common roots structures which may differ for different clusters. An example of such a graph for one of the nine clusters from Figure 6.17 is shown in Figure 6.24. These DAGs can be used to find the structural tree of the differing molecular functions of clusters.

Another way to use this data for enrichment purposes is to analyze the distribution of various functional groups within cluster as they compare to the overall data set. Note that it is possible for more than one gene in a cluster to have the same most informative tag. Also note that some genes do not have a most informative tag and some genes have a most informative tag for which the function is unknown. Thus, the total sample size for all of the subset categories is less than or equal to the number
Figure 6.24: Directed Acyclic Graph (DAG) for informative GO nodes of one of the nine clusters found in the K-means analysis of synthesis of GO and expression data
Table 6.1: Molecular Function classification for the most informative GO tags obtained from genes within each of the nine clusters obtained by clustering by K-Means by the synthesis of expression and the GO data of genes in the data set.

Table 6.1 shows the molecular function classification, broken up by Binding, Enzyme Activity, Receptor Activity, and Transporter or Other Activity for the most informative GO tags for genes in each cluster. The apportionment of functions within and across clusters appears to show a difference between clusters based on these functional classes. Some clusters, like clusters three and eight contain Enzyme Activity genes for more than half of the genes in the cluster. Binding, which makes up just a small portion of clusters three and eight, is more prominently featured in clusters one, four, five, and seven. Receptor Activity, absent in some clusters like four and eight, make up a decent portion of clusters seven and nine.

A Fisher Exact Test executed on the elements of Table 6.1 is marginally significant ($p = 0.10$) and thus indicates that the enrichment of the GO information could trend
towards clustering with different molecular function classifications. The addition of the enrichment of the GO data adds an extra layer of confirmation to the results obtained by the synthesis of expression and GO data in comparison with clustering by expression data alone. Enrichment of the yeast data will be shown in the next chapter in our full comparison of yeast clustering outcomes.

6.8 The Effect of the Choice of Ontology on Clustering Expression Data

According to Fang et al. (2006), "experimental studies show that among these three categories of GO [biological process, molecular function, and cellular component], biological process agrees best with the hypothesis that similar expressions indicate a functional relationship" (402). To examine the effect of gene ontology on clustering, I used GO annotations in the molecular function annotation and the cellular component annotation for the same genes from the yeast expression data for which I have been using biological process annotation. I compared clustering results using distance matrices obtained from molecular function, cellular component, and biological process annotations to see if indeed the choice of ontology has an effect on the clustering results of expression data (with the hypothesis from the above quotation that biological process leads to the most highly correlated curves within clusters of the three ontologies.)
Figure 6.25: K-means Clusters of yeast gene expression profiles clustered by Gene Ontology biological processes of n=398 genes.
Figure 6.26: Mean Curves of yeast gene expression profiles clustered by Gene Ontology biological processes of n=398 genes.
Recall that clustering via the $K$-means algorithm on biological process GO data from the yeast data set showed nine clusters as displayed again in Figure 6.25 with mean curves displayed in Figure 6.26. These clusters appeared to show better clustering than clustering molecular function GO data from the human fibroblast data set. The statement from Fang et al. (2006) indicates that this is due to the choice of ontology.

Figure 6.27 shows the nine clusters found by $K$-means analysis for clustering the yeast expression data by a GO matrix of molecular function GO data. Mean cluster profiles are shown in Figure 6.28. Visual comparison indicates that these clusters do not appear as distinct from one another as those found using biological processes. Figure 6.29 shows the nine clusters found by $K$-means analysis for clustering the yeast expression data by a GO matrix of cellular component GO data. Mean cluster profiles are shown in Figure 6.30. Visual comparison again shows that these clusters do not appear as distinct as those found using biological processes. The choice of nine clusters in clustering expression using each of the three GO data sets came from the Hartigan Rule, which found nine clusters when clustering using biological processes data (and between eight and 10 in the other ontologies) and this will be used to compare between ontology.

Table 6.2 shows the sum of the within cluster sums of squares for repeated runs of the $K$-means algorithm for clustering the yeast expression data by a GO distance matrix for the three different ontologies. These distances are normalized on a $[0,1]$
Figure 6.27: K-means Clusters of yeast gene expression profiles clustered by Gene Ontology molecular function of n=398 genes.
Figure 6.28: Mean Curves of yeast gene expression profiles clustered by Gene Ontology molecular function of n=398 genes
Figure 6.29: K-means Clusters of yeast gene expression profiles clustered by Gene Ontology cellular component of n=398 genes
Figure 6.30: Mean Curves of yeast gene expression profiles clustered by Gene Ontology cellular component of n=398 genes
Table 6.2: Summary statistics for the sum of the within cluster sums of squares for K-means analysis of clustering yeast expression data by a GO distance matrix for the three choices of the GO ontologies: n=398 clustered in each analysis; 100 runs of the clustering algorithm for each ontology

<table>
<thead>
<tr>
<th>GO Ontology</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Confidence Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>360.0</td>
<td>12.6</td>
<td>(357.5, 362.5)</td>
</tr>
<tr>
<td>MF</td>
<td>504.5</td>
<td>4.9</td>
<td>(503.5, 505.5)</td>
</tr>
<tr>
<td>CC</td>
<td>826.1</td>
<td>63.1</td>
<td>(813.7, 838.5)</td>
</tr>
</tbody>
</table>

scale and the sum of the within clusters sums of squares is found as described in Chapter 2. As the distributions of the sums of the WCSS appear to be normally distributed within the runs in each ontology, a two-sample t-test shows that clustering by Biological Processes has a significantly lower sum of WCSS than clustering by either molecular function or cellular component. Also, clustering by molecular function has a significantly lower sum of WCSS than cellular component.

These results confirm the statement from Fang et al. (2006) as well as my clustering on GO data earlier in this chapter. Hence, indeed it appears that clustering expression data by biological processes yields the most highly correlated expression profiles within the each cluster among the three ontologies. Thus, the results I showed in this chapter for clustering yeast data using biological processes was best among all choices for the ontology. Hence, the clustering by the addition of scaled distance matrices for expression and GO shown earlier in this chapter in 6.19 were indeed optimal over the choice of ontology.
But, not all data sets are annotated in all three ontologies and, for instance in the case of the human fibroblast data, the choice of ontology that was used provided the most GO data available for that data set. But, when the choice does exist, it appears that biological processes is the best choice for ontology to obtain clustering results from gene expression data.

One more question that needs to be answered, when multiple GO ontologies are available as in the yeast data, is whether the clustering should be done using the preferred ontology like biological processes or using a synthesis of the GO data from all three ontologies. I compared the clustering results of the yeast expression data using the BP ontology to clustering using a synthesis of ontologies using two different synthesizes.

The first way to use all GO data is to use a combined distance matrix using all of the GO information. This leads to larger DAGs with more nodes in each pair of graphs using the union-intersection distance. Using this new distance matrix, I clustered the yeast data using $K$-means with nine clusters and ran the algorithm 100 times. The mean of the sum of the within cluster sums of squares for these runs is 600.8 with a standard deviation of 25.1. Thus, in comparison with the sum of the within cluster sums of squares using just biological processes (row 1 in Table 6.2,) the clustering using all GO annotation data has a significantly higher sum of WCSS. The clustering does not improve using a distance matrix of all GO data as perhaps these larger GO graphs lead to less meaningful union-intersection values.
Another way to combine all of the GO information is to find the three separate
distance matrices from the three ontologies and create a new distance matrix by
adding the three matrices together and multiplying the resulting matrix by the scalar
\( \frac{1}{3} \). K-means clustering is then done on this new distance matrix where each entry is
the average of the corresponding entries in the three matrices for the three different
ontologies. Under 100 runs of the algorithm, the mean of the sum of the WCSS is 585
with a standard deviation of 5.9. Again, in comparison with the sum of the WCSS for
the clustering using biological processes, this clustering using all GO annotation data
has a significantly higher sum of WCSS. Thus, neither method of using a synthesis
of all GO data appears to provide an improvement over clustering expression values
on biological processes alone when that annotation is available.

6.9 Chapter Summary

This chapter introduces gene ontology data and methods to incorporate GO data
into clustering gene expression data. Clustering expression values by expression data
alone, as was done in all previous chapters, clusters expression data but ignores gene
ontology. GO data yields biologically based clustering and when this information is
combined with the expression data, provides clustering in which both GO and ex-
pression data are significant. This was seen in the analysis of adding scaled distance
matrices and clustering via the K-means algorithm as well from knowledge guided
analysis. The choice of ontology and its role in clustering was examined and it was
shown that clustering by biological process GO data was best among the three ontologies.

Enrichment of GO data and the quality of the clustering in terms of, for the human fibroblast data, the molecular functions of the gene products of the elements of each cluster were examined. The results were marginally significant for clustering assignment based on functional class. If this data had been annotated instead in the biological process annotation or if the molecular function annotation data were not missing for 60% of the genes for which the expression data were available, perhaps the results would have been significant.

To analyze the synthesis of expression and GO data when some of the genes are missing GO data, I introduced a classification technique to use all of the data available while using inference to account for the missing data. I then examined this technique on the yeast data set and showed that this method, while fairly conservative, is viable and provides added information than techniques which just ignore the genes for which no GO information is available or treat all the missing data as a large miscellaneous cluster.

Most of the results incorporating gene ontology data in this chapter were clustered using non-parametric techniques. Ease of use implementing expression and ontology data together led to the use of \( K \)-means clustering even though it was not the clustering algorithm recommended from my simulations on expression data in chapters 4 and 5. Further work on this project could involve finding a way to incorporate gene
ontology data as covariate data in the model-based clustering techniques like SSClust and MCLUST so that these methods cluster on both expression and GO data in a meaningful way.
Chapter 7

Conclusions, Recommendations, and Discussion

7.1 Review of Methods and Conclusions

In this thesis, I first introduced the issues surrounding clustering time-course array data as well as some examples of these data sets. Next, I described and detailed some popular non-model based and then model-based clustering methods while introducing some evaluation techniques of clustering methods including the misclassification rate and the overall success rate of a clustering method. In Chapter 4, I applied some of these clustering techniques to various simulated data sets designed to mimic popular time-course gene expression patterns (in shape and amplitude.) These simulations showed that the model-based MCLUST method proved to be the best in simulations to find the correct number of clusters and give proper cluster assignments. This was in contrast to the simulation study performed in Ma et al. (2006) which showed SSClust as the method which clustered that particular simulated data best. I described some of the flaws with that particular data set and the need for more simulations. This chapter also showed the consistency of clustering methods to changes in the number of data curves and the number of true clusters in the data sets for the model-based clustering techniques.

I then applied these non-model-based and model-based clustering methods to two
real gene expression data sets in Chapter 5. I also treated these results as truth to
create simulated data sets based on observed variations of time-course data. These
data sets differed from those in the prior chapter in that they were based on real data
and had smaller variance between clusters. In these simulations, the model-based
SSClust method proved to be the best method as it had the highest Overall Success
Rate in these simulations. Overall, MCLUST performed best in the situation where
the variance between clusters was large, as in the simulated data sets from Chapter
4, but if the difference between clusters is expected to be small as in the real data
sets from Chapter 5, I would recommend using SSClust as the clustering method of
choice.

These results in Chapters 1-5 were all based on gene-expression data alone. In
Chapter 6, I introduced gene ontology and how that data can be incorporated with
the gene expression data to get clustering that has some biological significance. Clus-
tering expression values alone without looking at the ontology can lead to biologically
nonsensical clustering. The conclusions from this chapter are that the methods that
introduce GO data into expression clustering yield satisfactory, and in some cases,
Improved clustering than that of expression alone.

To reach these conclusions, I examined methods that cluster genes by both ex-
pression and ontology information and applied these methods to data sets for which
expression and GO data was available. One of these methods included adding scaled
distance matrices and the other was knowledge guided analysis. Of these methods,
knowledge guided analysis provides more weight to the GO information to get clusters which share a common biological process while adding scaled distance matrices yields clustering with smaller within cluster sums of squares. The choice of which clustering method to use would depend on how important a common biological process within the clusters was in comparison to the necessity for a low variance between the expression values within the cluster.

In these methods that used scaled expression and GO data matrices, I clustered using the $K$-means algorithm although the same techniques can be done using hierarchical clustering. $K$-means was used in order to use validation methods for determining the number of clusters since there is no commonly accepted termination criterion for hierarchical clustering, according to Zaiane, Foss, Lee, and Wang (2002), without a priori examining each data set. The model-based methods I examine in this thesis all have validation techniques to terminate their algorithm and $K$-means has the Hartigan Rule, of which I examined the consistency via bootstrap techniques in Chapter 5.

The choice of which ontology to use to cluster the expression data, when data from multiple ontologies are available for the same genes, is also examined. Based on the results of clustering the yeast data, biological process is the choice of ontology which provides clusters with the higher correlation between GO data and expression data Fang et al. (2006). Chapter 6 also examined the clustering results by looking at the enrichment of the GO classes within clusters.
Also introduced in Chapter 6 was a classification technique to infer clustering on missing GO data and this method can be used to cluster expression data using both expression and GO data when some of the genes are missing GO annotations. The technique I used in Chapter 6 allows more of the data to be used without casting aside the expression data for which no GO information is available into a miscellaneous cluster as is done in Pan (2006). The gene ontology project and its website (www.geneontology.org) are constantly updated with new and changing ontologies for various genomes and my methods from Chapter 6 are sufficiently flexible to analyze new and updated data sets.

The clustering methods I use in this thesis are general and are not limited by the size of the data set or the number of clusters, with computing power exponentially higher than when Rand (1971) developed criteria for clustering based on both accuracy and computational speed. Computational speed was not an issue for the methods and results in this thesis and thus accuracy trumped computational speed. These clustering methods can be used to analyze new gene expression array data sets in future research endeavors. Also, with knowledge in the field of gene ontology growing every day, there is no shortage of new data that will emerge on which the clustering techniques I examined, recommended, or developed, can be used and used properly to get optimal clustering results for both expression and GO data.

The remainder of this thesis will discuss conclusions and recommendations based on my research and will put my work in perspective of the larger field of clustering
<table>
<thead>
<tr>
<th>Clustering Algorithm</th>
<th>Number of Clusters</th>
<th>SSD</th>
<th>Penalized SSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical Clustering</td>
<td>5</td>
<td>1827</td>
<td>4085</td>
</tr>
<tr>
<td>K-Means (Expression)</td>
<td>19</td>
<td>921</td>
<td>4015</td>
</tr>
<tr>
<td>SSClust</td>
<td>10</td>
<td>1171</td>
<td>3703</td>
</tr>
<tr>
<td>MCLUST</td>
<td>9</td>
<td>1311</td>
<td>3933</td>
</tr>
</tbody>
</table>

Table 7.1: The number of clusters, the sum of squared distances of each element to its cluster centroid, and the penalized sum of squared distances multiplied by the square root of the number of clusters, \( n \); Yeast Data

gene expression data.

7.2 Recommendations

Let's return to the question of which of our clustering method is "best" to cluster time-course gene expression array data. This section will include a discussion what exactly is meant by "best." In the context of the methods and the data used in this thesis, let's look as to how best to cluster the yeast data set.

First, let's compare the clustering results of the yeast expression profiles from Chapter 5 using MCLUST, SSClust, and K-means. Figures 5.5, 5.4, and 5.3 show the optimal clustering results for these clustering algorithms for the yeast data set as we determined from the results of that chapter. Table 5.3 showed the clustering performance of these clustering algorithms in terms of stability, in terms of comparing misclassification rates due to small perturbations in the data set. These earlier results showed that SSClust had more stable clustering results than MCLUST or K-means in terms of a higher overall success rate at all levels of noise.
Now, let's compare the quality of clustering between methods by comparing the optimal clustering results from each method we found earlier by comparing the sum of squared deviations for all curves from their cluster's centroid from which that curve is assigned. Table 5.1 displays the number of clusters found for the yeast data set. To compare clustering methods where the number of clusters found is different for each method, a penalty terms on adding more clusters needs to be included to make proper comparisons. One alternative would have been to compare average squared deviations in each clusters. Instead, a similar result is achieved by penalizing the addition of more clusters by multiplying by the square root of n, the number of clusters. This method is motivated by a discussion of clustering techniques from Aldenderfer and Blashfield (1984). Table 7.1 shows these results and these results again show that SSClust has the best clustering, in terms of lowest penalized sum of squared distances within cluster.

Now, in addition to looking at clustering the expression profiles, let's look at the inclusion of gene ontology information into the clustering as I did in Chapter 6. Table 6.2 showed that if gene ontology is to be used for the yeast data set, then biological processes is the annotation which is to be used since it had the smallest sum of within cluster sums of squares. Figure 6.19 shows the K-means clustering of expression profiles by a combined distance metric using both expression and ontology information and we showed that both expression and ontology information influenced cluster assignment. The sum of the within cluster sums of squares for the 12 clusters
<table>
<thead>
<tr>
<th>Biological Process</th>
<th>Number of Genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell Cycle</td>
<td>41</td>
</tr>
<tr>
<td>Development</td>
<td>59</td>
</tr>
<tr>
<td>Metabolism</td>
<td>56</td>
</tr>
<tr>
<td>Regulation</td>
<td>45</td>
</tr>
<tr>
<td>Stimuli</td>
<td>49</td>
</tr>
<tr>
<td>Transport</td>
<td>42</td>
</tr>
<tr>
<td>Biogenesis</td>
<td>34</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>72</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>398</strong></td>
</tr>
</tbody>
</table>

Table 7.2: Biological Process classification for the most informative GO tags obtained from genes from the yeast data set; n=398 genes

is 1198 and the penalized sum of squared distance is 4145.

Thus, when compared to the results of Table 7.1, the addition of GO information in $K$-means clustering does not provide better sum of squared distance clustering than $K$-means clustering without GO or the best model-based method, SSClust. $K$-means with GO gives an 11.9% increase in the penalized sum of squared distances over SSClust.

Let’s next examine the biological classifications of elements in each of these clustering techniques. To classify the biological processes of the clusters in Figure 6.19, I determined cluster enrichment for each approach (expression verses expression plus GO.) This is comparing the amount of biological information in the best clustering for expression data in $K$-means when expression profiles are clustered without GO information and using GO information in order to see if clustering using GO information actually provides better biological groupings in the clustering assignments.
Using the same methods outlined in Chapter 6 in the section on "Enrichment of GO Data", which I performed on the human fibroblast data, I analyzed the yeast data set. I found the most informative GO tags as described in Chapter 6 for each gene and then was able to analyze the biological processes associated with these GO tags for each element of each cluster. I split all of the elements into one of eight biological process classifications as suggested by Gustin (2007). These classifications include gene tags associated with the cell cycle, the development node, metabolism, regulation, detection of and response to stimuli (including signaling pathways), transport and transporter activity, biogenesis, and miscellaneous classifications including genes only mapped to the biological process root node. Table 7.2 shows the apportionment of BP classifications among 398 nodes mapped to GO information in the yeast data set. This analysis of the enrichment of GO data in this section is based on a counting method but I could have alternatively used a hypergeometric test for enrichment of GO data as described in Alexa, Rahnenfuhrer, and Lengauer (2006).

To examine how well $K$-means clustered based on these biological classifications, I will classify each cluster by it's most prevalent biological process and the percent of genes in that cluster which had that process as indicated by the most informative GO tag. For those clusters that had ties for two or more biological processes most frequently related to a cluster's most informative tags, ties were broken by the total number of genes classified to a process using all gene tags and not just the most informative from "ontotools" Carey (2003).
<table>
<thead>
<tr>
<th>Biological Process</th>
<th>Frequency (%)</th>
<th>Cluster Size</th>
<th>Cluster Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell Cycle</td>
<td>40</td>
<td>20</td>
<td>3</td>
</tr>
<tr>
<td>Cell Cycle</td>
<td>31</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>Cell Cycle</td>
<td>29</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>Biogenesis</td>
<td>57</td>
<td>7</td>
<td>17</td>
</tr>
<tr>
<td>Biogenesis</td>
<td>33</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>Development</td>
<td>41</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>Metabolism</td>
<td>24</td>
<td>42</td>
<td>7</td>
</tr>
<tr>
<td>Metabolism</td>
<td>21</td>
<td>39</td>
<td>13</td>
</tr>
<tr>
<td>Stimuli</td>
<td>29</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>Stimuli</td>
<td>25</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>Stimuli</td>
<td>22</td>
<td>32</td>
<td>8</td>
</tr>
<tr>
<td>Regulation</td>
<td>33</td>
<td>9</td>
<td>16</td>
</tr>
<tr>
<td>Regulation</td>
<td>22</td>
<td>46</td>
<td>11</td>
</tr>
<tr>
<td>Transporter</td>
<td>29</td>
<td>14</td>
<td>18</td>
</tr>
<tr>
<td>Transporter</td>
<td>25</td>
<td>20</td>
<td>6</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>31</td>
<td>36</td>
<td>5</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>26</td>
<td>27</td>
<td>9</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>26</td>
<td>31</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 7.3: Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in K-means where clustering is performed by expression data alone; 18 clusters found for n=398 genes.
<table>
<thead>
<tr>
<th>Biological Process</th>
<th>Frequency (%)</th>
<th>Cluster Size</th>
<th>Cluster Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metabolism</td>
<td>29</td>
<td>34</td>
<td>2</td>
</tr>
<tr>
<td>Metabolism</td>
<td>23</td>
<td>61</td>
<td>1</td>
</tr>
<tr>
<td>Stimuli</td>
<td>36</td>
<td>22</td>
<td>3</td>
</tr>
<tr>
<td>Stimuli</td>
<td>22</td>
<td>65</td>
<td>11</td>
</tr>
<tr>
<td>Transport</td>
<td>31</td>
<td>55</td>
<td>12</td>
</tr>
<tr>
<td>Development</td>
<td>50</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>Development</td>
<td>50</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td>Cell Cycle</td>
<td>24</td>
<td>25</td>
<td>5</td>
</tr>
<tr>
<td>Biogenesis</td>
<td>87</td>
<td>15</td>
<td>6</td>
</tr>
<tr>
<td>Regulation</td>
<td>30</td>
<td>40</td>
<td>7</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>100</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>94</td>
<td>50</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 7.4: Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in K-means where clustering is performed by a synthesis of added scaled distance matrices for expression data and GO data; 12 clusters found for n=398 genes.

Table 7.3 shows the most prevalent biological process corresponding to the 18 clusters of yeast expression data, where clustering is done by expression data alone. These clusters are shown in Figure 6.10. Some clusters which visually appear similar in terms of expression profile also have the same most prevalent biological process. These include clusters 11 and 16 which most correspond to “Regulation.” The frequency in Table 7.3 refers to the percentage of curves in that clusters which are mapped to the most prevalent biological process as indicated in that column. The cluster size is the number of curves in that cluster. Overall, these frequencies in Table 7.3 are low and overall, only 107 out of 398, or 26.8%, of genes correspond to the most prevalent biological process in that cluster.

Table 7.4 shows the most prevalent biological process corresponding to the 12
clusters of yeast expression data, where clustering is done by a synthesis of scaled distance matrices of expression and GO. These clusters are shown in Figure 6.19. Recall how clustering expression profiles by GO data alone yielded poor clustering in Figure 6.14 but that this addition of scaled distance matrices yielded markedly improved clustering. Overall, 160 out of 398, or 40.2%, of genes correspond to the most prevalent biological process in that cluster. Thus, when GO information is used, there is a 50% increase in the number of genes which are matched to the most prevalent biological process in the cluster. Hence, when the $K$-means clustering algorithm is used, clustering gene expression profiles by a synthesis of scaled expression data and GO data is preferred when compared to clustering by expression data alone.

Finally, let's compare the the best expression clustering results using MCLUST and SSClust for the same 398 genes which we analyzed above. Clustering these expression profiles using these clustering methods is shown in Figure 6.11 and Figure 6.12.

Tables 7.5 and 7.6 show the most prevalent biological processes associated with the clusters when expression data is clustered using MCLUST and SSClust. Using MCLUST, 101 out of 398 genes (25.3%) correspond to the most prevalent process in their cluster while 116 out of 398 genes (29.1%) are mapped using SSClust. Recall that 26.8% were mapped using $K$-means clustered by expression clustering alone. Thus, when clustering by expression data alone, SSClust has the lowest penalized sums of squared distances as shown in Table 7.1 and the highest percentage of genes
<table>
<thead>
<tr>
<th>Biological Process</th>
<th>Frequency (%)</th>
<th>Cluster Size</th>
<th>Cluster Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell Cycle</td>
<td>35</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>Cell Cycle</td>
<td>21</td>
<td>29</td>
<td>1</td>
</tr>
<tr>
<td>Stimuli</td>
<td>28</td>
<td>53</td>
<td>8</td>
</tr>
<tr>
<td>Stimuli</td>
<td>22</td>
<td>46</td>
<td>5</td>
</tr>
<tr>
<td>Transporter</td>
<td>19</td>
<td>104</td>
<td>4</td>
</tr>
<tr>
<td>Regulation</td>
<td>29</td>
<td>52</td>
<td>6</td>
</tr>
<tr>
<td>Development</td>
<td>60</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>Metabolism</td>
<td>24</td>
<td>58</td>
<td>9</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>31</td>
<td>26</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 7.5: Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in MCLUST, where genes are clustering using expression data only; 9 clusters found for n=398 genes

<table>
<thead>
<tr>
<th>Biological Process</th>
<th>Frequency (%)</th>
<th>Cluster Size</th>
<th>Cluster Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>42</td>
<td>32</td>
<td>1</td>
</tr>
<tr>
<td>Development</td>
<td>31</td>
<td>52</td>
<td>7</td>
</tr>
<tr>
<td>Transporter</td>
<td>18</td>
<td>28</td>
<td>2</td>
</tr>
<tr>
<td>Cell Cycle</td>
<td>21</td>
<td>36</td>
<td>9</td>
</tr>
<tr>
<td>Cell Cycle</td>
<td>18</td>
<td>25</td>
<td>4</td>
</tr>
<tr>
<td>Regulation</td>
<td>27</td>
<td>57</td>
<td>5</td>
</tr>
<tr>
<td>Metabolism</td>
<td>20</td>
<td>36</td>
<td>6</td>
</tr>
<tr>
<td>Metabolism</td>
<td>19</td>
<td>44</td>
<td>8</td>
</tr>
<tr>
<td>Stimuli</td>
<td>33</td>
<td>28</td>
<td>10</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>33</td>
<td>60</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 7.6: Most prevalent biological process for the most informative GO tags obtained from genes from the yeast data set for each cluster in SSCLust, where genes are clustering using expression data only; 10 clusters found for n=398 genes
which map to the biological process most closely associated with that cluster. Hence, when expression data alone is to be clustered with no GO information, SSClust is the recommended clustering method.

7.2.1 Extensions

These procedures above can be used to analyze any time course gene expression data set. As stated before, the methods that incorporated gene ontology data in this chapter were clustered using non-parametric techniques. Further work on this project could involve finding a way to incorporate gene ontology data as covariate data in the model-based clustering techniques like SSClust and MCLUST so that these methods cluster on both expression and GO data in a meaningful way. This would help complete the comparison above over all clustering methods whether or not GO data was used to see if the incorporation of GO data into the model-based clustering techniques of SSClust and MCLUST outperform all other methods in terms of biological groupings and similarity of expression profiles within the clusters. Such an analysis might finally yield a definitive answer for the “best” clustering method.

7.3 Status of this Field

With the rise in availability of gene-expression array data and gene ontology annotations, the number of problems concerning clustering these data sets has risen and this field has seen an influx of papers in recent years. Some recent papers have used Ma
et al. (2006) as a method from which to compare recent methods or ideas in clustering. Wang, Chen, and Li (2007) looked to not only cluster by gene-expression but to also analyze gene regulation. Sahoo, Dill, Tibshirani, and Plevritis (2007) presents a new method to analyze genes which feature abrupt transitions in expression levels over time. These are different ways of looking at the same problem of putting similarly behaving genes together by comparing aspects of their expression profiles.

Some papers have continually advocated for the inclusion of gene ontology data when analyzing gene expression data. Wang, Azuaje, and Bodenreider (2005) concludes that a clustering technique which integrates gene ontology data into the expression clustering “not only produces consistent results, but also it offers alternative, potentially meaningful views of the biological problem under study.” These papers reinforce the ideas of Pan (2006) that clustering methods which ignore gene ontology information are necessarily incomplete.

But, not everyone in this field is sold on the notion that including gene ontology information is necessary when that information is available. Ng, McLachlan, Wang, Ben-Tovim Jones, and Ng (2006) believes that there is still room to analyze newer expression clustering techniques on the ever changing types of array data available without necessarily including gene ontology information. They write that “this is because present [gene ontology] databases are necessarily incomplete and evolving.”

Recall that Boratyn et al. (2007) proposed a clustering method whose data matrix is the addition of scaled expression and ontology matrices. This is one of the few ways
which gives equal weight to the expression and ontology information. Clustering is then done on this combined data matrix based on the addition of the two scaled data matrices. Other methods like Pan (2006), which partitions by ontology data before clustering by expression, and Fang et al. (2006), which forms clusters by ontology information before comparing expression similarity within clusters, place a higher weight on the ontology data than the expression data.

Because of the level of uncertainty that some, like Ng et al. (2006) perceive about gene ontology information and its evolving databases, a clustering method derived from the method Boratyn et al. (2007) might be a most efficient way to combine ontology and expression information in clustering expression profiles giving less weight to the gene ontology information. This method is explained in the next section.

### 7.3.1 Additional Extensions and Concluding Remarks

For those researchers who want to include ontology data but give more weight to expression data, due to the uncertainty surrounding incomplete and evolving ontology data sets, a modification can be made to the method of Boratyn et al. (2007) which could be to cluster based on a weighted average of these data matrices (denoted $DM$ below.)

$$
DM_{combined} = \alpha DM_{expression} + (1 - \alpha) DM_{ontology} \quad (7.1)
$$
Here, $\alpha = 0.5$ would yield the results of equal weighting of data matrices as described in Boratyn et al. (2007). $\alpha = 1$ would indicate a complete disregard for the ontology data and completely cluster on expression profiles. This would be similar to clustering done in Chapters 4 and 5 of this thesis. $\alpha = 0$ would indicate a complete disregard for the expression data and cluster only on ontology data. The level of $\alpha$ used could be up to the research team based on their confidence in the use of gene ontology data. This is an interesting application that could be looked at as an extension of this project in terms analyzing what value of $\alpha$ would yield the best clustering based on the evaluation techniques I used in the previous sections of this chapter.

Overall, the debate and the interest remains strong in the field of clustering time-course gene expression array data and there appears to be no slowdown in sight with the increasing availability, popularity, and importance of these data sets. The work in this thesis provides perspective in comparing differing methods with different clustering procedures and different amounts of biological information used.
Bibliography


A. Raftery, K. Yeung, and C. Fraley. Model-Based Clustering and Data Transforma-


