Developing and Application of Statistical Algorithms for High-Dimensional Biological Data Analysis

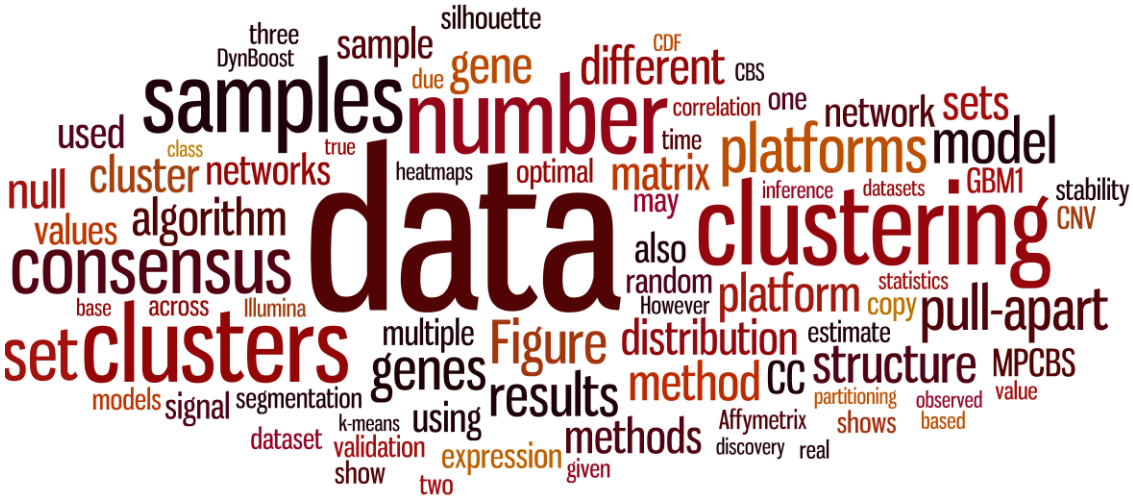
by

Yasin Şenbabaoğlu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Bioinformatics)
in The University of Michigan
2012

Doctoral Committee:

Assistant Professor Jun Li, Co-chair
Professor George Michailidis, Co-chair
Professor Daniel M Burns Jr.
Assistant Professor Maureen Sartor
Professor Florence d'Alché-Buc, Université d'Evry - France



The most frequently used words in this thesis.

The size of each word is proportional to its frequency.

To my family and friends,

ACKNOWLEDGEMENTS

I owe my deepest gratitude to my supervisors Jun Li, George Michailidis and Florence d'Alché-Buc, whose guidance and support from the initial to the final level were invaluable for the completion of this thesis. I have acquired important amounts of knowledge and wisdom from each one of them, which helped me grow both as a person and a researcher. I am honored to have worked with them and hope to continue collaborations in the future. I am also grateful to my committee members Dan Burns and Maureen Sartor, whose input and feedback for the projects were very important.

I would like to show particular thanks to Florence d'Alché-Buc, who accepted to host me for two summers at the Université D'Evry in France. When I typed 'machine learning in systems biology' in google and sent her an email to ask for a possible collaboration opportunity in 2009, I did not know that I would be welcome by such a warm and generous personality and have the privilege to work in the excellent research atmosphere of IBISC. This experience developed a strong appreciation in me for the theory of machine learning, intellectually enriched me and substantially broadened my research perspective.

I am indebted to my collaborators Nancy Zhang, assistant professor of statistics at Stanford University; Bo Li, PhD student in Bioinformatics supervised by Jun Li; and Nehemy Lim, PhD student in Systems Biology and Machine Learning supervised by Florence d'Alché-Buc. They each brought their prowess to the project we were collaborating on, and I learned a substantial amount from them.

It is a pleasure to thank Dan Burns once more, and Margit Burmeister, as co-directors of the Bioinformatics program. Together, they generously nurtured students through the program with their student-friendly attitude, guidance and support. I am also very thankful to the program staff, who were the other key element in making Bioinformatics a nurturing and student-friendly program. I would like to show my particular thanks to student services representative Julia Eussen and media consultant Alex Terzian.

This thesis would not have been possible without the support of my family and friends. Since the day my parents saw me leave for education in the United States on August 13, 2001, they have provided continual support and encouragement even if it meant suppressing the emotional pain caused by separation. I cannot thank them enough. I am also grateful to my dear friends whom I met in Ann Arbor but who may now be in different parts of the world. I particularly shared a lot with, and learned many things from my special friend and roommate Ajdin Kavara, who became one of the important figures in my life. I also would like to show particular thanks to Li lab members Bo Li, Weiping Peng, and Valerie Schaibley for their continued support. The conversations we have had in the more challenging parts of graduate school were invaluable to help me weather the hardships.

Lastly, I offer my thanks, regards, and blessings to all of those who supported me in any respect during the completion of my thesis but whose names I did not have space to mention here.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

Developing and Application of Statistical Algorithms for High-Dimensional Biological Data Analysis

by

Yasin Şenbabaoğlu

Co-chairs: Jun Li and George Michailidis

Various high-throughput technologies have fueled advances in biomedical research in the last decade. Two typical examples are gene expression and genomic hybridization microarrays that quantify RNA and DNA levels respectively. High-dimensional data sets generated by these technologies presented novel opportunities to discover relationships not only among interrogating probes (i.e genes) but also among interrogated specimens (i.e samples). At the same time, however, the necessity to model the variability within and between different high-throughput platforms has created novel statistical challenges. In this thesis, I address the opportunities and challenges with three algorithms.

First, I present DynBoost, a new method to infer gene-gene dependence relationships and nonlinear dynamics in gene regulatory networks. DynBoost is a flexible boosting algorithm that shares features from L_2-boosting and randomization-based algorithms to perform the tasks of parameter learning and network inference. The performance of the proposed algorithm was evaluated on a number of benchmark data sets from the DREAM3 challenge and the results strongly indicated that it

outperformed existing approaches.

Second, I revisit consensus clustering (CC) and some other clustering methods in the context of unsupervised sample subtype discovery. I show that many unsupervised partitioning methods are able to divide homogeneous data into pre-specified numbers of clusters, and CC is able to show apparent stability of such chance partitioning of random data. I conclude that CC is a powerful tool for minimizing false negatives in the presence of genuine structure, but can lead to false positives in the exploratory phase of many studies if the implementation and inference are not carried out with caution in line with particular prudent practices.

Lastly, I present MPCBS, a new method that integrates DNA copy number analysis across different platforms by pooling statistical evidence during segmentation. I show by comparing the integrated analysis of Affymetrix and Illumina SNP array data with Agilent and fosmid clone end-sequencing results on 8 HapMap samples that MPCBS achieves improved spatial resolution, detection power, and provides a natural consensus across platforms.

CHAPTER I

Introduction

The advent of high-throughput experimental technologies in the last decade initiated a data-rich era for biology. The ability to gain quantitative information from all genes, mRNA transcripts, peptide products, and metabolites in whole systems resulted in a rapid increase in the number of large-scale datasets. However, the interpretation of these large-scale biological datasets created important challenges for investigators in deriving information about whole systems as the data originated from complex stochastic systems and had both experimental and technical sources of error. The interdisciplinary research efforts by biologists, statisticians, computer scientists, and mathematicians tackling these challenges led to important advances in both the statistical modeling of the sources of error and also the biomedical context from which the data were collected.

Two typical examples of high-throughput biological datasets are gene expression and comparative genomic hybridization microarrays that quantify the messenger RNA (mRNA) and DNA copy number levels in the genome respectively.

(a) **Gene expression microarrays:** This platform involves the use of nucleic acids, referred to as *probes*, attached to a solid surface to measure via hybridization the quantity of complementary nucleic acid transcripts present in a sample, referred to as the *target* (1). A greater number of target molecules that hybridize to the

probes emit light at a higher intensity, which is then detected by a specialized scanner. The expression levels for the genes in the system can be obtained from these intensity levels after several technical numerical steps such as background adjustment, normalization, and summarization.

Depending on the research question, data on the expression levels of genes can be collected either from varying experimental conditions at a fixed time point for a static representation of the system, or from different time points under the same experimental condition to observe the dynamic properties. In the former case, the dataset can be represented by an $N \times P$ matrix, where N and P denote the number of samples and the number of probes respectively. Data matrices from different experimental conditions can be compared to infer the differences in the gene expression levels at each condition. In the latter case, however, a given sample is assayed for T time points, resulting in a $T \times P$ data matrix for each sample. Ideally, multiple samples should be assayed at the same time points to obtain a collection of $T \times P$ matrices and reduce the effects of the technical and biological sources of error. Such a collection of *time series* datasets can then be used to interrogate the dynamic properties of the system such as regulatory relationships among genes.

High-dimensional datasets generated by gene expression arrays present novel opportunities to discover relationships both among interrogating probes (*i.e.* genes) and among interrogated samples.

(1) **Relationships among probes:** This case carries the important promise of deciphering system-wide interactions that regulate the inner workings of a cell. Regulation in the cell can occur at multiple levels including but not limited to gene regulatory interactions, protein-protein interactions and signal-transduction interactions. The interacting agents in these regulatory systems

are thought to constitute a network model with edges representing interactions. This is a rather idealistic abstraction in a Platonic sense as also pointed out by (33).

> "... (A) network model ... is a concept rather than an observation, or a measurement related to an observation (e.g., gene expression or fluorescence intensity). In the same vein as the equations of mathematical-physics do not "hide" within the physical phenomena they describe, the networks we proposed as gold standard models ... do not truly "exist" in nature. Networks are a conceptual representation of cellular interactions."

In Chapter 2, I focus on one particular regulatory regime in the cell, namely gene regulatory networks (GRN), or also known as transcript control networks. The interacting agents in these networks are genes, RNA transcripts, and translated proteins. Proteins that bind to the promoter sequence of a gene, or to other proteins bound to this sequence are called 'transcription factors'. Transcription factors, either alone or through forming complexes with other proteins, can alter the expression levels of genes' mRNA transcripts. In the network model, this interaction is modeled with a directed edge from the gene that has coded for the transcription factor to the gene whose expression level is altered. In addition to the directionality, the edges are also signed; with a positive sign indicating increased expression levels (activation) and a negative sign indicating decreased expression levels (suppression). This first type of GRN reverse-engineering algorithm, termed "gene-to-sequence" regulation has led the way to the so-called "physical modeling" approach for developing GRN reverse-engineering algorithms (6). This approach seeks to specifically model the binding of transcription factors to promoter sequences, and hence differs from the second broad category that seeks to model "gene-to-gene" interactions.

The second broad category of GRN reverse-engineering algorithms follows the

'influence modeling' approach where the goal is to relate the expression of a gene to the expression of other genes in the cell (gene-to-gene interaction) (6). Even though interactions actually take place between gene products such as mRNA transcripts or proteins, the gene regulatory network model does not use separate nodes for gene products, but only a single node to represent them together with the relevant coding gene. As in the physical interaction model mentioned above, regulations can either 'activate' (positive interaction) or 'suppress' (negative interaction) expression levels.

A plethora of methods have been developed to infer the topology and dynamics of GRNs from both static and time series datasets. In spite of this rich collection of methods, certain types of errors continue to challenge the modeling efforts, meaning that there is still significant room for improvement (9; 31). In particular, an important open question is related to the development of efficient methods to infer the underlying gene regulation networks from temporal gene expression profiles (11). Modeling temporal data has a crucial advantage over modeling spatial data in that temporal interactions are able to lend the model the causality interpretation without any leap of faith or extra assumptions. Spatial data, on the other hand, are best suited to infer statistical or probabilistic dependencies between nodes. Causality can be inferred from spatial data under some strong assumptions that are not known to be true for biological systems (2).

The challenge of gene regulatory network inference can be addressed by leveraging the power of ensemble methods in machine learning. For instance, boosting is an important supervised machine learning method that aggregates multiple weak learners to arrive at a better prediction than the base learners alone. However, the development of boosting algorithms for structure learning has so far been restricted to spatial data. Not being able to

leverage the valuable information from temporal data has limited the success of these methods.

In Chapter 2 of this thesis, I present DynBoost, a new method that achieves reverse-engineering of GRNs based on the 'influence modeling' approach mentioned above by employing time series datasets and nonlinear dynamics in an efficient boosting algorithm. This work is submitted to the 2012 meeting for the *European Conference in Computational Biology*, where the accepted papers will be published in *Oxford Bioinformatics*. For this project, I co-implemented the DynBoost algorithm and performance tests, generated and analyzed the results, and drafted the paper.

(2) **Relationships among samples:** The case regarding the relationships among samples falls squarely in the context of unsupervised class discovery. This line of research has been used effectively in the last decade to discover disease subtypes from gene expression datasets. Hierarchical clustering gained rapid popularity in the first half of the last decade particularly due to producing easy-to-interpret results and not requiring user-specified parameters. However, stability issues in the implementation of this method made the results and the discovered classes questionable. In one example, breast cancer was first reported to have six subtypes in a 2001 report that so far received 4157 citations (2). Then, in 2003 a group with the same leading authors published a report leaving out one of the six previously-declared subtypes (3) and this report has been cited 2477 times. Yet, in 2006 a more sophisticated analysis by a different group, but still relying on hierarchical clustering, contested the robustness of the two of the remaining five subtypes (4). Although this latter group published extensively on both breast cancer subtypes and biology, the number of citations for this report remains at far smaller levels at 14.

Consensus clustering (CC) (11) was developed in a condition of unsupervised

class discovery when such stability and robustness issues ailed the reported classes. The consensus rate between two samples measures how frequently they are grouped together in multiple clustering runs under a certain degree of perturbation. The base clustering method for the multiple runs can be any partitioning method, including the previously popular hierarchical clustering. CC rapidly gained popularity over other existing algorithms in the context of unsupervised class discovery due to a number of reasons. First, it was not a new partitioning method *per se*, it allowed the investigators to use any partitioning method they were already familiar with. Second, CC allowed for a visualization of the robustness of the clusters as well as an inference for the optimal number of classes. CC's promise was to present a visual tool for the stability and robustness of the clusters, however the potential of the multiple runs to exaggerate the inherent biases of the base partitioning method did not receive much attention. Moreover, the significance levels associated with the inferences, and the sensitivity/specificity of this method under null conditions of differing strength have not been systemically studied.

In Chapter 3 of this thesis, I revisit consensus clustering (CC), motivate the over-interpretation potential of this method with a real biological dataset, further analyze the sensitivity and specificity results in simulated datasets of 'known structure' or 'known lack of structure', and recommend prudent practices for the implementation of CC. This work is in preparation for submission to *PLOS Computational Biology*. For this project, I re-coded the CC algorithm along with other clustering methods such as CLEST and GAP; implemented these and some other existing methods such as the silhouette width. I performed tests, generated and analyzed the results, and drafted the paper.

(b) **Array-comparative genomic hybridization (aCGH):** Similar to gene ex-

pression microarrays, this technique also relies on the hybridization of *probe* and *target* nucleic acid sequences. However, the sequences in aCGH platforms are DNA fragments as opposed to the RNA molecules on gene expression arrays. The probes on a typical aCGH measurement interrogate the DNA copy numbers of total genomic DNA isolates from differentially labeled test and reference samples. There are multiple aCGH platforms such as Illumina, Affymetrix, and Agilent, which differ from one another in various components of the measurement procedure including the type of probe used, namely copy number variation (CNV) and single nucleotide polymorphism (SNP) probes, sample amplification procedures and the data format generated. For instance, Agilent arrays produce two-color ratio data in a test/reference format, while Affymetrix and Illumina arrays measure each sample independently. Due to the different techniques employed, these three methods produce data values with distinct noise characteristics, different proportions of low-quality SNPs and distinct local signal trends.

The multiple dimensions of variability among these different platforms created a novel statistical challenge for integrating DNA copy number calls from *multiple platforms*. In contrast with the *multiple-time-points* and *multiple-sample* settings above where the probes and samples can be studied to understand interactions or similarities between one another, the *multiple-platform* setting in this thesis involves data from interrogating a single sample on different platforms at only one time point. Collecting DNA copy number data from multiple time points is likely to be a fruitless approach because DNA copy number gains and losses occur at extremely low frequency when compared with the changes in the levels of gene expression transcripts. The multiple-platform approach can, however, be extended to the multiple-sample case where biological replicates of the same system under study are collected and each replicate is assayed in multiple platforms. However, the common situation today is to be able to find data from multiple

samples on a single platform or from a single sample on multiple platforms. Since the multiple-sample case for DNA copy number analysis was studied elsewhere (13) and is beyond the scope of this thesis, I present below the motivation for the joint estimation of DNA copy number from multiple platforms.

(1) DNA copy number variation (CNV) is an important source of human genetic variation. As much as 12 % of the human genome (housing thousands of genes) is variable in copy number, which is likely to constitute a significant proportion of normal phenotypic variation (14). Moreover, CNVs are known to be associated with specific gene functions and human diseases as well. Thus, detecting and cataloguing CNVs at higher resolutions is both genetically and medically important.

In recent years, a growing number of genetic studies have relied on collecting genome-wide data on DNA copy number variants. The conventional approach for analyzing these datasets is to apply one of the CNV detection algorithms to search for genomic intervals of altered signal intensity. When multiple technical platforms (or different versions of the same platform) are used to interrogate the same biological samples, the conventional approach would involve running the same CNV detection algorithm separately on the data from each platform, and then combining these segmentation results with a simple intersection or union operation. Regardless of the choice of the operation, this approach for integrated copy number analysis lacks a theoretical foundation and presents difficulties in making a consensus decision when platforms disagree on the calling of a CNV. What platform to have more confidence in, and how much weight the calls from each platform should have are questions that are not answered by separate runs of the CNV detection algorithm.

For example, Affymetrix, Agilent and Illumina arrays are three platforms that fundamentally represent three distinct marker panels and different molecular

assay methods. Thus, they report different magnitudes, different boundaries and different degrees of uncertainty for the same underlying copy number event. An integrated analysis, where information from all platforms are used at the same time to detect CNVs and to estimate the levels of change, is expected to maximize resolution and accuracy. The difference of this integrated analyis from the conventional approach is that, in the conventional case, statistical evidence for a given copy number event is pooled *after* segmentation. However, in the integrated analysis, statistical evidence for each copy number event is pooled *during* segmentation and the CNV calls made are the consensus calls that do not require further combination.

In Chapter 4 of this thesis, I present MPCBS for such an integrated estimation of DNA copy number from multiple platforms. This new method is based on a simple multi-platform change-point model where the built-in statistic automatically pools the statistical evidence for copy number events during segmentation. This work was published in *Oxford Bioinformatics* (15). For this project, I debugged the MPCBS implementation and increased its efficiency. I performed tests, generated and analyzed the results.

Bibliography

[1] Huber, W., Irizarry, R.A., Gentleman, R. (2005) Preprocessing overview. In: Gentleman R, Carey VJ, Huber W, Irizarry RA, Dudoit S, editors. Bioinformatics and Computational Biology Solutions Using R and Bioconductor. New York: Springer p. 4.

[2] Sorlie, T., Perou, C.M., Tibshirani, R., Aas, T., Geisler, S., Johnsen, H., et al. (2001) Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proc Natl Acad Sci* USA;98:1086974.

[3] Sorlie, T., Tibshirani, R., Parker, J., Hastie, T., Marron, J.S., Nobel, A., et al. (2003) Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proc Natl Acad Sci* USA;100:841823.

[4] Alexe, G., Dalgin, G.S., Ramaswamy, R., DeLisi, C., Bhanot, G. (2006) Data perturbation independent diagnosis and validation of breast cancer subtypes using clustering and patterns. *Cancer Informatics*; 2: 24374.

[33] Stolovitzky, G., Prill, R. J. and Califano, A. (2009) Lessons from the DREAM2 Challenges. *Annals of the New York Academy of Sciences*, 1158: 159195. doi: 10.1111/j.1749-6632.2009.04497.x

[6] Gardner, T.S. and Faith, J. (2005) Reverse-engineering transcription control networks. *Physics of Life Reviews* 2, 65-88.

[32] Stolovitzky G, Monroe D, Califano A (2007) Dialogue on reverse-engineering assessment and methods: The dream of high-throughput pathway inference. *Annals of the New York Academy of Sciences* 1115:122.

[9] Marbach D, Prill RJ, Schaffter T, Mattiussi C, Floreano D, Stolovitzky G. (2010) Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286-6291

[2] Bansal M, Belcastro V, Ambesi-Impiombato A, di Bernardo D (2007) How to infer gene networks from expression profiles. *Molecular systems biology* 3:78.

[31] Smet,R.D. and Marchal,K. (2010) Advantages and limitations of current network inference methods. *Nat. Rev. Microbiol.*, 8, 717729.

[11] Zoppoli P, Morganella S, Ceccarelli M (2010) TimeDelay-ARACNE: Reverse engineering of gene networks from time-course data by an information theoretic approach. *BMC Bioinformatics*, 11:154.

[11] Monti S, et al. (2003) Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. Machine Learning. 52:91-118.

[13] Zhang, N.R., Siegmund, D.O., Ji, H., and Li, J. (2010). Detecting simultaneous change-points in multiple sequences. *Biometrika*, 97, 631-645.

[14] Carter, N. (2007) Methods and strategies for analyzing copy number variation using DNA microarrays. *Nat. Genet.* 39:S16S21.

[15] Zhang, N.R., Senbabaoglu, Y., Li, J.Z. (2009) Joint estimation of DNA copy number from multiple platforms. *Bioinformatics* 26(2):153-160.

CHAPTER II

A novel boosting algorithm to infer nonlinear dynamics and interactions in gene regulatory networks

2.1 Abstract

Motivation: Reverse engineering of gene regulatory networks remains a central challenge in computational systems biology, despite recent computational advances. A number of approaches using either perturbation (knock-out) or wild-type time series data have appeared in the literature addressing this problem. Nonlinear dynamical models are particularly appropriate for this inference task employing time series data. Recent benchmark *in-silico* challenges have accelerated the progress on this front with numerous advances in learning the parameters of such models; however, these modeling efforts have not enjoyed the same level of success for identifying the network structure with high fidelity. In this study, we introduce a novel nonlinear autoregressive model based on operator-valued kernels that simultaneously learns the model parameters, as well as the network structure.

Results: A flexible boosting algorithm (DynBoost) that shares features from L_2-boosting and randomization-based algorithms is developed to perform the tasks of

parameter learning and network inference for the proposed model. Specifically, at each boosting iteration, a regularized operator-valued kernel based vector autoregressive model (OKVAR) is trained on a random subnetwork. The final model consists of an ensemble of such models. The performance of the proposed algorithm is evaluated on a number of benchmark data sets from the DREAM3 challenge and the results strongly indicate that it outperforms existing approaches.

2.2 Introduction

The ability to reconstruct cellular networks plays an important role in our understanding of how genes interact with each other and how this information flow coordinates gene regulation and expression in the cell. Gene regulatory networks (GRN) have the potential to provide us with the cellular context of all genes of interest, as well as with a means to identify specific subnetworks that are malfunctioning in a given disease state (6; 14). A diverse suite of mathematical tools has been developed and used to infer gene regulatory interactions from spatial and temporal high-throughput gene expression data (see 2; 16 and references therein). A fair comparison for the relative merits of these methods requires their evaluation on benchmark datasets, which the DREAM (Dialogue for Reverse Engineering Assessments and Methods) project (33) provided. This project aims to enhance the strengths and understand the limitations of various algorithms to reconstruct cellular networks from high-throughput data (32). In addition to the choice of the algorithm, network reconstruction heavily depends on the input data type employed. Data that measure the response of the cell to perturbations -either by knocking out or silencing genes- are particularly useful for such reconstructions since they offer the potential to obtain a detailed view of cellular functions. The downside is that obtaining large-scale perturbation data is expensive and relatively few methods have been proposed in the literature to infer regulatory networks from such data due to computational challenges (34; 12). Data

from time-course gene expression experiments have the potential to reveal regulatory interactions as they are induced over time. A number of methods have been employed for this task; including dynamic Bayesian networks (35; 19), Granger causality models (20; 21; 29), and state-space models (23; 26). The first set of methods are computationally very demanding, while the latter two employ linear dynamics, hence limiting their appeal. Other approaches are based on assumptions about the parametric nature of the dynamical model and resort to time-consuming evolutionary algorithms to learn the network (30). Moreover, in spite of the rich collection of methods employed to solve the topology and dynamics of GRNs, certain types of errors continue to challenge the modeling efforts, meaning that there is still significant room for improvement (9; 31). In this study, we introduce a framework for network inference based on nonlinear autoregressive models, which are particularly appropriate when little is known about the true underlying dynamics. Our framework is rich and flexible, and it can efficiently capture the complex dynamics of the network as well as its underlying structure.

2.3 System and Methods

2.3.1 Non linear autoregressive models and network inference

Let $\mathbf{x_t} \in \mathbb{R}^{\mathbf{p}}$ denote the *observed* state of a GRN comprising of p genes. Further, let S be the set of these p genes. We assume that a first-order stationary model is adequate to capture the temporal evolution of the network state, which can exhibit nonlinear dynamics captured by a function $H : \mathbb{R}^p \to \mathbb{R}^p$; i.e. $\mathbf{x}_{t+1} = H(\mathbf{x}_t) + \mathbf{u}_t$, where $\mathbf{u}_t$ is a noise term. The regulatory interactions amongst the genes is captured by an adjacency matrix A, which is the target of our inference procedure.

Note that for a linearly evolving network, A can be directly estimated from the data. In a nonlinear setting, a direct estimation is not possible but averaging the

values of the empirical Jacobian matrix of the function H over the whole set of time points provides an estimate. Specifically, denote by $\mathbf{x}_0, \ldots, \mathbf{x}_{N-1}$ the observed time series of the network state. Then, $\forall (i,j) \in \{1, \ldots, p\}^2$, an estimate of the adjacency matrix A is given by:

$$\hat{A}_{ij} = g\left(\sum_{t=0}^{N-1} \frac{\partial H(\mathbf{x}_t)_i}{\partial (\mathbf{x}_t)_j}\right), \tag{2.1}$$

where g is a smooth thresholding function. Note that in the presence of sufficient number of time points ($N >> p$) one can use the above posited model directly to obtain an estimate of A, provided that a good functional form of H is selected. However, the presence of more genes than time points makes the problem more challenging, which, together with the absence of an obvious candidate functional form for H, make a *nonparametric* approach an attractive option.

Such an approach is greatly facilitated by adopting an ensemble methodology where H is built as a linear combination of nonlinear autoregressive *base* models defined over subsets of genes (e.g. subnetworks). Let M be the number of subnetworks and $\mathcal{S}_m \subset \mathcal{S}$ $(m = 1, \ldots, M)$ be the subset of genes that constitute the m^{th} subnetwork. Each subset has the same size k. We assume that H can be written as a linear combination of M autoregressive functions of the type $h : \mathbb{R}^p \to \mathbb{R}^p$ such that:

$$\widehat{\mathbf{x}}_{t+1} = H(\mathbf{x}_t) = \sum_{m=1}^{M} \rho_m h(\mathbf{x}_t; \mathcal{S}_m) \tag{2.2}$$

The paramater set $\mathcal{S}_m$ defines the subspace of $\mathbb{R}^p$ where h operates. This componentwise subnetwork approach is intended to overcome the intractability of searching in high-dimensional spaces and to facilitate model estimation. In our framework, subnetworks do not have any specific biological meaning and are allowed to overlap.

Efficient ways to build an ensemble of models include bagging, boosting and randomization-based methods such as random forests (7; 3). The latter two approaches have been empirically shown to perform very well in classification and re-

gression problems. In this study, we employ an L_2-boosting type algorithm suitable for regression problems (8; 5) enhanced with a randomization component where we select a subnetwork at each iteration. The algorithm sequentially builds a set of predictive models by fitting at each iteration the residuals of the previous predictive model. Early-stopping rules developed to avoid overfitting improve the performance of this algorithm.

Next, we discuss a novel class of base models.

2.3.2 A new base model

The ensemble learner is a linear combination of M base models denoted by h (Eq. 2.2). Even though h works on a subspace of $\mathbb{R}^p$ defined by $\mathcal{S}_m$, for sake of simplicity we present here a base model $h : \mathbb{R}^p \to \mathbb{R}^p$ that works with the whole set of genes, e.g. in the whole space $\mathbb{R}^p$. Here, we introduce a novel family of nonparametric vector autoregressive models that are based on matrix-valued functions. The model we propose is inspired by the framework of reproducing operator-valued kernel Hilbert spaces, which are appropriate for vector-valued function approximation. We abbreviate this model with OKVAR, using the uppercase letters in Operator-valued-Kernel-based Vector AutoRegressive models. Similar models have recently become popular in machine learning for multitask problems (18) and structured classification.

In a similar fashion as the scalar case, it is also possible to use matrix-valued kernels for kernel ridge regression, elastic modeling and support vector regression. The properties for an operator-valued kernel[1] can be found in Micchelli and Pontil (2005) (18). In this work, we propose a kernel that does not satisfy all the properties of an operator-valued kernel, i.e. it is not generally semidefinite positive, but it allows to take into account each coordinate pair of two vectors of $\mathbb{R}^p$, as shown next.

[1] As output space is $\mathbb{R}^p$, the operator is a linear application and thus a matrix

Let $\mathbf{x}_0, \ldots, \mathbf{x}_{N-1}$ be the observed network states. Then, model h is defined as

$$h(\mathbf{x}_t; \mathcal{S}) = \sum_{k=0}^{N-1} K(\mathbf{x}_k, \mathbf{x}_t).\mathbf{c}_k \tag{2.3}$$

where $K(\cdot, \cdot)$ is an operator-valued kernel and each $\mathbf{c_k}$ ($k \in \{0, \ldots, N-1\}$) is a vector of dimension p. In the following, we denote the matrix of the N row vectors $\mathbf{c_k^T}$ of dimension p with C ($C \in \mathcal{M}^{N,p}$).

In this work, we define the following matrix-valued extension of the Gaussian kernel: $\forall(\mathbf{x}, \mathbf{z}) \in \mathbb{R}^{\mathbf{2p}}$,

$$K(\mathbf{x}, \mathbf{z})_{ij} = \exp\left(-\gamma_{ij}(x_i - z_j)^2\right). \tag{2.4}$$

Here, K depends on a matrix hyperparameter $\Gamma := \gamma_{\{1 \leq i,j \leq p\}}$. For any given pair of states $(\mathbf{x}, \mathbf{z})$, each coefficient $K(\mathbf{x}, \mathbf{z})_{ij}$ measures how close coordinate i of state $\mathbf{x}$ and coordinate j of state $\mathbf{z}$ are. When γ_{ij} is very high, the proximity of those coordinates is not taken into account in the model. One great advantage of such a kernel is that, contrary to a scalar Gaussian kernel, it allows the comparison of all coordinate pairs of the two network states and does not reduce them to a single number.

Matrices Γ and C need to be learned from the available training data. If K is fixed, C can be estimated using penalized least square minimization as in (4). However, learning Γ and C simultaneously is more challenging, since it involves a non-convex optimization problem. Thus, we propose to decouple the learning of hyperparameter Γ and parameter C by first estimating Γ and then using this estimate to learn C. Noting that X, the matrix of the N row vectors $\{\mathbf{x}_0^T, ..., \mathbf{x}_{N-1}^T\}$ is in $\mathcal{M}^{N,p}$, we define Γ using a symmetric matrix $W := w_{\{1 \leq i,j \leq p\}}$ such that w_{ij} codes for an independence measure between genes i and j:

$$\gamma_{ij} = \frac{\alpha \exp(-\beta w_{ij})}{\frac{1}{N^2 p^2} \sum_{k,\ell,i,j} (x_{ki} - x_{\ell j})^2} \tag{2.5}$$

where $\alpha, \beta \in \mathbb{R}$ are user-specified hyperparameters. In this setting, learning Γ reduces to estimating W, which is an easier task that can be addressed with a statistical independence test such as the Hilbert-Schmidt Independence Criterion (HSIC) (11).

2.4 Algorithm

2.4.1 DynBoost

The proposed algorithm is called *DynBoost*, since H models the temporal evolution between network states $\mathbf{x_t}$ with an L_2-boosting approach. As seen in Algorithm 1, it generates $H_m(\mathbf{x}_t)$, an estimate of $\mathbf{x}_{t+1}$ at iteration m, and updates this estimate in a while-loop until an early-stopping criterion is met, or until the prespecified maximum number of iterations M is reached.

Algorithm 1 DynBoost

Inputs :

- Network states : $\mathbf{x}_0, \ldots, \mathbf{x}_{N-1} \in \mathbb{R}^p$
- Early-stopping threshold ϵ

Initialization :

- $\forall t \in \{0, \ldots, N-1\}$, $H_0(\mathbf{x}_t) := (\bar{\mathbf{x}}^1, \ldots, \bar{\mathbf{x}}^p)^T$
- Iteration $m = 0$, STOP=false

while $m < M$ and STOP=false **do**

Step 0: Update $m \leftarrow m + 1$

Step 1: Compute the residuals $\mathbf{u}_{t+1}^{(m)} := \mathbf{x}_{t+1} - H_{m-1}(\mathbf{x}_t)$

Step 2: STOP := true if $\forall j \in \{1, \ldots, p\}$, $\|\mathbf{u}^{j(m)}\| \leq \epsilon$

if STOP=false **then**

Step 3: Select $\mathcal{S}_m$, a random subset of genes of size $k \leq p$

Step 4: (a) Learn an interaction matrix $W_m \in \{0,1\}^{k \times k}$ from $\mathbf{u}_1^{(m)}, \ldots, \mathbf{u}_N^{(m)}$, (b) estimate the parameters C_m and (c) estimate ρ_m by a line search.

Step 5: Update the m^{th} boosting model: $H_m(\mathbf{x}_t) := H_{m-1}(\mathbf{x}_t) + \rho_m h(\mathbf{x}_t; \{\mathcal{S}_m, W_m, C_m\})$

end if

end while

$m_{stop} := m$

Compute the Jacobian matrix $J_{m_{stop}}$ of $H_{m_{stop}}$ across time points, and threshold to get the final adjacency matrix $\hat{A}$.

In the DynBoost loop, $H_0(\mathbf{x}_t)$ is initialized with the mean values of the genes across the time points. The steps for estimating H in a subsequent iteration m are as follows: *Step 1* computes the residuals $\mathbf{u}_{t+1}^{(m)}$ for time points $t \in \{0, \ldots, N-2\}$. Computing the residuals in this step confers DynBoost its L_2-boosting nature. In *Step 2*, an early-stopping decision is made based on the comparison between the norms of the residuals and a prespecified stopping criterion ϵ. If the norms for all dimensions (genes) are less than ϵ, the algorithm exits the loop. In *Step 3*, a random subset $\mathcal{S}_m$ of size k is chosen among the genes of S that have a sufficiently high residual. This step constitutes the **randomization component** of the algorithm. *Step 4* uses the current residuals in the subspace to estimate the interaction matrix W_m and parameters C_m. ρ_m is then optimized through a line search. The m^{th} boosting model $H_m(\mathbf{x}_t)$ is updated in *Step 5* with the current W_m, C_m, and ρ_m estimates. If the prespecified number of iterations M has not been reached, the algorithm loops back to *Step 1*. Otherwise, it exits the loop and obtains the adjacency matrix estimate $\hat{A}$ by computing and thresholding the Jacobian matrix as in Eq. 2.1.

We next delineate how the interaction matrix W_m and model parameters C_m and ρ_m are estimated from residuals in *Step 4*.

2.4.2 Learning the interaction matrix

An important feature of the learning algorithm in *Step 4* lies in the decoupling of graph learning from model parameter learning. In *Step 4(a)* DynBoost employs HSIC, a kernel-based independence test suitable for capturing nonlinear interactions (11). The original HSIC implementation provides a test statistic that allows for a binary decision between the null hypothesis of independence and the alternative hypothesis of dependence. However, it is of interest to be able to quantify the strength of the interaction between any given pair of genes with a test that goes beyond a binary decision. Therefore, we enhanced the HSIC procedure with permutation tests so that

it yields a p-value π_{ij} for the significance of the interaction between genes i and j. Ranking the p-values from lowest to highest allows one to select the most significant interactions within the network. HSIC discovers up to a user-specified number of significant edges (`NE`≡number of edges) from the residual time series associated with a subset $\mathcal{S}_m$. The pseudo-code for our implementation is presented in Algorithm 2.

Algorithm 2 Selection of edges using HSIC for sequential data

Inputs

- Set of k genes : $\mathcal{S}_m$, Number of edges to pick : `NE`
- Level of the test : α
- Residuals : $\mathbf{u}_1^{(m)}, \ldots, \mathbf{u}_{N-1}^{(m)} \in \mathbb{R}^k$

for $i, j := 1$ to k **do**

Compute the HSIC p-value π_{ij} between sequence $(u_{1i}^{(m)}, ..., u_{(N-1)i}^{(m)})$ and the lagged sequence $(u_{2j}^{(m)}, ..., u_{Nj}^{(m)})$

Compute the HSIC p-value π_{ji} between sequence $(u_{1j}^{(m)}, ..., u_{(N-1)j}^{(m)})$ and the lagged sequence $(u_{2i}^{(m)}, ..., u_{Ni}^{(m)})$

Let $\pi_{i \leftrightarrow j} = \min\{\pi_{ij}, \pi_{ji}\}$

end for

Find the edges $(e_i, e_j) \in \mathcal{S}_m \times \mathcal{S}_m$ with $\pi_{i \leftrightarrow j} \leq \alpha$

Rank those edges by increasing order of p-value

Outputs : Select the first `NE` edges

The genes in $\mathcal{S}_m$ are selected randomly from the set $\mathcal{S}$ in *Step 3* of the DynBoost algorithm. This random selection is motivated by (10) where combining features of random forests and boosting algorithms gave robust results. In *Step 4(a)*, we apply HSIC on this random subset as a weak graph-learner to increase the robustness of the algorithm and reinforce its ability to focus on subspaces.

2.4.3 Autoregression using OKVAR

At each iteration m, an OKVAR model such as the one previously described in Eq. 2.3 is defined to work in the k dimensional subspace associated with the subset $\mathcal{S}_m$. Let $P^{(m)}$ denote a $p \times p$ diagonal matrix defined as a selection operator: $p_{ii}^{(m)} = 1$ if gene i is in $\mathcal{S}_m$ and $p_{ii}^{(m)} = 0$ otherwise. Formally, $h_m = h(\cdot; \{\mathcal{S}_m, W_m, C_m\})$ applies

only to selected residuals $\tilde{\mathbf{u}}_t^{(m)} = P^{(m)}\mathbf{u}_t^{(m)}$. However, we use the $\mathbf{u}_t^{(m)}$ notation below for simplicity. Once the matrix W_m is obtained from HSIC, we define Γ_m as in Eq. 2.5. Then, we only need to learn parameter C_m to complete *Step 4*. This estimation can be done via the functional estimation of h_m within the framework of regularization theory, i.e. the minimization of the empirical square loss and the square ℓ_2 norm of the function h_m, which imposes smoothness in the model. However, the cost function to be minimized must reflect the twofold goal of obtaining a final model H that fits the data well and predicts future time points successfully as well as accurately extracting the underlying regulatory matrix A. Following Eq. 2.1, the adjacency matrix A is estimated by the empirical Jacobian of H, expressed in terms of the empirical Jacobian $J^{(m)}$ of the base models h_m $(m = 1, \ldots, m_{stop})$ using the observed data (not residuals): $J_{ij} = \sum_{m=1}^{m_{stop}} \rho_m J_{ij}^{(m)} = \frac{1}{N}\sum_{m=1}^{m_{stop}} \rho_m \sum_{t=0}^{N-1} J_{ij}^{(m)}(t)$ where for a given time point t, the coefficients $J_{ij}^{(m)}(t)$ can be written as:

$$J_{ij}^{(m)}(t) = \frac{\partial h_m(\mathbf{x}_t)_i}{\partial(\mathbf{x}_t)_j} = \sum_{k=0}^{N-1} \frac{\partial(K^{(m)}(\mathbf{x}_k, \mathbf{x}_t)\mathbf{c}_k^{(m)})_i}{\partial(\mathbf{x}_t)_j}$$

When K is chosen as the Gaussian matrix-valued kernel defined in Eq. 2.4, $J_{ij}^{(m)}(t)$ is expressed in terms of the coefficients of matrix C_m in the following way:

$$J_{ij}^{(m)}(t) = -2\ \gamma_{ij}^{(m)} \sum_k c_{kj}^{(m)}\ (x_{ki} - x_{tj}) \exp\left(-\gamma_{ij}^{(m)}(x_{ki} - x_{tj})^2\right)$$

This expression suggests that controlling the sparsity of the coefficients of C_m will impact the sparsity of $J^{(m)}$ and will avoid too many false positive edges. The ℓ_1 norm of C_m is thus added to the previously proposed empirical square loss and ℓ_2 term to define the following cost function:

$$\mathcal{L}(C_m) = \sum_{t=0}^{N-1} \left\| \mathbf{u}_{t+1}^{(m)} - h_m(\mathbf{u}_t^{(m)}) \right\|^2 + \lambda \|h_m\|_{\mathcal{H}}^2 + \mu \|C_m\|_1 \tag{2.6}$$

The optimization problem can be stated as:

$$\begin{aligned} \underset{C_m \in \mathcal{M}^{N,p}}{\text{minimize}} \quad & \mathcal{L}(C_m) \\ \text{subject to} \quad & \lambda \geq 0,\ \mu \geq 0 \\ & \|h_m\|_{\mathcal{H}}^2 = \sum_{i,j=0}^{N-1} \mathbf{c_i^{(m)}}^T K^{(m)}(\mathbf{u}_j^{(m)}, \mathbf{u}_i^{(m)}) \mathbf{c}_j^{(m)} \\ & \|C_m\|_1 = \sum_{t=0}^{N-1} \sum_{j=1}^{p} |c_{tj}^{(m)}| \end{aligned}$$

This elaborate regularization model combining ℓ_1 and ℓ_2 penalties was first introduced by (36) as the **elastic net model**. The authors showed that this model not only achieves sparsity like lasso-penalized models, but also encourages a grouping effect, which is relevant for our case to highlight possible joint regulation among network variables (genes). We used a projected scaled subgradient method (27) to minimize the cost function in Eq. 2.6.

2.5 Implementation

2.5.1 Data description

The performance of the Dynboost algorithm is evaluated on a number of GRNs obtained from DREAM3 in-silico challenges. Specifically, the time series consisting of 21 points corresponding to size10 and size100 networks of E.coli (2) and Yeast (3) were selected, because networks of size 10 are able to model biological pathways more realistically, and networks of size 100 show scalability of the algorithm to larger numbers of nodes. The data were generated by imbuing the networks with dynamics from a thermodynamic model of gene expression and a Gaussian noise (25). These networks are subgraphs of the currently accepted *E. coli* and *S. cerevisiae* gene regulation networks, and exhibited varying patterns of sparsity and topological structure (Table 2.1). The total degree of a gene is the sum of its in- and out-degrees, while the average of the totals across genes gives the **average-degree** for the entire network. **Density** is the ratio of the number of edges in the network to the maximum possible number of edges. **Modularity** index and the number of modules given

Size10	Ecoli1	Ecoli2	Yeast1	Yeast2	Yeast3
Average-degree	2.2	3.0	2.0	5.0	4.4
Density	0.244	0.333	0.222	0.556	0.489
Modularity	0.016 (2)	0 (1)	0.260 (3)	0 (1)	0 (1)
Size100	Ecoli1	Ecoli2	Yeast1	Yeast2	Yeast3
Average-degree	2.5	2.38	3.32	7.78	11.02
Density	0.025	0.024	0.033	0.079	0.111
Modularity	0.643 (6)	0.661 (7)	0.681 (8)	0.328 (6)	0.088 (14)

Table 2.1: Average-degree, density, and modularity for DREAM3 networks.

in parentheses are the optimal values due to Newman (2004) (22). Yeast2 and Yeast3 have markedly higher average-degree, density; but lower modularity for both size10 and size100 networks. Ecoli2 is seen to be different from Ecoli1 in that it is denser, less modular, and has higher average-degree for size10; whereas these comparisons are reversed for size100. Yeast1 is observed to be closer to Ecoli networks for all three statistics.

2.5.2 Hyperparameters and model assessment

Since the DynBoost algorithm depends on a number of tuning parameters, the following choices were made: the algorithm stops when the norm for the residual vector is below an $\epsilon = 10^{-2}$ threshold. The size of random subnetworks k in *Step 3* of the algorithm was optimized using grid-search; and set to 3 genes for size10 and to 25 for size100 networks. In *Step 4(a)*, the level of the HSIC independence-test is set to a conservative $\alpha = 1\%$, and data points were permuted 200 times to obtain the p-value of the test. The most significant 1 and 10 edges (`NE`) are then selected in HSIC for size10 and size100 networks respectively. Selecting a low number of edges in HSIC helps obtaining a weak graph from each base learner, which is an important component of boosting methods. If the algorithm fails to find any significant interactions in HSIC, the subnetwork is discarded and a new $k \times k$ subnetwork is randomly chosen. This procedure is repeated for a maximum of 100 trials if `NE` significant edges are not found. The parameters of the Gaussian matrix-valued kernel α and β (Eq. 2.5), and the regularization parameters of the elastic net model (Eq. 2.6) in *Step 4(b)* are optimized through grid-search. For size10 networks, the ridge penalty is set

to $\lambda = 10^3$ and the ℓ_1 penalty to $\mu = 10^{-4}$. For size100 networks, the corresponding values for λ and μ are both set to 10^{-3}. α and β are set to 1 for both sizes.

The performance of the algorithm is assessed using the following standard metrics: the receiver operating characteristic curve (ROC), and the area under it (AUROC), the area under the precision-recall curve (AUPR), F1-score, sensitivity, specificity, positive predictive value, and Matthew's correlation coefficient. Due to lack of space, we present only AUROC and AUPR results but note that the remaining ones were given due consideration. Function g in (Eq. 2.1) is a hyperbolic tangent transformation applied to the normalized coefficients of the Jacobian[2]: $\hat{A}_{ij} = \tanh\left(\frac{J_{ij}}{\|J\|_F} - \delta\right)$, with δ being a user-specified threshold.

2.5.3 Consensus from multiple runs and bootstrapping

As DynBoost residuals diminish rapidly, there is a risk that the potential regulators and their targets may not be fully explored by the random subnetwork procedure of the algorithm. To address this issue, the algorithm was run 100 times and a *consensus* network was built by combining the prediction for each single run. Then, for each pair of nodes the frequency that the edge appears over multiple runs was calculated. If the frequency was above a threshold the edge was kept, otherwise discarded; thus yielding the final network prediction.

Another way to increase stability (17) is to combine bootstrapping of the data with runs of the algorithm. Specifically, 100 bootstrapped samples were obtained from the time series of each network. Given the dependence over time, a block-bootstrap algorithm was used with a block length of 7 for both size10 and size100 networks due to (24). The Dynboost algorithm produced a reconstructed network from each bootstrapped sample unless early stopping occurred in the first iteration due to small residuals. The edge frequency was calculated as in the multiple-runs case.

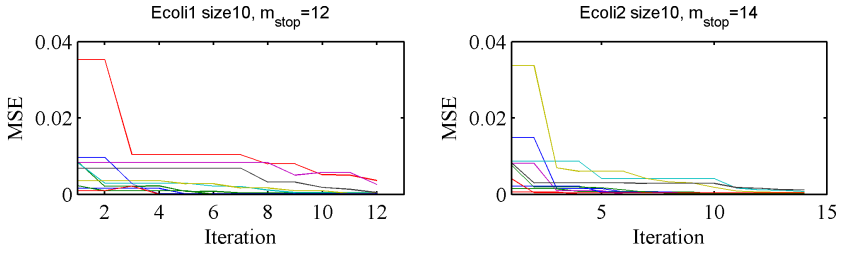
In both cases (multiple runs and bootstrapping), it is necessary to adjust the consensus threshold level according to the size, density, and modularity of a network. In general, the larger the size for a biological network, the bigger are the combinatorial challenges for

[2]For a matrix Q, $\|Q\|_F = \sqrt{\sum_{i,j} Q_{ij}^2}$ is the Frobenius norm of Q.

discovering true edges and avoiding false ones. Therefore, the consensus threshold has to be set to smaller values for larger networks. For a fixed size, the threshold will depend on the density and modularity of the network. Denser and more modular networks have greater instances of co-regulation for certain genes, which lowers prediction accuracy for network inference algorithms (9) and leads to a greater number of false positives. Thus, we recommend using lower consensus thresholds for denser and more modular networks as well.

2.6 Results and discussion

2.6.1 Numerical results



(a) Size10 Ecoli1 **(b)** Size10 Ecoli2

(c) Size100 Ecoli1 **(d)** Size100 Ecoli2

Figure 2.1: Mean squared error of genes at termination. (a) Size10 Ecoli1 (b) Size10 Ecoli2 (c) Size100 Ecoli1 (d) Size100 Ecoli2. The algorithm terminated after 12, 14, 4, and 4 iterations respectively.

The DynBoost algorithm succeeds to fit the observed data and exhibits fast convergence. Figure 2.1 shows the results from size10 and size100 Ecoli2 networks. We note that the algorithm is rich and flexible enough to have the mean-squared-error for genes diminish rapidly towards zero in only 5-10 iterations.

Size 10	Ecoli1		Ecoli2		Yeast1		Yeast2*		Yeast3*	
	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR
Base learner	0.614	0.226	0.616	0.406	0.578	0.343	0.580	0.233	0.612	0.300
Multiple-run DynBoost	**0.654**	**0.301**	0.718	0.659	0.628	0.160	0.720	0.352	0.684	**0.525**
Bootstrap DynBoost	0.620	0.241	**0.923**	**0.876**	**0.722**	**0.393**	**0.733**	**0.386**	**0.695**	0.359
Team 236	0.621	0.197	0.650	0.378	0.646	0.194	0.438	0.236	0.488	0.239
Team 190	0.573	0.152	0.515	0.181	0.631	0.167	0.577	0.371	0.603	0.373

Table 2.2: AUROC and AUPR for DREAM3 size10 networks. Multiple-run and bootstrap results are from consensus networks. The numbers in **boldface** are the maximum values of each column. (* Consensus thresholds for Yeast2 and Yeast3 are different due to their higher density and average-degree.)

The performance of the DynBoost algorithm for prediction of the network structure is given in Tables 2.2 and 2.3. The results show a comparison between the base learner alone, boosting with multiple runs and boosting with bootstrapping. The base learner is an elastic OKVAR model learnt using HSIC on the whole set of genes $\mathcal{S}$. The AUROC and AUPR values given strongly indicate that DynBoost was able to outperform the teams in DREAM3 that exclusively used the same set of time series data, namely team 236 and team 190. Even though the base learner outperformed the best competitor for only Yeast2 and Yeast3, the multiple-run and/or the bootstrap consensus results achieved superior AUROC and AUPR results for all networks. We particularly note that the DynBoost consensus runs exhibited excellent AUPR values compared to those obtained by teams 236 and 190.

Size100	Ecoli1		Ecoli2*		Yeast1		Yeast2		Yeast3	
	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR
Base learner	0.509	0.017	0.500	0.057	0.506	0.023	0.512	0.049	0.563	0.059
Multiple-run DynBoost	0.548	**0.036**	0.577	0.059	0.554	0.025	0.560	0.059	0.548	0.053
Bootstrap DynBoost	**0.588**	0.031	**0.583**	**0.075**	**0.563**	0.029	**0.607**	**0.077**	**0.590**	**0.067**
Team 236	0.527	0.019	0.546	0.042	0.532	**0.035**	0.508	0.046	0.508	0.065

Table 2.3: AUROC and AUPR for DREAM3 size100 networks. Multiple-run and bootstrap results are from consensus networks. The numbers in **boldface** are the maximum values of each column. (* Ecoli2 has a strong star-topology, which suggests a different consensus threshold for this network.)

A comparison between algorithms for size100 networks (Table 2.3) shows that the DynBoost base learner again outperforms Team 236 in only Yeast2 and Yeast3 AUROC scores. Consensus runs, on the other hand, achieve superior AUROC and AUPR performance for all networks. It is noticeable however, that size100 AUROC values are about 10 to 20% lower than size10 counterparts and AUPR values in all rows have stayed lower than 10%. These results are more related to the size of the network rather than the choice of algorithm. A similar decline is also observed in Team 236 results, the only team that exclusively used

time series data for the size100 challenge. The primary reason for this decline is the abundance of false positives in exponentially larger subspaces that drive the scores in spite of the high success in discovering true edges of gold-standard networks. Sparser models may increase the performance for size100 challenges.

It should be noted that there is no clear winner between the multiple-run and the bootstrap consensus runs of the Dynboost algorithm. Even though the bootstrap version achieves in general higher AUROC and AUPR values for both size10 and size100 networks, there are instances where it underperforms. The randomization associated with bootstrap samples may be more effective in increasing the stability of the algorithm compared with the randomization due to the different runs. However, the complex nonlinear dynamics of the data prevent us from reaching a definite conclusion from our experimental results.

The consensus thresholds for multiple-run and bootstrap experiments were chosen taking into account network properties such as size, density, modularity, average-degree, and topology. For size10 networks, Yeast2 and Yeast3 have substantially higher density and average-degree suggesting lower consensus thresholds. Thus, for multiple-run experiments, we used a threshold of 50% for Ecoli1, Ecoli2, Yeast1; and 40% for Yeast2 and Yeast3. The threshold for bootstrap experiments was 50% for Ecoli1, Ecoli2 and Yeast1; and 30% for Yeast2 and Yeast3. For size100 networks, we made use of the prior information that Ecoli2 has a star-topology comprised of few central hubs that regulate many genes. Since it is more difficult to reconstruct such special modularities, one should expect to observe lower edge frequencies in bootstrap and multiple runs. Thus, a smaller consensus threshold would be appropriate. For the multiple-run experiments, we used 20% for Ecoli2 and 40% for all other networks. For bootstrap runs we further lowered the threhold to 5% for Ecoli2 and 35% for the remaining networks.

Although there is no information on the structure of team 236's algorithm, its authors responded to the post-competition DREAM3 survey stating that their method employs Bayesian models with an in-degree constraint (25). This in-degree constraint may explain their particularly poor AUROC and AUPR performance for the high average-degree networks Yeast2 and Yeast3 (average-degree values in Table 2.1). Team 190 (Table 2.2)

reported in the same survey that their method is also Bayesian with a focus on nonlinear dynamics and local optimization. This team did not submit predictions for the size100 challenge.

2.6.2 Discussion

Gene regulatory network inference has been cast as a feature selection problem in numerous works. For linear models, lasso-penalized regression models have been effectively used for this task (23; 9; 28). As an alternative to lasso regularization, an L_2-boosting algorithm was proposed in (1) to build a combination of linear autoregressive models that work for very large networks. In nonlinear nonparametric modeling, random forests and their variant extra-trees (13) have recently won the DREAM5 challenge by using static data and solving p regression problems. In this approach, importance measures computed on the explanatory variables (genes) have the potential to identify regulators for each one of the target genes.

Compared to these approaches, DynBoost shares features with boosting and randomization-based methods, an example to the latter shared feature being the use of a random subnetwork at each iteration. DynBoost exhibits fast convergence in terms of mean squared error due to the flexibility of OKVAR in capturing nonlinear dynamics. Further, it uses an original and general way to extract the regulatory network through the Jacobian matrix of the estimated nonlinear model. The control of sparsity on the Jacobian matrix is converted into a constraint of the parameters of each base model h_m, for which the independence matrix W_m has been obtained by a statistical independence test. It should also be emphasized that prior information about the regulatory network can easily be incorporated into the algorithm by fixing known coefficients of the independence matrices used at each iteration. DynBoost also directly extends to additional observed time series from different initial conditions. Although we only showed one specific OKVAR model based on an extension of the Gaussian kernel, which is of special interest for network inference, we note that other kernels can be defined and be more appropriate depending on the focus of the study.

Bibliography

[1] Anjum, S., Doucet, A. and Holmes, C.C. (2009) A boosting approach to structure learning of graphs with and without prior knowledge. *Bioinformatics*, 25(22), 2929-2936.

[2] Bansal, M., Belcastro, V., Ambesi-Impiombato, A., di Bernardo, D. (2007) How to infer gene networks from expression profiles. *Molecular systems biology*, 3:78.

[3] Breiman, L. (2001). Random forests. *Machine Learning* 45:1532.

[4] Brouard, C., d'Alché-Buc, F. and Szafranski, M. (2011) Semi-supervised Penalized Output Kernel Regression for Link Prediction *ICML-11*, 593-600.

[5] Bühlmann, P. and Yu, B. (2003) Boosting with the L_2 loss. *Journal of the American Statistical Association*. 98(462): 324-339.

[6] Cam, H., Balciunaite, E., Blais, A., Spektor, A., Scarpulla, R.C., Young, R., Kluger, Y., Dynlacht, B.D. (2004). A common set of gene regulatory networks links metabolism and growth inhibition *Molecular Cell*, 16 (3), pp. 399-411.

[7] Dietterich, T.G. (2000). Ensemble Methods in Machine Learning. *Multiple Classifier Systems.*

[8] Friedman, J.H. (2001). Greedy function approximation: a gradient boosting machine. *Ann. Statist.* 29 11891232.

[9] Fujita, A., Sato, J.R., Garay-Malpartida, H.M., Yamaguchi, R., Miyano, S., Sogayar, M.C., Ferreira C.E. (2007). Modeling gene expression regulatory networks with the sparse vector autoregressive model. *BMC Syst Biol.*, 1, 39.

[10] Geurts, P., Wehenkel, L., d'Alché-Buc, F. (2007) Gradient boosting for kernelized output spaces. *ICML-2007*. 289-296.

[11] Gretton, A., Fukumizu, K., Teo, C.-H., Song, L., Schölkopf, B., Smola, A. (2007) A Kernel Statistical Test of Independence. *NIPS* 21, 2007.

[12] Gupta, R., Stincone, A., Antczak, P., Durant, S., Bicknell, R., Bikfalvi, A., Falciani, F. (2011) A computational framework for gene regulatory network inference that combines multiple methods and datasets. *BMC Systems Biology*, 5:52 doi:10.1186/1752-0509-5-52

[13] Huynh-Thu, V. A., Irrthum, A., Wehenkel, L., and Geurts, P. (2010). Inferring regulatory networks from expression data using tree-based methods. *PLoS ONE*, 5(9), e12776.

[14] Jesmin, J., Rashid, M.S., Jamil, H., et al. (2010). Gene regulatory network reveals oxidative stress as the underlying molecular mechanism of type 2 diabetes and hypertension. *BMC Med Genomics*;3:45.

[9] Marbach, D., Prill, R.J., Schaffter, T., Mattiussi, C., Floreano, D., Stolovitzky, G. (2010) Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286-6291

[16] Markowetz, F. and Spang, R. (2007) Inferring cellular networks - a review. *BMC Bioinformatics*, 8(Suppl 6):S5.

[17] Meinshausen, N. and Bühlmann, P. (2010). Stability selection (with discussion). *Journal of the Royal Statistical Society*: Series B, 72, 417-473.

[18] Micchelli, C.A. and Pontil, M. (2005). On learning vector-valued functions *Neural Computation*, 17(1):177-204

[19] Morrissey, E.R., Juarez, M.A., Denby, K.J. and Burroughs, N.J. (2010). On reverse engineering of gene interaction networks using time course data with repeated measurements. *Bioinformatics*, 26(18):2305-12.

[20] Mukhopadhyay N.D., Chatterjee S. (2007) Causality and pathway search in microarray time series experiment. *Bioinformatics*, 23:442-449

[21] Nagarajan R., Upreti M. (2010) Granger causality analysis of human cell-cycle gene expression profiles. *Stat. Appl. Genet. Mol. Biol*; 9:31

[22] Newman, M.E.J. (2004). Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69, p. 066133

[23] Perrin, B., Ralaivola, L., Mazurie A., Bottani, S., Mallet, J., d'Alché-Buc, F. (2003). Gene networks inference using dynamic Bayesian networks. *Bioinformatics*, 19 Suppl 2, II138-II148.

[24] Politis, D.N., White, H., and Patton, A.J., (2009). Correction: Automatic Block-Length Selection for the Dependent Bootstrap, *Econometric Reviews*, 28(4), 372-375.

[25] Prill, R.J., Marbach, D., Saez-Rodriguez, J., Sorger, P.K., Alexopoulos, L.G., et al. (2010) Towards a Rigorous Assessment of Systems Biology Models: The DREAM3 Challenges. *PLoS ONE* 5(2): e9202. doi:10.1371/journal.pone.0009202

[26] Rangel,. C, Angus J., Ghahramani Z., Lioumi M., Sotheran E., Gaiba A., Wild D. L.(2004): Modeling T-cell activation using gene expression profiling and state-space models. *Bioinformatics*, 20(9): 1361-1372.

[27] Schmidt, M., Fung, G. and Rosales, R. (2009) Optimization methods for l1-regularization. *University of British Columbia, Technical Report TR-2009-19*.

[28] Shojaie, A. and Michailidis, G. (2010a) Penalized Likelihood Methods for Estimation of Sparse High Dimensional Directed Acyclic Graphs. *Biometrika*, 97(3), 519-538.

[29] Shojaie, A. and Michailidis, G. (2010b) Discovering Graphical Granger Causality Using a Truncating Lasso Penalty. *Bioinformatics*, 26(18), i517-i523

[30] Sîrbu, A., Ruskin, H.J. and Crane, M. (2010) Comparison of evolutionary algorithms in gene regulatory network model inference. *BMC bioinformatics* 11(1), 59.

[31] Smet, R.D. and Marchal, K. (2010) Advantages and limitations of current network inference methods. *Nat. Rev. Microbiol.*, 8, 717729.

[32] Stolovitzky G., Monroe D. and Califano, A. (2007) Dialogue on reverse-engineering assessment and methods: The dream of high-throughput pathway inference. *Annals of the New York Academy of Sciences* 1115:122.

[33] Stolovitzky, G., Prill, R. J. and Califano, A. (2009) Lessons from the DREAM2 Challenges. *Annals of the New York Academy of Sciences*, 1158: 159195. doi: 10.1111/j.1749-6632.2009.04497.x

[34] Yip, K.Y., Alexander, R.P., Yan, K.K., Gerstein, M. (2010). Improved reconstruction of in silico gene regulatory networks by integrating knockout and perturbation data. *PLoS One*, 5(1):e8121

[35] Yu, J., Smith, V.A., Wang, P.P., Hartemink, A.J., Jarvis, E.D. (2004) Advances to Bayesian network inference for generating causal networks from observational biological data. *Bioinformatics*; 20:3594-3603

[36] Zou, H. and Hastie, T. (2005). Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society* B. 67(Part 2):301320.

CHAPTER III

A reassessment of consensus clustering for class discovery and cluster stability

3.1 Abstract

Consensus clustering (CC) is an unsupervised class discovery method highly popular for defining sample subtypes from high-dimensional data. The "consensus" rate between two samples measures how frequently they are grouped together in multiple clustering runs under a certain degree of perturbation. The matrix of pairwise consensus rates has been used as a between-sample distance matrix for (1) visualization of clusters, (2) assessing cluster stability and (3) estimating the optimal number of clusters. However, the sensitivity and specificity of CC have not been systemically studied. We investigated the properties of the consensus matrix and its derived statistics along with multiple other clustering methods in (1) a published glioblastoma dataset, simulated datasets (2) with realistic gene-gene correlation structure, (3) with known lack of structure, and (4) with known number of clusters and varying degrees of separation. We found that many clustering methods were able to divide simulated unimodal data (i.e. without subgroups) into prespecified numbers of clusters, and CC was able to show apparent stability of such chance partitioning of random data. We found that the consensus matrix by itself is not an automatically useful inference tool, but its distribution features can be. One summary of such features, the proportion of ambiguously clustered (PAC) pairs, reported the known number of clusters

better than other common indices such as CDF, $\Delta(K)$, Silhouette Width, GAP-PC, and in many situations, CLEST. We also showed by using synthetic null datasets that using only the most discriminant genes to validate clusters often exaggerates cluster structure, as the latter can be strongly driven by the gene-gene correlations in the data. We conclude that CC is a powerful tool for identifying cluseters in the presence of genuine structure, but can lead to false positives in the exploratory phase of many studies if the structure is subtle or absent, and if the implementation and inference are not carried out with caution. Our results led to a specific set of recommendations regarding prudent practices in using CC.

3.2 Author Summary

Consensus clustering (CC) is rapidly becoming the algorithm of choice for unsupervised class discovery from gene expression datasets. It has been used in real studies as both a visualization and inference tool, and has been cited 437 times since 2003. One recent study by The Cancer Genome Atlas (TCGA) consortum aimed at identifying and validating glioblastoma subtypes using CC-based cluster analysis. But as is typical in this type of report, neither the strength of the evidence nor the sensitivity of the method was evaluated. Using this dataset as an example, I highlight the necessity for caution in using CC for subtype discovery by comparing results from this dataset with those from a series of randomly simulated datasets known to lack structure. My findings underscore the value of the systematic use of null distributions simulated with gene-gene correlation from real data, and I describe an improved CC-based approach using (1) K-means as the base clustering method and (2) probability of ambiguous clustering (PAC) as the measure to infer the optimal number of clusters. To systematically compare the performance of multiple commonly used methods, I develop the 'identifiability chart' to directly visualize the parameter subspace where a given method can correctly infer the true number of clusters, where the key parameters are the known number of clusters in the simulation and the known degree of cluster separation. I show using identifiability zones that PAC outperforms most common clustering methods in scenarios closely resembling real studies such as those encountered

in today's gene expression analysis of tumor samples. I conclude by summarizing the potential for exaggerating cluster strength in CC and recommending prudent practices when conducting unsupervised class discovery with a new dataset, especially for those having potentially weak structure.

3.3 Introduction

Cluster analysis is a key method for the unsupervised discovery of molecular subtypes from high-dimensional biological datasets. For example, in the last decade, applying this method to microarray-based gene expression data has become the essential approach for defining molecular subtypes of various cancers (22–24). This class-discovery task involves two essential aspects: first, determining if there is evidence for the existence of substructure in the data; and second, if substructure exists, determining the optimal number of clusters. Since these two aspects involve different questions, one should employ different null datasets to investigate the answer to each. For the first question, a *global* null should be constructed to make the 'substructure vs. no-substructure' decision, whereas a set of *study-specific positive null datasets* should be used for the second question to estimate the number of clusters.

The construction of the null distribution for assessing the significance of clustering results is a critical step. A theoretically well-defined null distribution for the unsupervised discovery of clusters does not exist because it is not possible to make universally true model assumptions for the distributions that real high-dimensional datasets come from. Therefore, a suitable null distribution has to be formed empirically from the data under study. This empirical distribution has to account for the correlations among genes in the original data because the gene-gene correlation structure affects the shape of the sample distribution in high-dimensional space, thereby potentially giving rise to apparent cluster stability driven by highly correlated gene sets. The globall null we used in this study consists of a random unimodal distribution of principal component scores projected so as to have the same gene-gene correlation as in the published dataset. We refrain from using the terms *random* and

homogeneous for the global null because the gene-gene correlations confer some level of structure to the data as explained above.

If substructure exists in the data, the next question to answer is the optimal number of clusters. In contrast with the first question, the decision for the second one no longer involves testing a single global null hypothesis, but entails testing a consolidation of multiple positive null hypotheses. For instance, K clusters cannot be reported as optimal unless the null hypotheses of $K-1$ and $K+1$ are both rejected. To evaluate the power of clustering methods to distinguish between multiple alternative hypotheses, we simulated a series of *controlled positive* null datasets with known number of clusters, varying degrees of separation between clusters, and local density from the real dataset. In this setting, rejecting the set of *study-specific* positive null hypotheses would be relatively more difficult than rejecting the globall null hypothesis because the positive datasets inherently have more structure than a global null dataset.

Unsupervised class discovery is difficult to be cast into an axiomatic setting because real datasets have a complex variance-covariance structure, and thus the null distribution is hard to justify in a parametric framework. This difficulty led to the popularity of resampling-based methods, where multiple subsamples of the original dataset are clustered to assess the stability of results with respect to sampling variability. One such method, CLEST (10) computes cross-validation errors for a range of total cluster numbers, and compares the results to a null distribution to assess significance. Another resampling-based method, consensus clustering (CC)(11), has recently gained popularity over other clustering methods for tasks such as tumor subtype discovery from microarray gene expression datasets. CC calculates a "consensus" rate between all pairs of samples that measures how frequently each pair is grouped together in multiple clustering runs under a certain degree of subsampling. Consensus rates compared across different numbers of clusters can then be used for (1) visualization of putative clusters, (2) estimating the optimal number of clusters, and (3) assessing the stability of clusters.

The main assumption of CC is that if the items under study were drawn from distinct sub-populations, different subsamples drawn from the same set of items would exhibit simi-

lar cluster numbers and compositions. Hence, clusters robust to sampling variability would provide evidence for the real structure in the data(11). This assumption is easily validated in cases of well-separated clusters; however, the question of whether robust clusters might arise from *structureless* data has not been systemically studied. This question underscores a potential limitation of resampling-based methods, namely the difficulty of formally evaluating the significance of produced results by explicitly modeling the assumptions underlying the data-generation process. Although this limitation is acknowledged in the literature (11), due caution in using CC as an inference tool has not been widely heeded. Specifically, the possibility of CC to show robust clusters even with poorly separated items has been largely underestimated in certain research areas such as cancer genomics.

CC is a sensitive heuristic that has proven useful for visualizing cluster stability. However, it is important to distinguish the utility of CC as an inference tool from its visualization function. The apparent stability of clusters in CC may not necessarily provide sufficient evidence for their existence; and usually requires further work for cluster validation. For instance, outliers separated by a base clustering method (i.e average-linkage hierarchical clustering) will appear as robust clusters in CC, because repeated runs of the base method will exaggerate the weak clustering signal from outliers. Making inferences from CC, thus, has the drawback of potentially declaring structure in the data when there is no significant separation or local compactness. In this study, we demonstrate this tendency using simulated datasets with known lack of clusters. We also present ways to systemically study the strength of clustering in a real dataset. Specifically, we emphasize that it is necessary to compare clustering results from the original data with those from a suitably-formed null distribution.

Validation of clusters is particularly important when using gene expression data because clustering results in this context are especially sensitive to noise and susceptible to overfitting due to relatively small sample sizes and very high dimensionality of the data ($N << p$) (11). However, cluster validation is an elusive task in most studies due to the lack of an *a priori* objective criterion such as known class labels with which predictions can be compared. When external labels for clusters are not available, an evaluation based on

internal validation measures, which base their quality estimate on the information intrinsic to the data alone, becomes necessary (25).

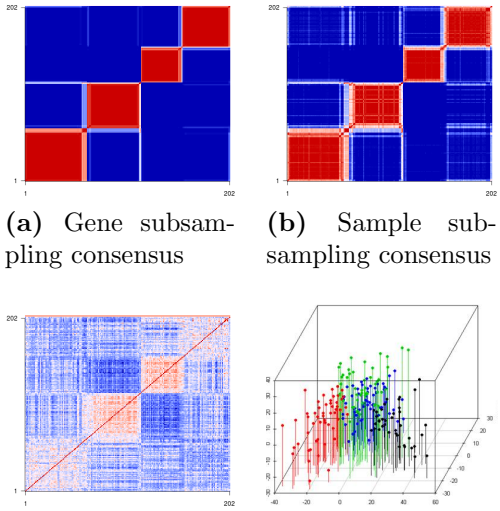
There are multiple types of clusteredness measures for internal validation, including compactness, connectedness, separation, or combinations based on these. An example that uses a compactness measure is the GAP-statistic (9), which compares the pooled within-cluster sum of squares to its expected value under a null distribution. Compactness alone may not be sufficient to model the clusteredness in a real dataset as the degree of separation between clusters can strongly contribute to the clustering signal. Thus, measures combining intra-cluster homogeneity (compactness) and inter-cluster separation have become particularly popular. A well-known example is the Silhouette Width (8 and Materials and Methods 3.6.7.4), which is based on a nonlinear combination of the two measures, and has been widely used in cancer genomics to validate the number of classes as well as to characterize tumor samples as 'core' and 'non-core'.

For this study, we chose to investigate clustering signals in multiple glioblastoma multiforme (GBM) datasets. GBM was the first cancer type studied by The Cancer Genome Atlas (TCGA) Research Network (2), and was reported in a recent study to have four molecular subtypes according to gene expression clusters discovered by CC (3). For our analysis, we employed the TCGA's first and second GBM cohorts as well as a validation dataset from multiple previous studies (4–7) that was also used in (3). These datasets are hereafter referred to as **GBM1**, **GBM2**, and **Validation** respectively.

3.4 Results and Discussion

3.4.1 PCA visualization and sample-sample correlation structure demonstrate the over-interpretation potential of consensus clustering - example from a real dataset.

CC consists of repeated runs of a base clustering method on randomly subsampled datasets. For this study, we implemented CC with k-means (as the base clustering method) on 500 subsamples obtained by taking a random 80 % subset of genes or samples. Motivated



(a) Gene subsampling consensus

(b) Sample subsampling consensus

(c) Sample-sample correlation

(d) First three PCs

Figure 3.1: (a,b) GBM1 $K = 4$ gene-subsampling and sample-subsampling consensus heatmaps, (c) sample-sample correlation heatmap, (d) 4 k-means clusters visualized on axes x (PC1), y (PC3), z (PC2). The variances explained by PC1-PC2-PC3 are 21.6%, 9.9%, and 7.9% respectively. The color scale on consensus heatmaps ranges from 0 to 1, where 0 corresponds to blue, 1 corresponds to red, and 0.5 corresponds to white. The same color scale is used throughout the paper unless otherwise stated.

by the TCGA study by Verhaak *et al.*(3), we ran consensus clustering on **GBM1** with $K = 4$ (number of clusters hereafter is denoted with K), and compared the clusters in consensus heatmaps with those visible in the sample-sample correlation heatmap and the three-dimensional PCA visualization of the dataset (Figure 3.1). The consensus heatmaps in Figures 3.1a and 3.1b show four crisp clusters. The crispness of clusters can be interpreted as a strong indication for inherent structure in **GBM1**, however the signal for $K = 4$ in the sample-sample correlation heatmap (Figure 3.1c) is substantially weaker with many samples having strong correlation with samples in a different cluster. The three-dimensional PCA plot (Figure 3.1d) does not show distinct gaps among the four clusters either. These findings point at the potential of CC to over-state the robustness of clusters and raise the questions of (1) when it is possible to know if one is merely partitoning data from a unimodal distribution into multiple categories, (2) how optimal K should be determined, and (3) what the nature of cluster validation should be. In the following, we present the performance of CC on null datasets first with simple illustrative examples, and then with

more realistic ones that carry the same gene-gene correlation structure as **GBM1**.

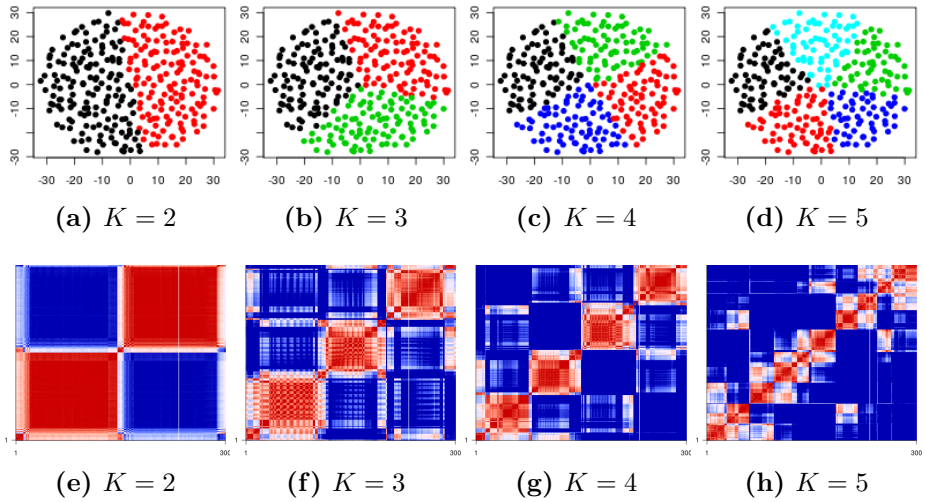
3.4.2 Performance of CC on null datasets

3.4.2.1 Simple illustrative examples: Circle1 and Square1

One of the main questions answered by CC-based cluster analysis is whether the dataset of interest has structure. The underlying assumption is that the structure of the distribution that the data come from can be captured by repeated resampling of the genes or samples. However, the significance of the results reported by CC can be better understood by comparing them to a null distribution. There have been several null distributions proposed in the literature for testing the hypothesis of the presence of structure in the data. For instance, the null datasets can be sampled randomly from a uniform or unimodal distribution where the parameters of the distribution are estimated by the statistics in the test dataset. Another variant for generating the null datasets is that these statistics can be derived from either the original vectors of the data, or the ones projected onto a principal component space. Some of these null distributions are stronger than others, so the decision of significance relies heavily on the choice of the null distribution.

We first tested the performance of CC on individually realized datasets from a random unimodal distribution that is known to lack compactness and separation. The datasets we generated, namely $Circle1$ and $Square1$ are imposed a grid distribution on PC1 and PC2. They have more structure compared to a matrix completely filled with $Normal(0, 1)$ samples. Specifically, they have higher gene-gene correlation and more variance explained by PC1 and PC2 than a $Normal(0, 1)$. However, they are still unimodal in the sense that they lack compactness and separation.

Circle1 and Square1: For $Circle1$, we generated 400 samples, each having 1000 genes, with a $Normal(0, 1)$ error perturbation in a 20×20 square grid formation in the PC1-PC2 space . PC vectors beyond the second only had the error component, so there was very weak gene-gene correlation. Samples occupying the four corners in the two-dimensional PC-space were trimmed to leave a circular topology of $\sim$ 300 samples. Figure 3.2 a-d



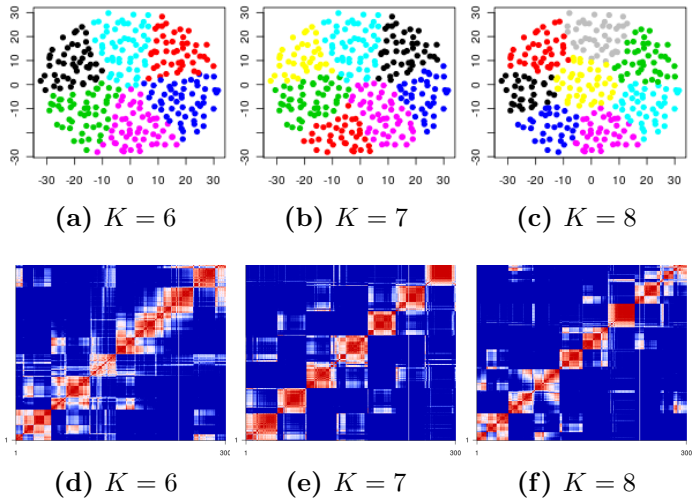
(a) $K = 2$ **(b)** $K = 3$ **(c)** $K = 4$ **(d)** $K = 5$

(e) $K = 2$ **(f)** $K = 3$ **(g)** $K = 4$ **(h)** $K = 5$

Figure 3.2: (a-d) *Circle*1 k-means partitioning for $K = \{2, 3, 4, 5\}$ displayed on PC1 (17.7%) on the x-axis *vs* PC2 (15.1%) on the y-axis. (e-h) K-means consensus heatmaps for $K = \{2, 3, 4, 5\}$ with 80% sample subsampling. Even though the distribution is unimodal, consensus heatmaps can show apparent clusters.

show *Circle*1 samples in the PC1-PC2 space with colors obtained from a single k-means partitioning at $K = \{2, 3, 4, 5\}$. While there is no inherent structure in the data, k-means can nonetheless partition the samples into K subgroups. Importantly, we see in Figure 3.2 e-k that consensus runs are able to show a high stability of clusters, particularly for $K = 2$ and $K = 4$, than would be conceived of from the k-means results. Further, the stability is even more improved for larger K (such as 7 or 8) as seen in Figure 3.3.

The apparent stability in the example above is potentially contributed by random occurrance of locally dense clusters. A second factor is the presence of outliers or "corners" of the sample distribution. To explicitly model this, we generated the *Square*1 simulation, 100 samples with 1000 genes each, from a random unimodal distribution. The samples were arranged in a 10×10 square grid on the PC1-PC2 plot with a $Normal(0, 1)$ error perturbation. As in the *Circle*1 case, PC vectors beyond the second only had the random error component, and a very weak gene-gene correlation structure. Figure 3.4 a-d show *Square*1 samples in the PC1-PC2 space with colors obtained from a single k-means partitioning at $K = \{2, 3, 4, 5\}$. Again, Figure 3.4 e-f show k-means-based CC yielded consensus heatmaps for the corresponding K values with 80% sample subsampling. CC was able to show appar-

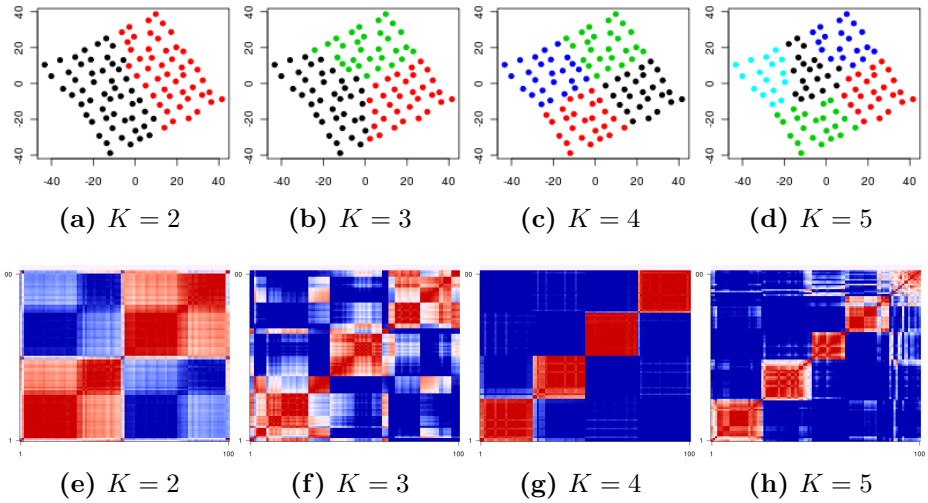


(a) $K = 6$ **(b)** $K = 7$ **(c)** $K = 8$

(d) $K = 6$ **(e)** $K = 7$ **(f)** $K = 8$

Figure 3.3: (a-c) *Circle*1 k-means partitioning for $K = \{6, 7, 8\}$ displayed on PC1 (17.7%) on the x-axis *vs* PC2 (15.1%) on the y-axis. (d-f) K-means consensus heatmaps for $K = \{6, 7, 8\}$ with 80% sample subsampling. Consensus heatmaps for the unimodal *Circle*1 distribution can show apparent clusters even with $K = 7$ and 8.

ent stability especially for the case of $K = 4$. The observation that $K = \{2, 3\}$ were not as 'clean' is because the four corners of the distribution helped to anchor the $K = 4$ partitions and lend them stability.



(a) $K = 2$ **(b)** $K = 3$ **(c)** $K = 4$ **(d)** $K = 5$

(e) $K = 2$ **(f)** $K = 3$ **(g)** $K = 4$ **(h)** $K = 5$

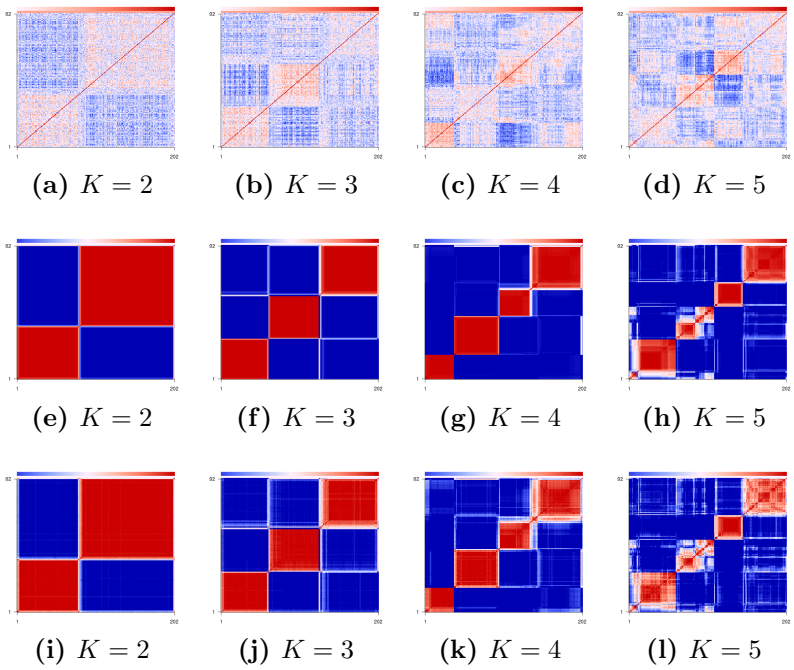
Figure 3.4: (a-d) *Square*1 k-means partitioning for $K = \{2, 3, 4, 5\}$ displayed on PC1 (21.8%) on the x-axis *vs* PC2 (19.1%) on the y-axis. (e-h) K-means consensus heatmaps for $K = \{2, 3, 4, 5\}$ with 80% sample subsampling. Even though *Square*1 has unimodal distribution, consensus heatmaps show apparent clusters for certain K values. Visual evidence alone can be misleading, hence formal approaches are needed to test validity.

We show with these examples that CC is able to show apparent stability of chance partitioning of individually realized datasets from a random unimodal distribution, and thus has potential to lead to over-interpretation of cluster stability in a real study. Another lesson learned here is that, visual evidence alone can be misleading and formal inference approaches are needed to test validity of clusters. This is particularly relevant in that many studies utilizing CC neglected to evaluate the strength of evidence and relied on visualization of CC matrix to declare clusters. Next, we elaborate on constructing more realistic empirical null dataset having the same gene-gene correlation as **GBM1** and then evaluating **GBM1**'s clustering signal in the context of the signal found in null datasets.

3.4.2.2 More realistic examples: *pcNormal*

The gene-gene correlation structure in a data set determines the "shape" of sample clusters in the high dimensional space whereas the sample-sample correlations can allow a direct visualization of cluster strength. Therefore, the gene-gene correlation structure can be a key parameter in the unsupervised discovery of classes, and needs to be accounted for in a null distribution. Thus, a suitable empirical null distribution for class discovery can be created by incorporating a random uniform or unimodal simulation with the gene-gene correlation structure of the real data set. This can be achieved by forming a principal component score matrix from multivariate Gaussian distributions and multiplying it with the principal component loading vectors from the real data set (Materials and Methods 3.6.1).

We generated 50 null datasets from a Gaussian distribution having the same gene-gene correlation structure as **GBM1** and called this the *pcNormal* distribution. When needing to run one-to-one comparisons with **GBM1**, we chose a single representative data set from this distribution as explained in Materials and Methods (3.6.2) and denoted it with **Sim25**. This data set has known lack of structure, and hence an unbiased method should show no clusters. However, we observe in Figure 3.5 that CC shows stable clusters for tested values $K = \{2, 3, 4\}$. For a given K, the base clustering method k-means will result in some chance partitioning in the null data sets, and we see that CC is able to show apparent stability of



(a) $K = 2$ **(b)** $K = 3$ **(c)** $K = 4$ **(d)** $K = 5$

(e) $K = 2$ **(f)** $K = 3$ **(g)** $K = 4$ **(h)** $K = 5$

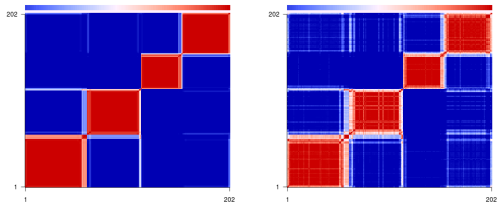
(i) $K = 2$ **(j)** $K = 3$ **(k)** $K = 4$ **(l)** $K = 5$

Figure 3.5: (a-d) Sample-sample correlation heatmaps, (e-h) 80% gene-subsampling consensus heatmaps, and (i-l) 80% sample-subsampling consensus heatmaps for Sim25. Sim25 is chosen as a characteristic random data set from the null distribution. For each K, the order of samples on all three heatmaps is the order obtained from average-linkage hierarchical clustering on the relevant gene-subsampling consensus matrix. The base clustering method for CC is k-means. CC shows stable clusters for $K = \{2, 3, 4\}$ even though the data set is known to lack structure.

such partitioning over repeated runs even though the data set is known to lack structure.

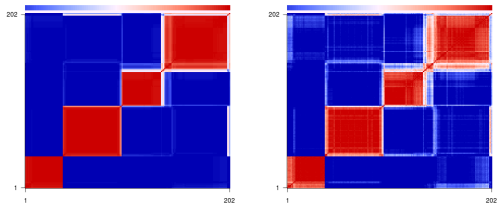
3.4.3 CC heatmaps of real datasets may be indistinguishable from the null distribution

We previously showed that the sample-sample Pearson's correlation coefficient matrix for **GBM1** shows weak sample clustering structure (Figure 3.1c). In other words, many samples do not clearly belong to one cluster and many have high correlations with samples from another cluster. However, the clustering signal for a potentially optimal K-value, $K = 4$, is much more 'crisp' in consensus heatmaps as seen in Figures 3.1a and 3.1b, and reproduced in Figures 3.6a and 3.6b. **Sim25** is a characteristic dataset among the 50 datasets in the pcNormal simulation, selected because its silhouette width is the median among the 50 datasets (Materials and Methods 3.6.2). Consensus clustering with the same



(a) GBM1 gene subsampling

(b) GBM1 sample subsampling



(c) Sim25 gene subsampling

(d) Sim25 sample subsampling

Figure 3.6: (a) GBM1 80% gene-subsampling consensus heatmap, (b) GBM1 80% sample-subsampling consensus heatmap. (c) Sim25 80% gene-subsampling consensus heatmap, (d) Sim25 80% sample-subsampling consensus heatmap. Even though Sim25 has known lack of structure, consensus heatmaps with $K = 4$ (a potentitally optimal value for GBM1) show almost as stable clusters as those of GBM1, making the clustering signal in GBM1 indistinguishable from the null distribution.

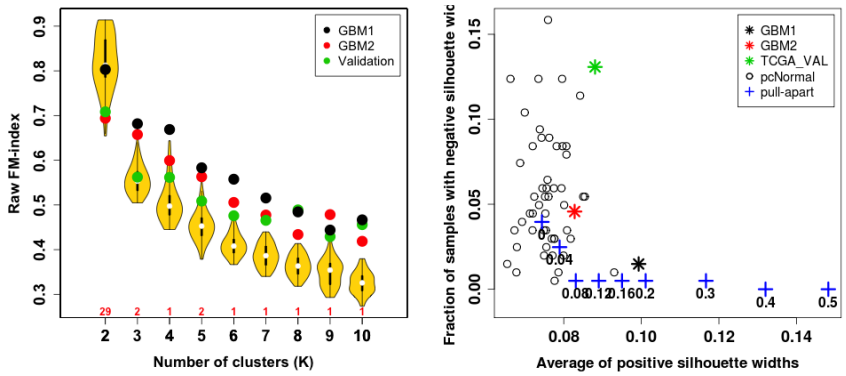
K value of 4 is able to generate apparently robust clusters for this null dataset (Figures 3.5g and 3.5k, and reproduced in 3.6c and 3.6d). This side-by-side comparison underlines the potential of using CC heatmaps to conclude structure when there is none, and/or to infer the number of classes in the dataset when there is very little evidence for it. It is also important to point out that this should not be taken to imply that any particular prior study has definitely over-stated their findings. Our results show that a null dataset could have generated nearly as strong CC-based evidence as presented in many prior studies. Whether a given study could be a true positive or a false positive requires quantitative assessment, as we will discuss in the rest of this paper.

3.4.4 CLEST and silhouette width results show that some aspects of GBM1 can be distinguished from those of the null distribution.

The results above suggest that CC indices, as a sample-sample distance matrix, can be over-sensitive in reporting apparently stable clusters. It is prudent not to solely rely on the visualization of the CC heatmap to estimate the presence or the number of clusters, but to base cluster inferences on quantitative summaries of the data. Two such inference tools are CLEST and Silhouette width. CLEST is a resampling-based method that randomly partitions the original dataset into a learning set $\mathcal{L}$ and a test set $\mathcal{T}$. The learning set is used to build a classifier for K clusters of the data. The classifier is then used to derive the *ground truth* partitions of the test set, and the quality of the partitions obtained by an unsupervised clustering algorithm is then assessed with the concordance between the two partitions. Silhouette width is a nonlinear combination of two internal clusteredness measures, namely compactness and separation of clusters (Materials and Methods 3.6.7.4). In the following, we apply these two methods to investigate the clustering signals in **GBM1** and compare them with those of the 50 *pcNormal* null datasets.

Figure 3.7a shows the CLEST results of **GBM1**, **GBM2**, and **Validation** over a range of K values along with the distributions of CLEST results for the *pcNormal* null datasets. We first observe that **GBM1** has higher FM values than **GBM2** and **Validation**, and these three real datasets had higher FM values that the simulated null datasets. Second, we see that $K = 4$ is not clearly optimal, but rather the FM-value differences are equally strong across $K = 2 - 8$. These results suggest that **GBM1** has more structure than null datasets, but CLEST does not confirm $K = 4$ as the optimal number of clusters.

Figure 3.7b shows a scatter plot of two statistics derived from **silhouette widths**. The first one, average of positive silhouette widths, is positively correlated with clustering signals and is shown on the x-axis. The fraction of negative silhouette widths, on the other hand, is negatively correlated with clustering signals and is shown on the y-axis. The clusters for this figure were obtained from a single k-means run with $K = 4$. We can see that **GBM1** is within the range of 50 *pcNormal* simulations along the y-axis, but is an outlier along the



(a) CLEST violin plot **(b)** Silhouette width scatter plot

Figure 3.7: (a) CLEST results for GBM1, GBM2, Validation, and *pcNormal* null datasets. (b) $K = 4$ silhouette width analysis for GBM1, GBM2, Validation, *pcNormal* null datasets, and pull-apart positive datasets. In (a), GBM1's FM-score from CLEST beats those of *pcNormal* simulations at as early as $K = 3$. In (b) the x-axis shows the average of positive silhouette widths is while the y-axis shows the fraction of negative silhouette widths. GBM1 is within the range of 50 *pcNormal* simulations (shown with hollow circles) along the y-axis. However, it appears as an outlier along the x-axis when compared with these null datasets. The pull-apart degree for positive datasets ranges from 0 to 0.5 (shown with blue pluses). Along the x-axis, GBM1 is close to the positive dataset with pull-apart degree 0.2.

x-axis. **GBM2** and **Validation**, on the other hand, are in the range of the simulations for both axes, which suggests weaker structure than **GBM1**. We also see in this figure that **GBM1** is most similar to the positive dataset simulated with pull-apart degree 0.2. This value is an indication of the strength of **GBM1**'s clustering signal with four clusters. Even though this value indicates more structure than null datasets, the implementation of silhouette scatter plots for other K values did not confirm four as the optimal value (data not shown). The simulation of positive datasets is explained further in Materials and Methods (3.6.4).

Different clustering measures emphasize different features of heterogeneous datasets such as compactness or separation. The average of all silhouette widths in a dataset, for instance, is strongly influenced by the existence of one or more highly compact clusters. Figure 3.1d demonstrates this effect by **GBM1**'s "protrusions" from the centroid towards two corners of the PC1-PC2-PC3 cubic space. This local density of protrusions in hyperspace are difficult to match by random simulations. These protrusions may be causing the observed difference

between **GBM1** and simulations in terms of the average positive silhouette width. However, this statistic alone would not be sufficient to determine whether the observed difference is due to real clustering structure or local densities.

The results in Figure 3.7 suggest that **GBM1** may indeed have a certain degree of clustering structure, and the over-interpretation potential of CC might be showing almost as stable clusters for the null dataset in Figures 3.6c and 3.6d. In the following, we will discuss the utility of CC as an inference tool for K, and analyze the different statistics derived from consensus matrices for their performance in accurately determining the number of clusters in the data, i.e. the optimal K value.

3.4.5 CC as an inference tool: The signal for optimal-K in CDF plots may not be visible in $\Delta(K)$ plots.

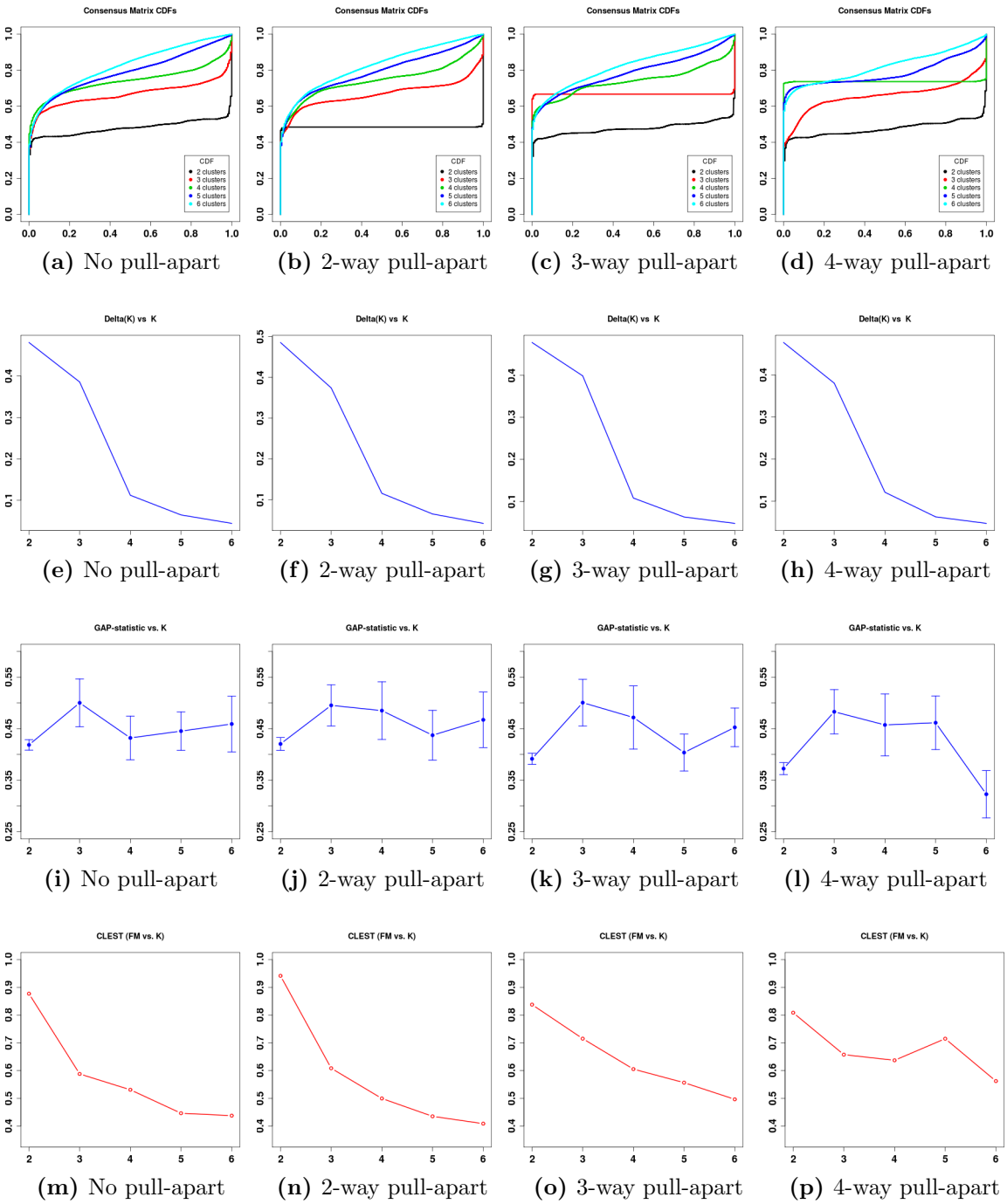
CC is a sensitive heuristic that has proven useful for visualizing cluster stability. However, the functionality and feasibility of the consensus matrix for inferring the optimal number of clusters in a dataset has not been sufficiently explored. Two statistics derived from the consensus matrix, namely the cumulative distribution function (CDF) and the proportional change in the area under the CDF curve ($\Delta(K)$) have been proposed as a way to estimate optimal K (explained in more detail in Materials and Methods). We investigated the informativeness of these statistics using simulated datasets having clusters of known separation. Real datasets may have poor separation between clusters, thus it is important to understand how different separation levels affect the outcome for CC-based inference and visualization, and to compare the sensitivity of different methods.

Starting from **Sim25**, we obtained K clusters using k-means and gradually increased the distance between clusters to obtain pulled-apart datasets. Separation among clusters was obtained by adding a known fraction of the cluster centroid to the principal component score matrix of the same cluster. The combined score matrix from all clusters was then multiplied by the principal component vectors of **Sim25** so that the pull-apart datasets have a similar gene-gene correlation structure as **Sim25**. We implemented this pulling apart procedure for $\{2, ..., 6\}$ clusters with separation degree a in $[0, 0.8]$, where 1 represent

the full magnitude of a cluster centroid.

Figure 3.8 shows CDF, $\Delta(K)$, GAP, and CLEST plots for $K = \{2, \ldots, 6\}$ from a randomly generated unimodal dataset (a,e,i,m), a 2-way pull-apart dataset with pull-apart degree 0.08 (b,f,j,n), a 3-way pull-apart dataset with pull-apart degree 0.12 (c,g,k,o), and a 4-way pull-apart dataset with pull-apart degree 0.12 (d,h,l,p). For pull-apart datasets, the CDF curve for the true K shows perfect or nearly perfect 0s and 1s in (b,c,d) while the unimodal dataset in (a) does not have such a perfect curve. In contrast to the clearly different curves in CDF plots, $\Delta(K)$ plots all look alike in (e-h) and suggest an optimal K value of 3 in all cases according to the elbow of the curves. GAP plots in (i-l) all suggest an optimal K value of 3 as that is the minimum number where the GAP-statistic is higher than the lower bound at the next K value. The lower bound is defined as the mean of the GAP-statistics across resamplings minus one standard deviation of the same values. The CLEST plots in (m-p) do not lend themselves to a direct interpretation of the optimal K value. The null simulations are needed to determine which K is most significant, however the shape of the curves suggest that the power to identify the true K at these separation levels is low as we do not see increases in FM values at the true K value.

The first lesson learned in this figure is that when the clusters of a dataset are sufficiently separated, the shape of the CDF curve will assume a three-step phase-transition form for the true K value and will make interpretations based on area unnecessary (a-d). The second lesson is that, area-based interpretations such as $\Delta(K)$ may be uninformative and potentially misleading even in the case of genuine structure (e-h). We note that the pull-apart datasets used here assume similar group size for all clusters and the same amount of pulling apart for all samples in a given cluster. In cases where group sizes are highly unbalanced or where samples on cluster boundaries are closer to one another, these assumptions may not hold. However, this limitation can be remedied by increasing the pull-apart degree for clusters.



(a) No pull-apart **(b)** 2-way pull-apart **(c)** 3-way pull-apart **(d)** 4-way pull-apart

(e) No pull-apart **(f)** 2-way pull-apart **(g)** 3-way pull-apart **(h)** 4-way pull-apart

(i) No pull-apart **(j)** 2-way pull-apart **(k)** 3-way pull-apart **(l)** 4-way pull-apart

(m) No pull-apart **(n)** 2-way pull-apart **(o)** 3-way pull-apart **(p)** 4-way pull-apart

Figure 3.8: a-d: CDF plots for (1) a randomly generated unimodal dataset, (2) a 2-way pull-apart dataset with degree of pull-apart = 0.08, (3) a 3-way pull-apart dataset with degree of pull-apart = 0.12, and (4) a 4-way pull-apart dataset with degree of pull-apart = 0.12. CDF curves for $K = \{2, \ldots, 6\}$ are shown with black, red, green, blue and cyan lines respectively. For pull-apart datasets (b,c,d), the CDF curve for true K shows perfect 0s and 1s while the unimodal dataset in (a) does not have such a curve. **e-h:** $\Delta(K)$ plots across $K = \{2, \ldots, 6\}$ for the corresponding datasets (1)-(4). Despite the variability in CDF plots, these plots all look alike and the elbow occurs at K=4 suggesting an optimal K value of 3. **i-l:** GAP plots across $K = \{2, \ldots, 6\}$ for the corresponding datasets (1)-(4). The optimal K value in the original interpretation is 3 in all four figures. **m-p:** CLEST plots across $K = \{2, \ldots, 6\}$ for the corresponding datasets (1)-(4). These plots do not suggest an optimal K interpretation directly, null simulations are needed to determine which K is most significant.

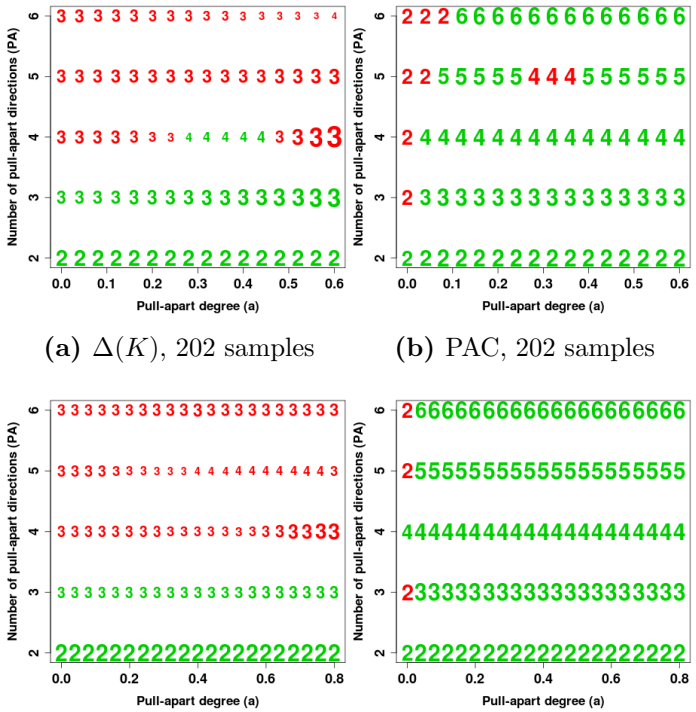
The predictions for the optimal K by a clustering method or measure can be plotted on a two-dimensional space with the true number of clusters on one axis and the pull-apart separation degree on the other. Showing true and false predictions in different colors would provide an easy way to understand the identifiability zone for that particular technique. One example of such an identifiability zone is given in Figure 3.9.

3.4.6 Identifiability zones show that a simple PAC measure performs best

$\Delta(K)$ *vs.* PAC: The CDF curves obtained from consensus matrices can be used as an inference tool in ways other than the $\Delta(K)$ method. One alternative is calculating the percentage of sample pairs that have a certain degree of ambiguity as to whether the pair should be assigned to the same cluster. Given a sub-interval $(u_1, u_2) \in [0, 1]$, the proportion of consensus matrix values in (u_1, u_2) is a measure of the probability of ambiguous clustering (PAC). A low PAC measure indicates a more rapid transition from an almost-perfect exclusion (sample pairs that are not in the same cluster) to an almost-perfect inclusion (sample pairs that are assigned to the same cluster). It also means that a there is a higher percentage of perfectly or nearly-perfectly clustered samples in that particular clustering scheme, hence showing the robustness of clusters. We observe in Figure 3.9 that the simple PAC measure performs substantially better than $\Delta(K)$ in correctly inferring the true number of clusters in pulled-apart data sets. The identifiability zone for PAC is also better than other clustering methods/validation measures we tested in this study such as CLEST, GAP-PC, and silhouettwe width (Figures 3.10-3.13).

3.4.7 Large sample size improves identifiability zones for CLEST and modified GAP-PC, but not for original GAP-PC or silhouette width

CLEST: The original CLEST interpretation for optimal K requires user-specified values p and d. The p value is the percentage of the null simulations having a greater score (such as FM) than the 'query' data set. This criterion alone is not stringent enough to declare an optimal K value because the absolute difference between the query data set and the simulations may be negligible. Therefore, the d value determines a threshold by which the



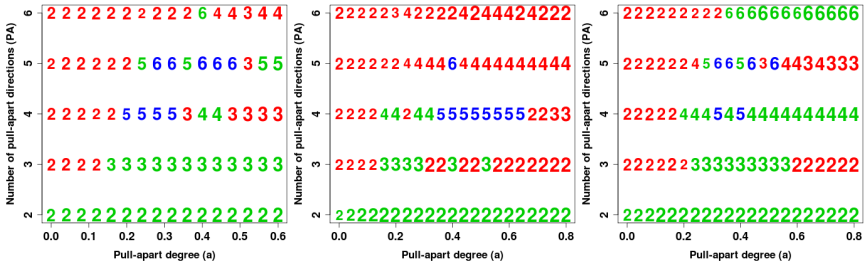
(a) $\Delta(K)$, 202 samples **(b)** PAC, 202 samples

(c) $\Delta(K)$, 1000 samples **(d)** PAC, 1000 samples

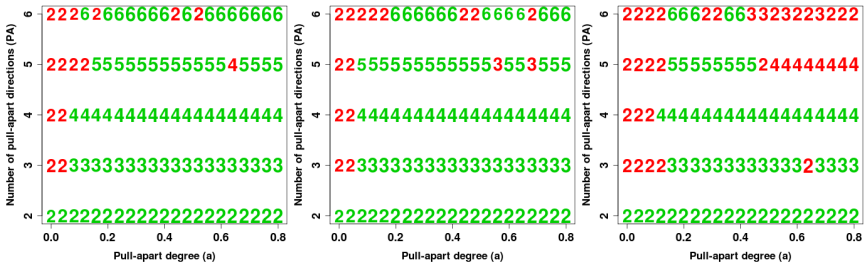
Figure 3.9: Identifiability zones for (a) $\Delta(K)$ and (b) PAC (probability of ambigous clustering). The x-axis shows the strength of the real pull-apart signal, denoted with a. The y-axis shows the true number of pull-apart clusters in the dataset, denoted with P. The numbers in the plots indicate the inferred optimal-K value for the particular (a, P) pair. The size of each plotting symbol is proportional to its respective score. The inferred optimal-K is correct if it is equal to P. The colors red, green and blue indicate underestimated, correctly estimated, and overestimated optimal-K respectively. The identifiability zone is defined as the collection of green symbols (correct estimates). The identifiability zone for PAC is observed to be better than $\Delta(K)$ and also other clustering methods we tested in this study (see Figures 3.10-3.13).

query data set has to be greater than simulations. If multiple K values meet these criteria, the K with maximum d value is chosen as the optimal K value. The pull-apart degrees where the optimal K from CLEST is the same as the true number of clusters can be viewed as the identifiability zone for the FM index of CLEST.

We observe in Figure 3.10 that the CLEST identifiability zone is rather narrow for 202 samples, but much wider for 1000 samples. Larger sample size allows more stable cross-validation. In the $N = 202$ case, the identifiability zone includes all 2-way pulled apart datasets regardless of the pull-apart degree. For 3-way pulled apart datasets, the structure



(a) 202 samples, pull-apart X **(b)** 202 samples, pull-apart Y **(c)** 202 samples, pull-apart Z

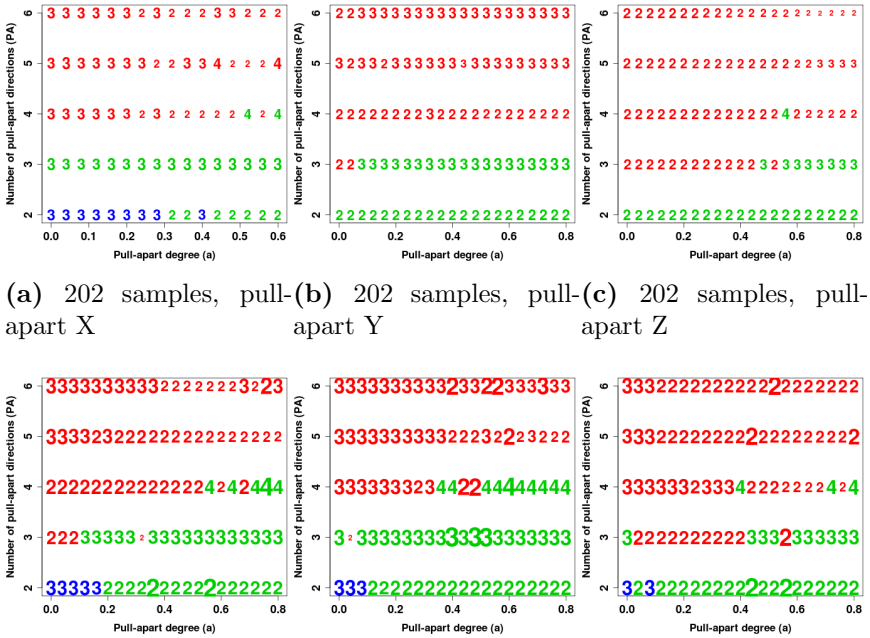


(d) 1000 samples, pull-apart X **(e)** 1000 samples, pull-apart Y **(f)** 1000 samples, pull-apart Z

Figure 3.10: CLEST identifiability zone: For each pull-apart degree (a) and pull-apart direction number (P), the external FM-index estimates the optimal K as the K value with highest score. We observe that the CLEST identifiability zone is rather narrow for 202 samples, but larger sample size (N=1000) allows more stable cross-validation (d,e,f).

signal is not picked up until after $a = 0.12$. For K values greater than three, CLEST exhibits a particularly poor identifiability zone. These results, interpreted together, may suggest that CLEST has a bias towards underestimating optimal K when clusters are not well-separated. The reason can be due to the external index measuring concordance between supervised clustering and unsupervised clustering; concordance may deteriorate at higher K values due to biases of supervised and unsupervised clustering methods chosen. When the number of samples is relatively small and K is high, there is a higher degree of bias associated with each cluster because there are fewer number of samples to assign to each cluster. In such cases with higher bias, cross-validated clusters may have lower concordance with the original unsupervised clusters.

GAP-PC : The GAP-statistic provides an estimate for the number of clusters in a dataset by comparing the change in within-cluster dispersion with that expected under



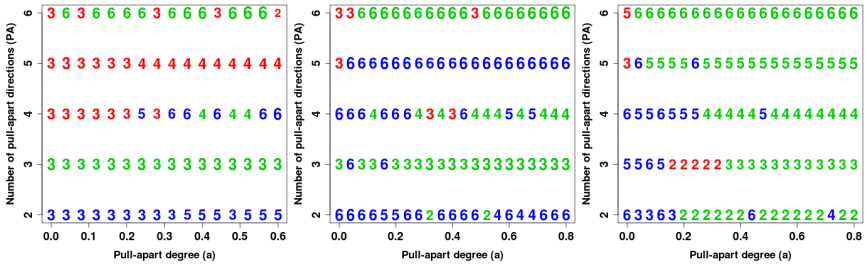
(a) 202 samples, pull-apart X **(b)** 202 samples, pull-apart Y **(c)** 202 samples, pull-apart Z

(d) 1000 samples, pull-apart X **(e)** 1000 samples, pull-apart Y **(f)** 1000 samples, pull-apart Z

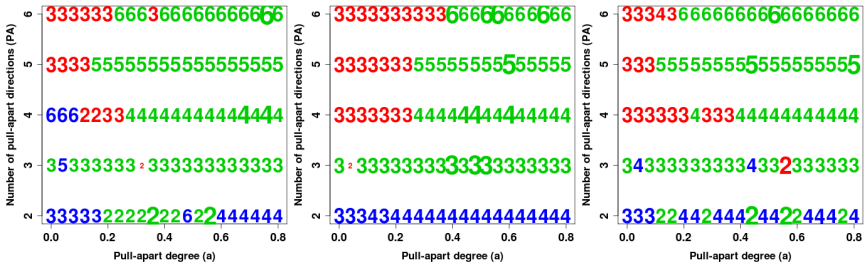
Figure 3.11: GAP-PC identifiability zone for (a-c) 202 samples, (d-f) 1000 samples. There is variability in the identifiability zones of the three pull-apart sets. Overall, the performance for pull-apart number 2 or 3 may be good, but those for greater than 3 are particularly poor. Large sample size does not help much towards stabilizing the identifiability zones.

an appropriate reference null distribution. The authors suggest in (9) that the optimal K is the smallest K where the mean GAP score is larger than the lower bound for K+1; where the lower bound is defined as the mean GAP score minus the standard error for that particular K value. In Figure 3.11, we present the identifiability zones for this technique on pull-apart data sets based on three different *pcNormal* simulations. For both 202 and 1000 samples, the performance for 2 or 3-way pull-apart data sets is good in at least one of the pull-apart sets X, Y, and Z. However, performance for greater pull apart numbers is particularly poor. This points at the "smallest K" criterion in the interpretation of GAP-PC results. Even though GAP-PC scores for K values greater than 3 may be showing significant structure in the data, the relative comparison with 2 and 3 might be rendering them meaningless if the scores for 2 and 3 are artificially higher.

To test this idea, we modified the decision rule in GAP-PC and set the K value with



(a) 202 samples, pull-apart X **(b)** 202 samples, pull-apart Y **(c)** 202 samples, pull-apart Z

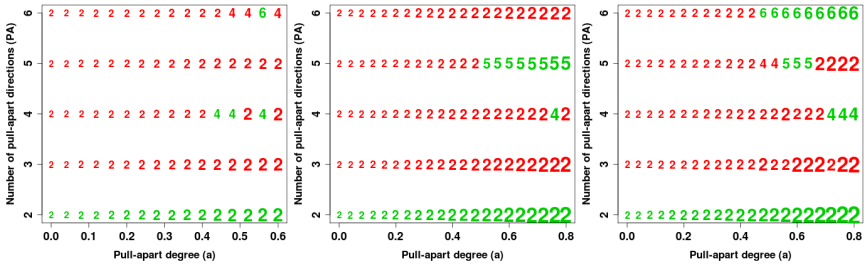


(d) 1000 samples, pull-apart X **(e)** 1000 samples, pull-apart Y **(f)** 1000 samples, pull-apart Z

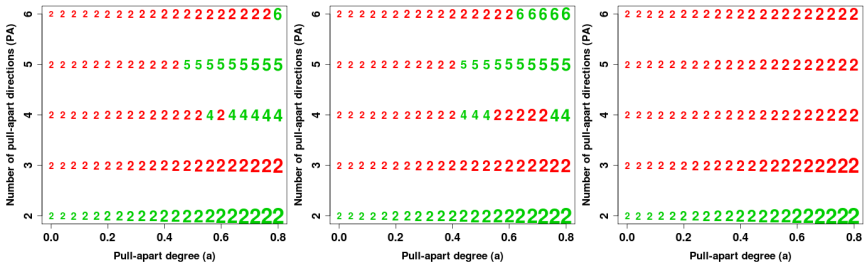
Figure 3.12: GAP-PC identifiability zone with modified decision rule for (a-c) 202 samples, (d-f) 1000 samples. Identifiability zones for 202 samples are better than those of the original GAP-PC decision rule, but a weakness emerges for determining optimal K when the true pull-apart number is 2. Large sample size improves identifiability zones this time. Identifiability zones for 1000 samples are both wide and stable.

the highest score as the optimal number of clusters. This decision rule is more intuitive and straigtforward. Indeed, we see in Figure 3.12 that the identifiability zones with this decision rule are much improved compared with the original GAP-PC decision rule. We observe a wider identifiability zone for both 202 and 1000 samples. Moreover, the stability of identifiability zones for 1000 samples is strikingly strong. The only particular weakness of this technique appears in the results for pull-apart number 2. This decision rule has difficulty in correctly identifying optimal K with 202 an 1000 samples for 2-way pulled-apart data sets.

Silhouette width : The K with highest average silhouette width across different K values is commonly accepted as the optimal K value. We see in Figure 3.13 that the identifiability zones for the silhouette width only cover pull-apart number 2 with high confidence.



(a) 202 samples, pull-apart X **(b)** 202 samples, pull-apart Y **(c)** 202 samples, pull-apart Z



(d) 1000 samples, pull-apart X **(e)** 1000 samples, pull-apart Y **(f)** 1000 samples, pull-apart Z

Figure 3.13: Silhouette width identifiability zone for (a-c) 202 samples, (d-f) 1000 samples: For each pull-apart degree (a) and pull-apart direction number, the average silhouette width across samples estimates the optimal K as the K value with highest score. Identifiability zones are robust for only $K = 2$ regardless of the number of samples. Large sample size does not improve identifiability zones.

Interestingly, large sample size ($N = 1000$) does not improve the identifiability zones. This might be due to $K = 2$ average silhouette width values increasing linearly with pull-apart degree a, and tending to be higher than average silhouette width for all other K values regardless of the pull-apart degree (Figure 3.27 in Supplementary Materials).

3.4.8 Gene-gene correlation makes it easy to "validate" ANY K by most discriminant genes

After an optimal-K estimate is determined for a dataset, the next goal is to validate these clusters. As we mentioned earlier, cluster validation in unsupervised class discovery is elusive due to the lack of external information such as class labels with which one can obtain error rates for the estimates/predictions. Thus, some methods using independent data have been employed to make an attempt at validating clusters. One method that has

become highly popular is to determine the most discriminant genes for a certain K value from the original dataset, and then use these to classify samples in an independent dataset. We next show that the gene-gene correlation structure in genomic datasets may indeed confound this validation approach.

In a popular implementation (5; 24), the best classifiers for each cluster are taken from the learning set, and a heatmap of all samples with only the best classifiers is constructed. Next, a heatmap is constructed for the independent dataset with the same genes. Observing a similar placement of discrete genesets in the latter heatmap as in the former is considered a validation of the number of classes found in the first dataset.

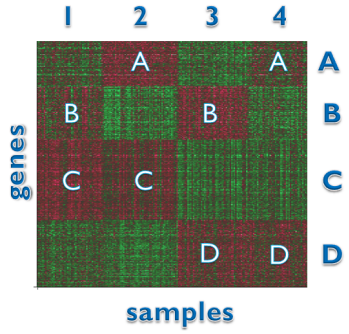
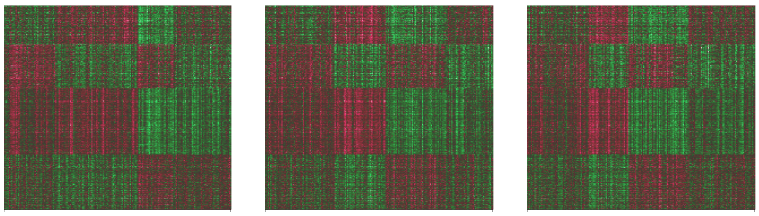
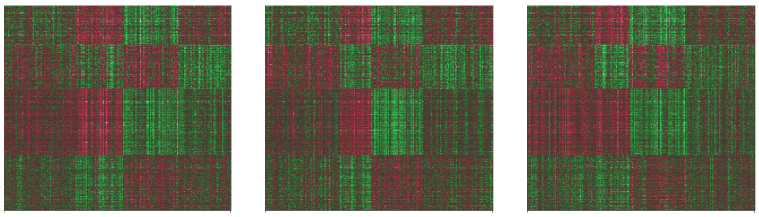


Figure 3.14: **Sim25** is a simulated random dataset with gene-gene correlation from **GBM1**. It is representative for a collection of 50 similarly simulated datasets in terms of silhouette width statistics. The heatmap above shows the most discrimant genes (groups $A - D$) for clusters $1 - 4$.

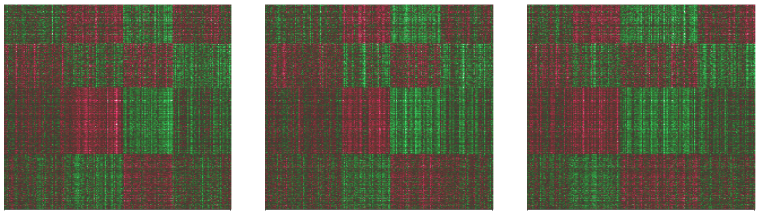
We used the *pcNormal* null simulations to test the hypothesis that gene-gene correlation could lead to a similar placement in the second heatmap even when the second dataset is unimodal. We first run k-means on **Sim25** samples with $K = 4$ to obtain four classes. Following the procedure in (3), we find the top discriminating genes for each class by computing the t-scores for each cluster against the other clusters, and finding the 4 sets of 210 genes with largest absolute t-scores. Next, we merge these 4 sets of 210 genes resulting in 551 unique genes due to some shared genes in different gene sets. The heatmap of 551 genes and 202 samples is seen in Figure 3.14. Discrete placement of gene sets similar to the ones in this figure have been viewed in genomics as a validation of the existence of four classes. However, when we take 9 other similarly simulated datasets, cluster the samples, and plot the heatmaps with the same 551 genes, we are able to observe the same gene



(a) pcNormal8 **(b)** pcNormal36 **(c)** pcNormal39



(d) pcNormal42 **(e)** pcNormal43 **(f)** pcNormal44



(g) pcNormal45 **(h)** pcNormal48 **(i)** pcNormal50

Figure 3.15: Heatmaps for 9 similarly simulated datasets as **Sim25**. The x-axis shows samples as partitioned into 4 clusters with k-means, and the y-axis shows the same 'most discriminant genes' from **Sim25**. The gene signature from **Sim25** is preserved even when the new data are random.

signature in all 9 cases (Figure 3.15a-3.15i). These 9 datasets are chosen from the entire range of 50 simulations so as to represent the spectrum of small to large silhouette widths in the null distribution (details given in Materials and Methods 3.6.3).

Gene-gene correlations give rise to blocks of genes in subsets of samples by virtue of persistent co-regulation. If mistaken as part of a prognostic signature, these blocks will reappear in independent datasets as evidence for subtypes. However, the results in Figure 3.15 indicate that using such top genes can easily yield the "discovery" of the same number of clusters even with unimodal data. It should be remembered that the correlations among genes can influence the results for both number of clusters and the assignment of samples into these clusters. Thus, the gene-gene correlation structure has to be accounted for in cluster validation methods.

We next show that the choice of the base clustering method can substantially change the results in CC, and k-means is a suitable choice in terms of both reliability and efficiency.

3.4.9 K-means is both robust and efficient as a base method for consensus clustering

The base clustering method in CC is another parameter of the algorithm that can be changed. Hierarchical clustering (HCLUST) with average-linkage has become popular in most applications, but here we underline the drawbacks of using this base method as it is not robust against outliers and has a strong tendency to lock in accidental features of the data. Outside the CC context, HCLUST with average-linkage has also been used widely (3; 11; 19), however complete-linkage has been shown to perform better for non-ratio based expression values (20), and it produced a much more stable clustering compared with average linkage in (21). Our results in Figures 3.16-3.18 confirm this finding.

Here, we compare 5 clustering methods; namely k-means, hierarchical clustering (HCLUST) with average and complete linkage, model-based clustering (MCLUST), and k-medoids, also known as partitioning around medoids (PAM). Each base method was run 500 times on **Sim25** (except MCLUST, which was run 100 times due to its high computational cost) with K=4 and 80% sample subsampling ratio. The consensus partitioning for each method was obtained by clustering the respective consensus matrix into four groups with average-linkage HCLUST after the subsample runs were completed.

The concordance of the k-means consensus partitioning with the partitioning from the other four base methods is shown in Figure 3.16. The values on the diagonal shows the level of concordance between the methods on the x and y axes, i.e. the number of samples assigned to the same cluster by these methods. Off-diagonal values, on the other hand, show the level of confusion as they indicate the number of samples assigned to different clusters by the two methods. Summing the rows in any one of the four panels, we see that k-means consensus run divides the 202 samples into two larger and two smaller clusters of 61, 66, 38 and 37 samples respectively. The magnitude of the values on the diagonals indicate that k-means has the highest concordance with MCLUST, and the lowest with HCLUST

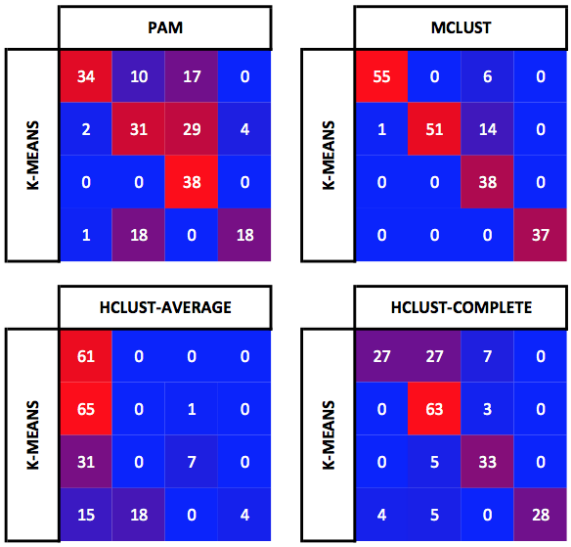


Figure 3.16: Sim25 confusion matrices between consensus runs of K-means and those of PAM, MCLUST, HCLUST average and complete-linkage.

average-linkage. It can be seen that the $K = 4$ consensus run for HCLUST average-linkage groups 172 samples (85% of 202) in a single cluster, which underlines the sensitivity of the average-linkage method to outliers. Figure 3.17 shows that HCLUST average-linkage does not have good concordance with the other three base methods either.

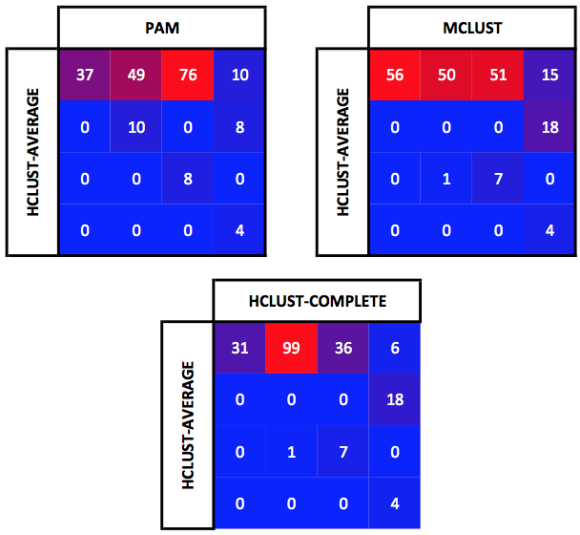


Figure 3.17: Sim25 confusion matrices between consensus runs of HCLUST average-linkage and those of PAM, MCLUST, HCLUST complete-linkage.

Figure 3.18 shows the agreements and disagreements between the consensus runs of other base methods. None of these matrices show better concordance between two methods than

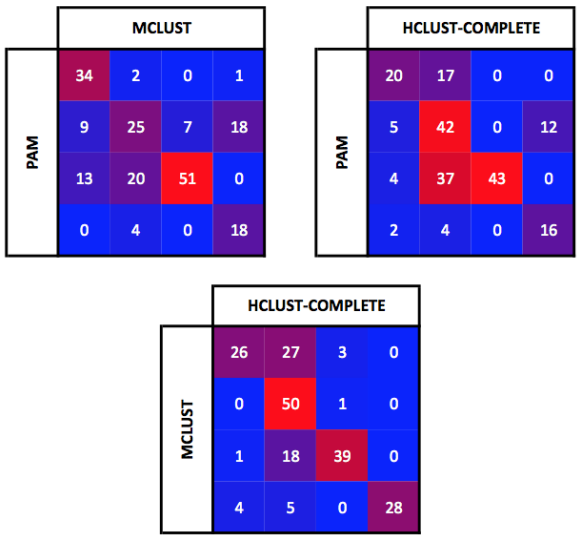


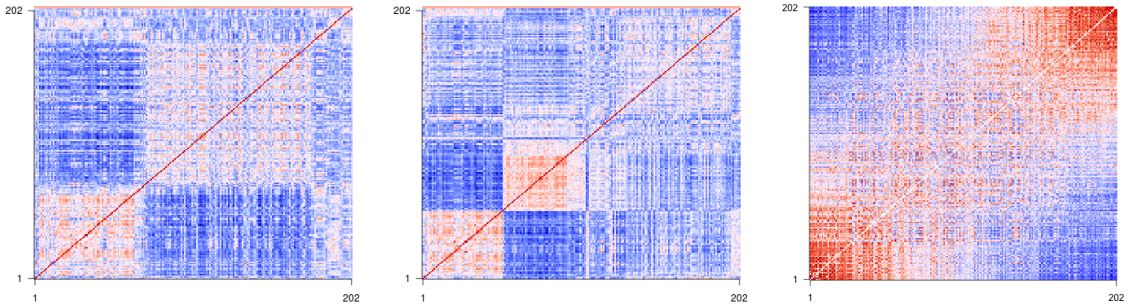
Figure 3.18: Other Sim25 confusion matrices between consensus runs of PAM, MCLUST, HCLUST complete-linkage.

the concordance between k-means and MCLUST shown in Figure 3.16. Thus, k-means and MCLUST appear as the most competitive choices for clustering on this particular dataset (**Sim25**). However, MCLUST is much less feasible to be used in a class discovery pipeline including consensus clustering due to its computational cost. Hence, we recommend the use of k-means as the base clustering method in CC as a both reliable and efficient tool.

3.4.10 Lessons learned (DOs and DON'Ts in class discovery)

Sample-sample correlation and consensus matrices can be prone to over-interpretation in the context of unsupervised class discovery. One example to the former is the set of three correlation heatmaps in Figure 3.19. In this figure, the same matrix is re-ordered differently according to the average-linkage hierarchical clustering order on the $K = 2$ and $K = 3$ consensus matrices and also the order from decreasing PC1 scores. We see that it is easy to re-order a sample-sample correlation matrix in different ways to support different hypotheses regarding the structuredness of the data.

In the broader context, we summarize our findings by stating that many clustering methods are able to divide structureless data into prespecified numbers of clusters, and CC is able to show apparent stability of such chance partitioning of random data. Thus, we



(a) Order from K=2 consensus matrix

(b) Order from K=3 consensus matrix

(c) Order from decreasing PC1 scores

Figure 3.19: GBM1 sample-sample correlation matrix ordered three different ways. The order of samples is obtained from (a) average-linkage HCLUST on the $K = 2$ gene subsampling consensus matrix (b) average-linkage HCLUST on the $K = 3$ gene subsampling consensus matrix (c) decreasing PC1 scores. It is easy to re-order the same sample-sample matrix in different ways to support different hypotheses about structuredness.

make the following recommendations for unsupervised class discovery and cluster validation.

DON'T rely on the visualization of the consensus matrix to declare the existence of a clustering structure, or to estimate optimal K. Probability of ambiguous clustering (PAC) obtained from the consensus matrix, on the other hand, is a simple yet powerful method to infer optimal K under certain conditions.

DO choose your base clustering method carefully as not all methods are equally robust. We observed that k-means is both reliable and efficient.

DO a formal test of cluster validity using simulated random unimodal data with the same gene-gene correlation, and compare clustering signal strength.

DON'T use the most discriminant genes for K clusters to validate K in a new dataset.

DO bear in mind that many clustering methods and measures are sensitive to sample size.

3.5 Conclusions

We applied consensus clustering to random data and observed that many unsupervised partitioning methods are able to divide unimodal data into prespecified numbers of clusters;

and consensus clustering shows apparent stability of such *chance partitioning of random data*.

We compared **GBM1** with random data having the same gene-gene correlation structure, and observed that consensus clustering heatmaps and summary statistics for **GBM1** were often within the empirical null distribution. However, other methods such as CLEST and silhouette width were able to distinguish **GBM1** from the null datasets likely because of the local clusters or outliers in **GBM1**. We observed that the clustering signal in the null distribution was exaggerated by consensus clustering so as to show stable clusters. Therefore, we caution against relying on consensus clustering to declare the existence of clusters or to estimate the optimal number of clusters.

There is a variety of clustering methods and validation measures. While each one of these may have its strengths and weaknesses, it is important to understand in particular whether the technique performs well on datasets with poorly separated clusters. It is common to encounter genomic datasets where clusters have poor separation, so it would not be appropriate to blindly adapt a clustering technique and draw simple yes-no conclusions. We saw that some of the more popular clustering methods/validation measures can be internally inconsistent, and/or have poor correlation with others if the clusters are poorly separated.

We also investigated the properties of the consensus matrix and its derived statistics using Gaussian-mixture datasets having incrementally increasing clustering signals and realistic gene-gene correlation structure. We found that the consensus matrix by itself is not a suitable inference tool, but its distribution features can be. One such feature, the proportion of ambiguously clustered (PAC) pairs, reflected the true structure of the data better than common strategies such as CDF, $\Delta(K)$, Silhouette Width, GAP-PC, and in most situations, CLEST. We also showed by using synthetic null datasets that validation of clusters with most discriminant genes can be highly error-prone as cluster structure is partly driven by the gene-gene correlations in the data. The limitations of our simulations include the following assumptions: (1) Samples in cluster boundaries are assigned to a single cluster, no partial memberships are used, (2) clusters are viewed as co-equal without any

nestedness, and (3) clusters are simulated with similar size, no outliers are included.

In summary, our results suggest that consensus clustering needs to be applied with caution as it is prone to over-interpretation. If samples are not well-separated, consensus clustering could lead one to conclude apparent structure when there is none, or declare cluster stability when it is subtle. In the presence of genuine structure, this method may be a powerful tool for identifying clusters, but in the exploratory phase of many studies can lead to false positives if it is not compared to a suitably formed null distribution. When partitioning poorly separated samples from a real dataset, it is necessary to objectively evaluate the evidence of clustering, and if appropriate, consider alternative models such as nested models, partial membership models or continuous distributions.

3.6 Materials and Methods

3.6.1 Generating a 'null' distribution for unsupervised class discovery

The findings from Figure 3.1 motivate the comparison of **GBM1**'s cluster signals with those in randomly simulated 'null' data sets to assess significance. It is curious to ask whether the weakness of the clustering signal revealed in these figures is severe enough to preclude **GBM1** from surpassing significance thresholds for cluster stability and existence. To investigate this question, we generated a collection of 50 random data sets that could function as a null distribution for the clustering signal in **GBM1**. Since consensus clustering involves running a base clustering method on a data set hundreds of times, this method can become computationally too expensive with a larger number of random data sets. Each data set in the collection, referred to as a *pcNormal* simulation, forms an ellipsoid in high-dimensional space. This set is generated by multiplying GBM1's PC vectors on the left by random Normal PC scores.

1. Using principal component analysis, we obtain the orthogonal matrix $\mathcal{A}$ of **GBM1** eigenvectors.

$$Y_{202\times 202} = GBM1_{202\times 1740} \times \mathcal{A}_{1740\times 202}$$

In this notation, Y is the 'score' matrix for **GBM1**.

2. Next, we simulate a random matrix $Y^{\mathcal{N}}$ where column i is distributed normally with zero mean and standard deviation equal to that of column i in Y.

$$Y_{.i}^{\mathcal{N}} \sim \mathcal{N}(0, s_i) \tag{3.1}$$

where s_i is the standard deviation of $Y_{.i}$ and $i = \{1, \ldots, 202\}$.

3. Multiplying $Y^{\mathcal{N}}$ with $\mathcal{A}^T$ yields one of the *pcNormal* simulations.

$$Q_{202\times1740}^{\mathcal{N}} = Y_{202\times202}^{\mathcal{N}} \times \mathcal{A}_{202\times1740}^{T} \tag{3.2}$$

4. We repeat this procedure 50 times to obtain a population of 50 *pcNormal* simulations.

$$(Q^{\mathcal{N}})^j = (Y^{\mathcal{N}})^j \times \mathcal{A}^T, j = \{1, \ldots, 50\}$$

3.6.2 Choosing a representative 'null' data set

A representative dataset among **pcNormal** simulations is chosen from a collection of 50 similarly simulated sets. For simplicity, this dataset is referred to as **Sim25**. More specifically, the criteria for **Sim25** is having representative values for certain clustering measures (namely average of positive silhouette widths, and the percentage of negative silhouette widths). The formulas followed are given below.

$$\begin{aligned} SW^n &= \text{percentage of negative silhouette widths} \\ SW^p &= \text{average of positive silhouette widths} \\ median^{p,a} &= [median(SW^n), median(SW^p)] \\ Sim25 &= \underset{i}{\text{argmin}} \quad d(median^{n,p}, [SW_i^n, SW_i^p]) \end{aligned} \tag{3.3}$$

where $[SW_i^n, SW_i^p]$ is the vector of silhouette width statistics for simulation $i \in \{1, \ldots, 50\}$, and d is the Euclidean distance function.

3.6.3 Choosing nine *pcNormal* simulations for validation by most discriminant genes

The median values across 50 *pcNormal* data sets for the average positive silhouette width (sw_p) and the fraction of negative silhouette widths (sw_n) were found. The Euclidean distance of the (sw_p,sw_n) pair from each data set to the median values is computed for all 50 simulations and ranked from lowest to highest. Nne simulations were then chosen as every 5th highest-ranking dataset. In other words, datasets with ranks $\{6, 11, 16, 21, 26, 31, 36, 41, 46\}$ were chosen. This ensures that the entire range for the silhouette width statistics is represented.

3.6.4 Generating a 'positive' distribution for unsupervised class discovery

We observe in Materials and Methods section 3.6.7 that six different measures of clustering do not agree on whether **GBM1** has more structure than randomly simulated datasets, or whether it can be within these null distributions. One reason that would bring about this apparent lack of concordance is the degree at which different methods can measure compactness and separation of clusters when cluster boundaries are not sharp. To be able to explore the behavior of different methods with varying degrees of boundary sharpness, we simulated positive datasets with increasing separation.

We typically start with a *pcNormal* simulation such as **Sim25** and execute the following steps:

1. Using principal component analysis, obtain the orthogonal matrix $\mathcal{A}$ of **Sim25** principal component vectors.

$$Y_{202\times202} = Sim25_{202\times1740} \times \mathcal{A}_{1740\times202}$$

   In this notation, Y is the 'score' matrix for **Sim25**.

2. Using a partitioning method such as k-means, assign each sample s_i $(i = 1, ..., 202)$ into one of K classes. The set of samples in class k $(k = 1, ..., K)$ is denoted with E_k.

3. For each class k, compute the sample-centroid C_k of Y_{E_k}.

4. For each class k and for a given pull-apart degree a, compute pulled-apart score matrix $Y^p_{E_k}$

$$Y^p_{E_k} = Y_{E_k} + a * C_k$$

5. Multiply Y^p with $\mathcal{A}$ to obtain the pulled-apart dataset X^p.

$$X^p_{202\times1740} = Y^p_{202\times202} \times \mathcal{A}^T_{202\times1740}$$

3.6.5 Generating the $Circle1$ and $Square1$ simulations

Circle1: We generated 400 samples with error perturbation, 1000 genes each, in a 20×20 square grid formation. Samples occupying the four corners in the two-dimensional PC-space were trimmed to leave a circular topology of ~ 300 samples. As each side of the grid measured 20 units, the corners were about 14 units away from the center, and the radius was 9.62 units.

Square1: To investigate the potential anchoring effect by the four corners of the dataset, we generated 100 samples with error perturbation, again 1000 genes each, arranged in a 10×10 square grid on the PC1-PC2 plot.

3.6.6 Base methods for consensus clustering

3.6.6.1 Hierarchical clustering (HC)

This is one of the most widely used partitioning algorithms, due to its conceptual and practical simplicity. Agglomerative HC (12) starts by considering the n data points as n clusters, and at each iterative stage, the closest pair of clusters is joined to form a new cluster. Divisive HC starts by considering a unique cluster, and at each iterative step a cluster is divided into two. In both methods, a hierarchical tree is constructed of all data points after $n - 1$ steps.

Initially, a distance or dissimilarity matrix of all pairs of points must be computed using some distance function (Euclidean, Minkowski, Manhattan, Mahalanobis, etc.). To define the distance $D(r, s)$ between clusters r and s, different linkages such as the single, complete, or average linkage may be chosen.

In **complete linkage**, $D(r, s)$ is defined as the distance between the most distant pair of objects, one from each cluster.

$$D(r, s) = \max\{d(i, j); \text{ object } i \text{ is in cluster } r \text{ and object } j \text{ is in cluster } s\}$$

At each stage of hierarchical clustering, the clusters r and s , for which $D(r, s)$ is minimum, are merged. **Single linkage** is the opposite of complete linkage in the sense that the maximum distance is replaced with the minumum distance.

In **average linkage**, $D(r, s)$ is defined as the average of distances between all pairs of objects, where each pair is made up of one object from each group.

$$D(r, s) = \frac{T_{rs}}{(N_r \times N_s)}$$

where T_{rs} is the sum of all pairwise distances between cluster r and cluster s. N_r and N_s are the sizes of the clusters r and s respectively. At each stage of hierarchical clustering, the clusters r and s , for which $D(r, s)$ is the minimum, are merged.

3.6.6.2 k-means

Given a set of observations $(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n})$ where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k sets ($k \leq n$), $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the within-cluster dispersion:

$$\underset{\mathbf{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\mathbf{x_j} \in S_i} ||\mathbf{x}_j - \mu_i||^2$$

where μ_i is the mean of points in S_i (13).

The method starts with k arbitrary cluster centers. Each step consists of labeling data

points with their nearest cluster center, and updating the centers of the new clusters. The procedure stops when the clusters formed at two consecutive steps are the same.

3.6.6.3 k-medoids

Proposed by Kaufman and Rousseeuw (14), k-medoids is a more flexible version of k-means. The within-cluster dispersion to be minimized is modified as

$$\sum_{i=1}^{k} \sum_{\mathbf{x_j} \in S_i} d(\mathbf{x}_j - \tilde{x}^{(i)})$$

where $\tilde{x}^{(i)}$ is the medoid of S_i, and $d(\cdot, \cdot)$ can be any dissimilarity measure. A medoid can be defined as the object of a cluster, whose average dissimilarity to all the objects in the cluster is minimal, i.e it is a most centrally located point in the cluster.

3.6.6.4 Model-based clustering (MCLUST)

In this approach, data points in each cluster are represented by a Gaussian probability distribution component of a finite mixture model (15; 16). Let $\theta_j = (\mu_j, \Sigma_j)$ be the parameter associated with the probability distribution $f_j(x; \theta_j)$, where μ_j is the center and Σ_j is the covariance structure of cluster j. If p_j denotes the probability that an observation belongs to the j-th cluster, then the classification likelihood of the n independent observations $x_1, x_2, \ldots, x_n$ will be given by:

$$L(\theta, p|x) = \prod_{i=1}^{n} \left[\sum_{j=1}^{K} p_j f_j(x_i; \theta_j) \right]$$

where $x = (x_1, x_2, \ldots, x_n)$ is the dataset, $\theta = (\theta_1, \theta_2, \ldots, \theta_K)$ is the parameter vector and K is the number of clusters. The parameters can then be estimated by maximizing the classification likelihood through the EM algorithm. Bayesian Information Criterion (BIC) is used to select the number of clusters K and the complexity of Σ_j.

3.6.7 6 ways to measure clustering signals

3.6.7.1 Cluster stability via empirical CDF plots

For a given consensus matrix M, the corresponding empirical cumulative distribution (CDF) can be defined over the range $[0, 1]$ as follows:

$$CDF(c) = \frac{\sum_{i<j} \mathbf{1}\{\mathcal{M}(i,j) \leq c\}}{N(N-1)/2}$$

where $\mathbf{1}\{\dots\}$ denotes the indicator function, $\mathcal{M}(i,j)$ denotes entry (i,j) of the consensus matrix $\mathcal{M}$, N is the number of rows (and columns) of $\mathcal{M}$, and c is the consensus index value (11).

3.6.7.2 Optimal K via proportional area change under CDF ($\Delta(K)$)

CDF curves are an important tool in investigating the optimal number of clusters for a dataset. CDFs shape and progression as K increases provide evidence for the appropriate number of clusters. A shape that more closely resembles a three-phase step function as mentioned before is indicative of a higher cluster stability. The objective in inspecting CDF progression is to select the largest K that induces a large enough increase in the area under the corresponding CDF (11). This progression can be quantitatively analyzed by measuring the area under the curves. The area under the CDF corresponding to $A(K)$ is given by the formula:

$$A(K) = \sum [x_i - x_{i-1}] CDF(x_i)$$

The progression, in turn, can be visualized by plotting the proportion increase $\Delta(K)$ in the CDF area as K increases. $\Delta(K)$ is computed as follows:

$$\Delta(K) = \begin{cases} A(K) & \text{if } K = 2 \\ \dfrac{[A(K) - A(K-1)]}{A(K-1)} & \text{if } K > 2 \end{cases}$$

3.6.7.3 Variance explained by principal components

The fact that proportion increases in CDF area ($\Delta(K)$ values) for real datasets can be placed within the distribution of simulations lacking sample structure brings about the question of whether the percentage of variance explained by principal components would differ between real and simulated datasets. Simulated datasets with known lack of structure would distribute both gene and sample variance almost evenly to all principal components; the decrease from the first principal component to the second would be very small (figure not shown). However, it is not straightforward how simulated datasets with known lack of sample structure but having the same gene-gene correlation structure as a real dataset, i.e. **GBM1**, would compare with the real dataset in terms of how much sample variance is carried by principal components. We plotted percentage of variance explained by first 10 principal components in real and simulated datasets to investigate this. The results are shown in Figure 3.21c in Supplementary Materials.

3.6.7.4 Silhouette width

The silhouette validation method (8) could be applied for evaluation of clustering validity and also could be used to decide how good the number of selected clusters is. Silhouettes are constructed in the following way. Consider any object i of the data set, and let A denote the cluster to which it is assigned, and then calculate

$$a(i) = \text{average dissimilarity of i to all other objects of A}$$

Now consider any cluster C different from A and define

$$d(i, C) = \text{average dissimilarity of } i \text{ to all objects of } C$$

Compute $d(i, C)$ for all clusters $C \neq A$, and then select the smallest of those.

$$b = \min_{C \neq A} d(i, C)$$

Let B denote the cluster which attains the minimum i.e., $d(i, B) = b(i)$ is called the neighbor of object i. The value $S(i)$ can now be defined as:

$$S(i) = \frac{(b(i) - a(i))}{\max\{a(i), b(i)\}}$$

It can be easily seen that $S(i)$ lies between -1 and +1.

We chose to compare **GBM1** with simulations according to two statistics derived from silhouette widths. One is the percentage of samples with negative silhouette widths. Negative silhouette widths indicate samples that are likely to be assigned to the wrong cluster by the partitioning algorithm. Hence, more samples with negative silhouette width means poorer clusteredness in the dataset. The second statistic we used was the average of positive silhouette widths. Higher silhouette widths are indicative of better clustering, so a higher average of positive silhouette widths show better clusteredness among samples.

3.6.7.5 Optimal K via GAP-statistics

The GAP-statistic provides an estimate for the number of clusters in a dataset by comparing the change in within-cluster dispersion with that expected under an appropriate reference null distribution (9). It is known to perform well at identifying well-separated clusters. However, when data are not well separated, the error rate increases approximately linearly with the proportion of overlapping points between two clusters, *i.e.* the GAP-statistic will choose $K = 1$ (as the optimal value) $p\%$ of the time for a $p\%$ proportion of overlapping points.

3.6.7.6 Optimal K via CLEST

CLEST measures the concordance in cross-validation between a supervised classification method (such as $DLDA$) and an unsupervised clustering method (such as k-means).

3.6.8 Datasets

3.6.8.1 Glioblastoma multiforme

- **GBM1**

  Glioblastoma is the first cancer type studied by the TCGA consortium. Primary glioblastoma arises *de novo* without antecedent history of low-grade disease, whereas secondary glioblastoma progresses from previously diagnosed low-grade gliomas (1). In order to exclude the confounding effects of secondary glioblastoma in data analysis, biospecimen repositories were screened for newly diagnosed glioblastoma for the purpose of this study.

  After quality control measurements, 185 newly diagnosed biospecimen qualified for copy number, gene expression and DNA methylation analyses (2). The first cohort also included 21 post-treatment cases to be used for exploratory comparisons, bringing the total number of samples to 206. Although it is possible that a small number of progressive secondary glioblastomas were in the cohort, 185 newly diagnosed glioblastomas represent predominantly primary glioblastoma (2).

  Gene expression (transcriptome) data were obtained from three different platforms: Affymetrix Human Genome U133 Plus 2.0 Array from the Broad Institute, Agilent 244K Array from University of North Carolina Lineberger Cancer Center and the Affymetrix GeneChip Human Exon 1.0 ST Array from Lawrence Berkeley National Laboratory. Raw data from each platform were combined with a linear statistical model to obtain unified expression measures for each gene.

  Of the original 206 samples, we adopted unified consensus gene expression values for 202. Relevant data for the remaining 4 samples were not available at the time we started our study. Thus, our gene expression data matrix contained 202 samples and 11861 genes before filtering for the most informative genes.

  Filtering: Adopting the strategy by Verhaak *et al.*(3), we carried out filtering in three steps and selected 1740 genes for subsequent analyses. (Numbers in parentheses

indicate how many remained after that filtering step):

1. Find genes with average β-value higher than 0.7 in at least two platforms ($\sim$ 9000 genes)
   - β_i measures correlation between observed values on platform i and the true expression value, hence this step is selecting for genes whose platform values are highly correlated with the true value in at least two platforms.
2. Find genes with maximum absolute deviation higher than 0.5 ($\sim$ 1900 genes)
   - This step is removing genes that do not show high enough variability among platforms
3. Remove genes with large difference in scale between platforms (1740 genes)
   - This step is removing genes whose expression values are unrealistically (orders of magnitude) different across platforms.

We adopted the same list of 1740 genes for our analysis. Throughout this paper, this 1740×202 dataset is referred to as **GBM1**.

- **GBM2** and **Validation**

  The gene expression dataset from the second TCGA cohort (**GBM2**) contains 175 tumor samples, and is also filtered to the same set of 1740 genes. The validation dataset used in (3) is a collection of samples from four previous studies (4–7). This dataset contains 260 samples, and the number of genes in common with the TCGA filtered gene set is 1676.

3.7 Supplementary Materials

3.7.1 Comparing clustering signals between GBM1 and the *pcNormal* null data sets

3.7.1.1 Cluster stability via empirical CDF plots

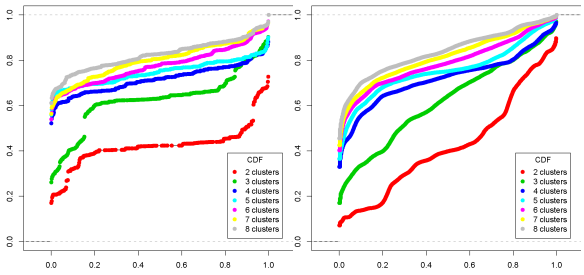
Figures 3.20a and 3.20b show the CDF plots for gene subsampling and sample subsampling consensus runs of **GBM1** respectively.Figure 3.20c and 3.20d show the same plots for **Sim25**.

The x-axis denotes the consensus matrix value c, and the y-axis denotes the corresponding CDF value $CDF(c)$. For sample pairs that are never grouped together, the consensus index value is 0. For pairs that are always found in the same cluster, this value is 1. A higher stability in cluster membership is suggested when there are more values close to both 0 and 1 for a fixed number of K clusters, and fewer intermediary values between 0 and 1; *i.e* sample pairs remain in the same cluster in the presence of perturbations. The y-axis on the CDF plots indicates the proportion of consensus index values smaller than or equal to the corresponding x-value.

Highly stable clusters generate a CDF curve that resembles a **three-phase step function**. The first phase is the vertical step at $c = 0$, which corresponds to sample pairs that are never found in the same cluster. The second phase is the horizontal line reaching across the $[0-1]$ range. Fewer values in this range are indicative of more stable clusters. The third phase of the step function is another vertical increase, this time at $c = 1$. This corresponds to sample pairs that get assigned to the same cluster even in the presence of perturbations.

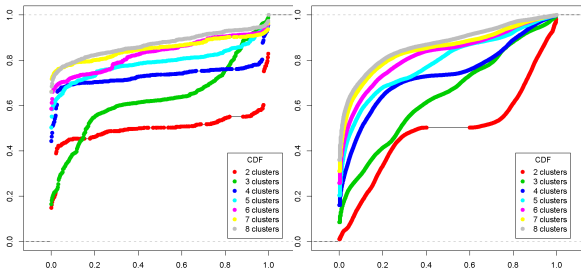
Less stable clusters exhibit a deviation from the three-phase step function and generate a gradual climb from 0 to 1. The area under the curve for a given K value is a measure of stability for cluster membership with larger areas indicating greater stability. The change in this area from K to $K+1$ clusters is used to evaluate the optimality of K as the number of classes in the data set. If there is not a significant increase in the area at $K+1$, the optimal number of clusters is estimated to be K.

'Sample subsampling' invokes a stronger perturbation to the **GBM1** and **Sim25** data



(a) GBM1 gene subsampling

(b) GBM1 sample subsampling



(c) Sim25 gene subsampling

(d) Sim25 sample subsampling

Figure 3.20: Empirical cumulative distribution function (CDF) plots for **GBM1** and **Sim25**, $K = \{2, \ldots, 8\}$. The x-axis is consensus index value c, and the y-axis is the corresponding $CDF(c)$.

sets compared to 'gene subsampling' at the same percentage. This is due to the difference in the number of samples ($N_s = 202$) and genes ($N_g = 1740$). A random 80% sample selection results in 162 samples with a loss of 40 samples, whereas a random 80% gene selection filters out 348 genes leaving 1392 genes in the dataset. It is highly likely that 40 samples that are randomly filtered out would carry a stronger clustering signal than the random set of 348 genes. It is often the case that genes function in groups, and the ones in the same group possess a similar expression profile. Thus, removing such a random set of genes from the dataset does not result in loss of a significant clustering signal; other genes in the same group can act as proxies. Therefore, the boundaries of clusters may not change drastically in the case of gene subsampling. However, the same cannot be said of sample subsampling. Removing samples from the dataset is much more likely to be a direct interruption to cluster boundaries, hence reducing cluster stability. Moreover, cancer specimens are collected independently, and one does not generally expect to observe that

samples in the dataset will function as a proxy to others removed from the dataset.

We are able to observe the consequences of the difference between gene subsampling and sample subsampling also in CDF plots (Figure 3.20a vs. Figure 3.20b, and Figure 3.20c vs. Figure 3.20d). The curves in these figures correspond to K values in $\{2, \ldots, 8\}$ for their corresponding datasets. Comparing the CDF curve of a particular K value between gene-subsampling and sample-subsampling, we observe that curves in the former case resemble the aforementioned **three-phase step function** much more closely than those from the latter, suggesting stronger apparent cluster stability with gene-subsampling.

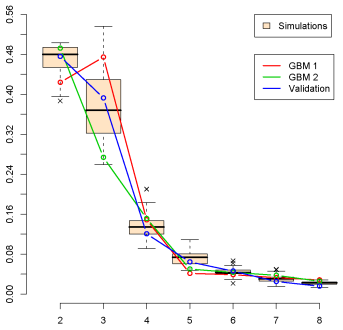
Figure 3.20a shows that CDF curves for **GBM1** gene-subsampling resemble a step function even when K is as small as 2. The CDF shape improves and approaches an ideal step function as K is raised to 3 and 4, however, there is no significant change after $K = 4$. This could be suggestive of four real clusters in the **GBM1** dataset, however, background signal for clustering stability needs to be accounted for before declaring real clusters. The equivalent plot for **Sim25** is shown in Figure 3.20c. Recall that **Sim25** carries the same gene-gene correlation structure as in **GBM1**, but possesses randomized sample-sample distances. We observe that $K = 2$ has stronger cluster stability in this dataset compared to **GBM1**. The CDF curve for $K = 3$ is similar to that of **GBM1**, and again the curves for $K = \{4, \ldots, 8\}$ are similar in the two plots. However, it is noteworthy that the curve for $K = 4$ in **Sim25**, when compared with the corresponding curve in **GBM1**, is closer to a step function. Overall, the cluster stability observed for **Sim25** is at least as good as, if not better than, that observed for **GBM1**. This suggests that the apparent clustering stability on consensus clustering CDF plots may have potential to mislead one to declare real clusters when sufficient evidence does not exist.

The same comparison can be made between Figure 3.20b and Figure 3.20d. These show CDF plots for sample-subsampling consensus runs of **GBM1** and **Sim25** respectively. Again, K values range from 2 to 8, with the CDF curve for $K = 2$ typically found at the bottom, and other curves following the previous K value. The gradual climb we observe for $K = 2$ in Figure 3.20b is indicative of poor clustering stability. As mentioned above, the removal of 40 samples from the **GBM1** dataset results in a strong enough perturbation

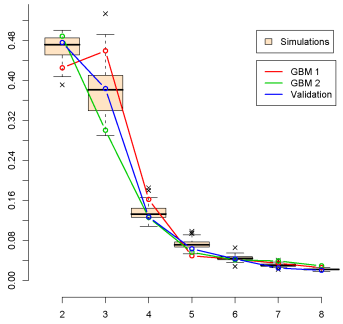
to eliminate most of the apparent cluster stability. In contrast, **Sim25** exhibits a better cluster stability for $K = 2$ (Figure 3.20d) than **GBM1** even though sample perturbation still substantially degraded clustering quality as compared with the $K = 2$ curve for gene-subsampling (Figure 3.20c). For $K = \{3, \ldots, 8\}$, CDF curves are again more similar across **GBM1** and **Sim25**. It can be noted that, in contrast with the gene-subsampling case, **GBM1** has more step-function-like CDF curves for $K = \{4, \ldots, 8\}$ while those of **Sim25** appear to make gradual climbs particularly in the $[0, 0.4]$ range (Figure 3.20b and Figure 3.20d). This is likely to be due to the real clustering signal in the **GBM1** dataset and to the fact that random samples from the same distribution may be more similar to one another than the samples in a real dataset such as **GBM1** (hence, lower CDF values for $c = 0$ with **Sim25**).

3.7.1.2 Optimal K via proportional area change under CDF ($\Delta(K)$)

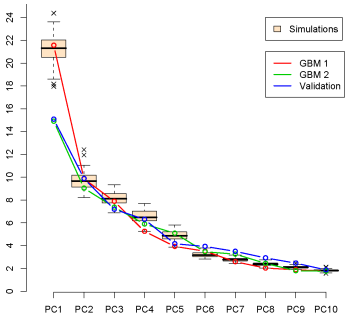
As with silhouette widths and gap-statistics, comparing the behavior of $\Delta(K)$ for real and simulated datasets will provide insight regarding the strength of cluster signals in a background distribution that is known to lack sample structure. Figures 3.21a and 3.21b show $\Delta(K)$ values for real datasets (**GBM1**, **GBM2**, **Validation**) and 50 pcNormal simulations with gene-subsampling and sample-subsampling respectively. The values for simulated datasets are shown in box-whisker plots while those of real datasets are overlaid with three lines. It is remarkable that the three real datasets can all be placed within the box-whisker ranges of simulated datasets with known lack of sample structure, for both gene-subsampling and sample-subsampling. This observation provides perspective for inspecting CDF progression as $\Delta(K)$ values for simulations present a null-distribution to compare real datasets against. Otherwise, the objective of selecting the largest K that induces a 'large enough' increase in the area under the corresponding CDF is not a well-defined one. What constitutes a 'large enough' increase is subject to personal interpretation.



(a) $\Delta(K)$ *vs.* K, gene subsampling



(b) $\Delta(K)$ *vs.* K, sample subsampling



(c) % variance explained by first 10 principal components

Figure 3.21: Proportion increase $\Delta(K)$ in the area under CDF curves for consensus runs of (a) gene subsampling, and (b) sample subsampling. The values for the set of 50 pcNormal simulations are shown in box-whisker plots, while the values of real datasets are overlaid with three lines (**GBM1**: red, **GBM2**: green, and **Validation**: blue). (c) Boxplots for percentage of variance explained by the first 10 principal components in 50 simulations; overlaid with lines corresponding to **GBM1**, **GBM2**, and **Validation** data sets. Outliers are shown with black crosses.

3.7.1.3 Variance explained by principal components

Principal components of simulated datasets carry almost the same percentage of sample variance as those of **GBM1** (Figure 3.21c). We note that the first principal component of **GBM2** and the **Validation** do not carry as large a sample-variance percentage as those of the simulated datasets because simulated datasets are generated with the gene-gene correlation structure of **GBM1**. **GBM2** and **Validation** datasets do not have a gene-gene correlation structure as strong as **GBM1**. Furthermore, these observations also imply that **GBM2** and **Validation** datasets do not have a sample-sample correlation structure that is strong enough to compensate for the deficiency in the gene-gene correlation structure.

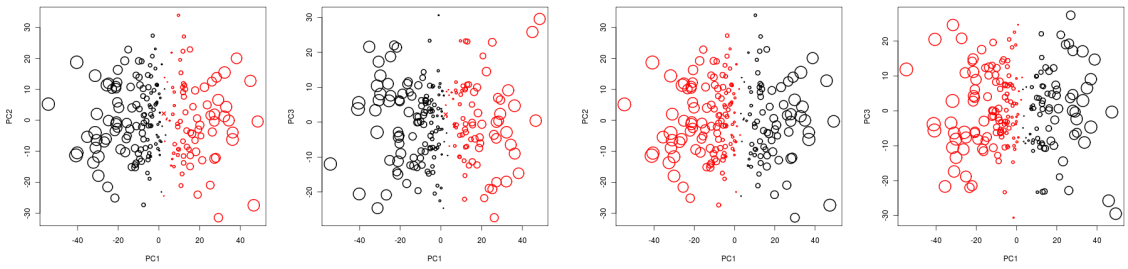
These datasets are likely to have a worse clustering structure compared to **GBM1**.

3.7.2 The progression of diagnostic measures and plots across increasing pull-apart degree a

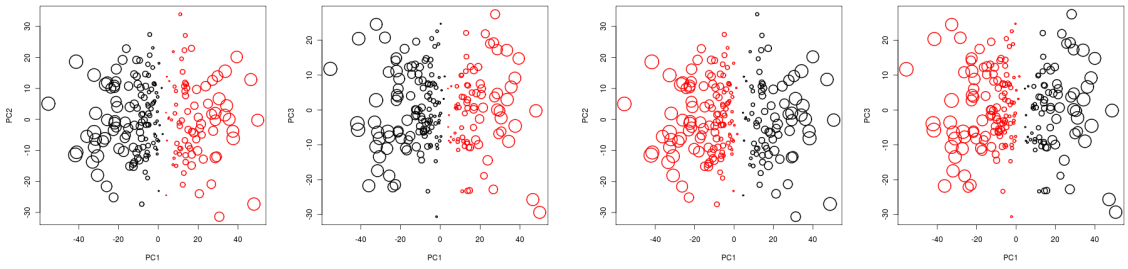
We present below the diagnostic plots used in our study to analyze the performance of different clustering methods and measures on pulled-apart data sets. Number of clusters for pulled-apart data sets vary in $\{2, ..., 6\}$ and pull-apart separation degree a vary in a range over $[0, 0.6]$.

3.7.2.1 Pull-apart bubble plots

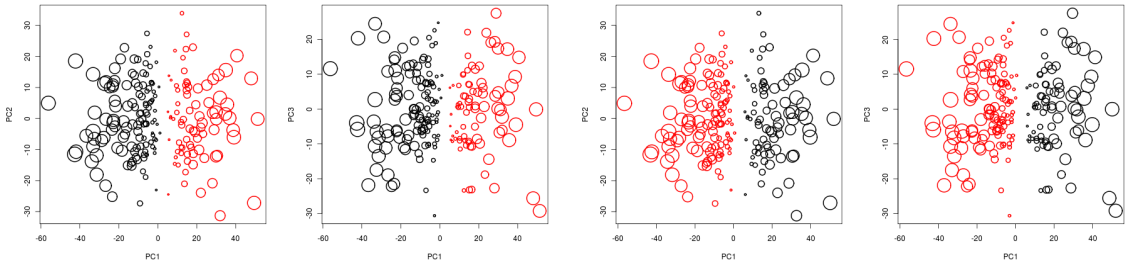
Each figure in a panel shows a two-dimensional scatter plot of the samples in (left) the PC1-PC2 space, and (right) the PC1-PC3 space. K-means was used to obtain a partitioning of the samples. The sizes of the bubbles indicate the magnitude of the silhouette width. Negative silhouette widths are shown with fixed-size crosses.



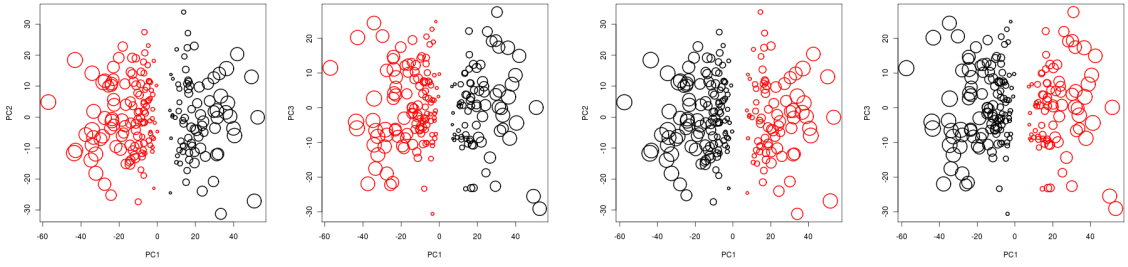
(a) 2-way pull-apart, $a = 0$ **(b)** 2-way pull-apart, $a = 0.04$



(c) 2-way pull-apart, $a = 0.08$ **(d)** 2-way pull-apart, $a = 0.12$

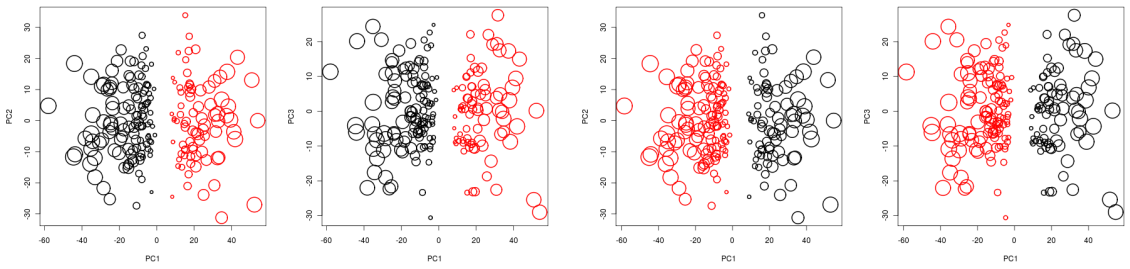


(e) 2-way pull-apart, $a = 0.16$ **(f)** 2-way pull-apart, $a = 0.2$

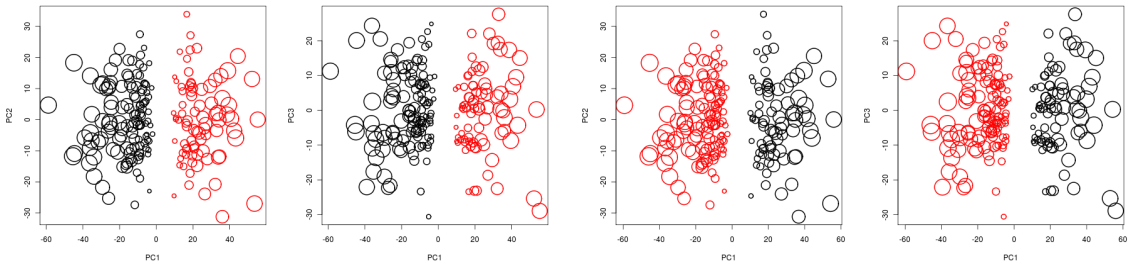


(g) 2-way pull-apart, $a = 0.24$ **(h)** 2-way pull-apart, $a = 0.28$

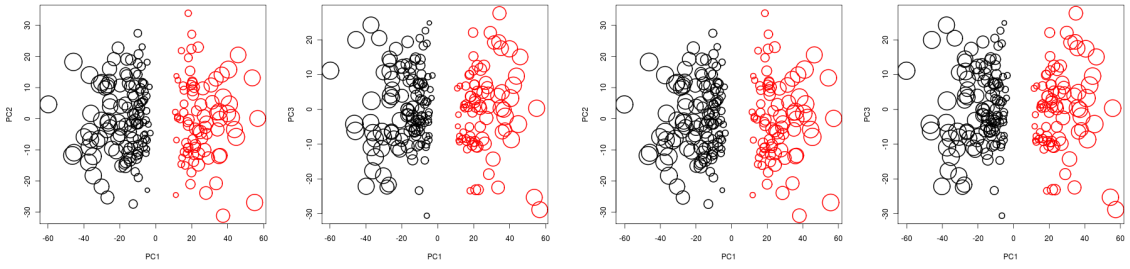
Figure 3.22: PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 2 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 2 clusters.



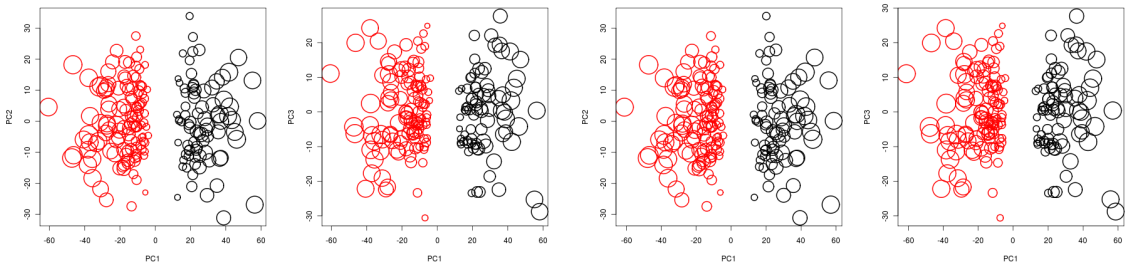
(i) 2-way pull-apart, $a = 0.32$ **(j)** 2-way pull-apart, $a = 0.36$



(k) 2-way pull-apart, $a = 0.4$ **(l)** 2-way pull-apart, $a = 0.44$

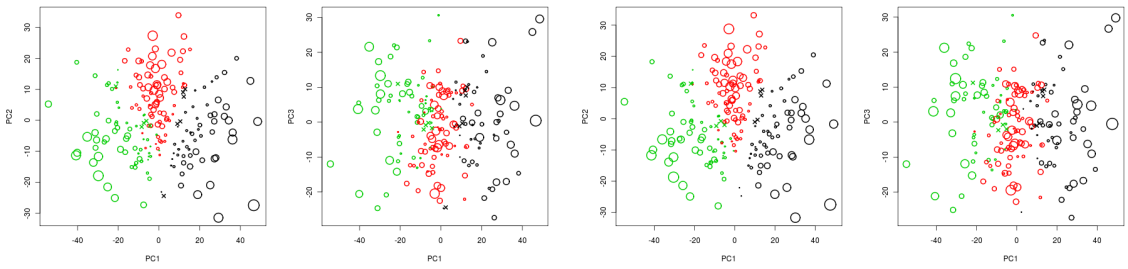


(m) 2-way pull-apart, $a = 0.48$ **(n)** 2-way pull-apart, $a = 0.52$

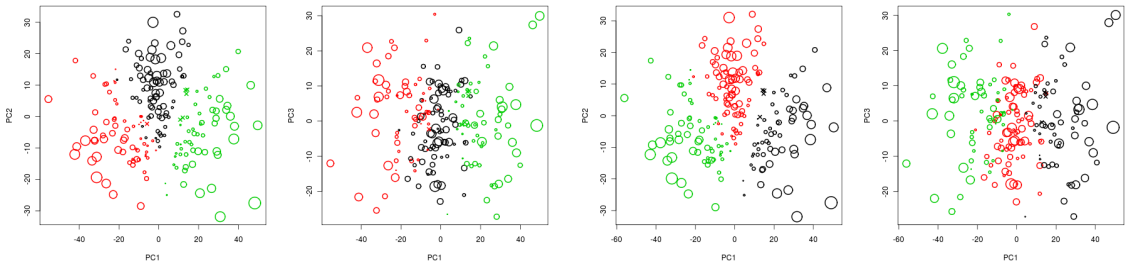


(o) 2-way pull-apart, $a = 0.56$ **(p)** 2-way pull-apart, $a = 0.6$

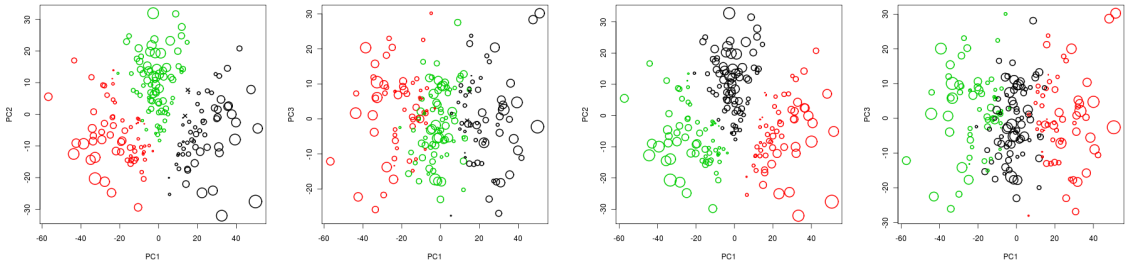
Figure 3.22: (Continued) PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 2 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 2 clusters.



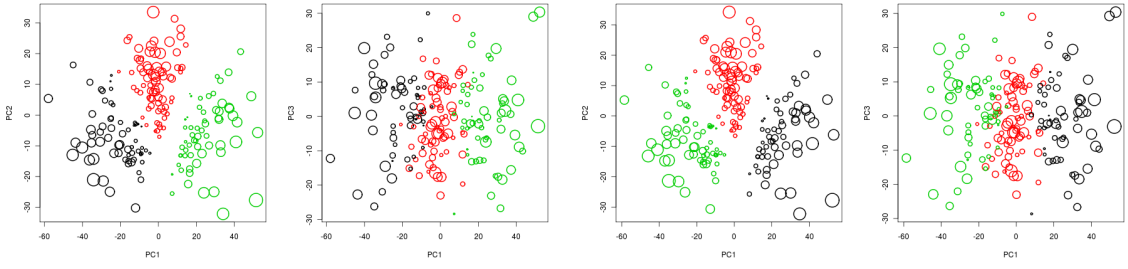
(a) 3-way pull-apart, $a = 0$ **(b)** 3-way pull-apart, $a = 0.04$



(c) 3-way pull-apart, $a = 0.08$ **(d)** 3-way pull-apart, $a = 0.12$

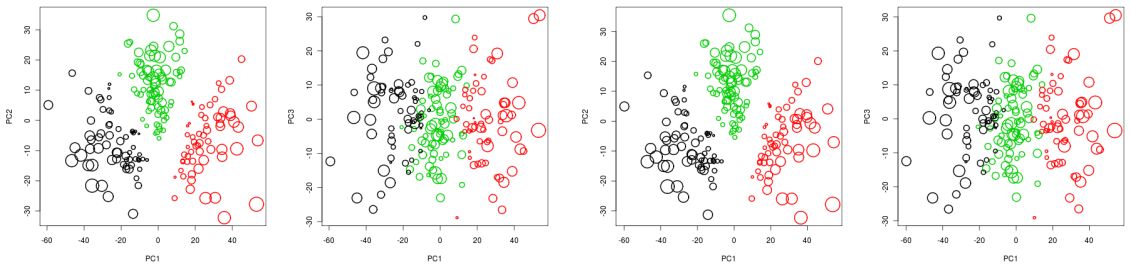


(e) 3-way pull-apart, $a = 0.16$ **(f)** 3-way pull-apart, $a = 0.2$



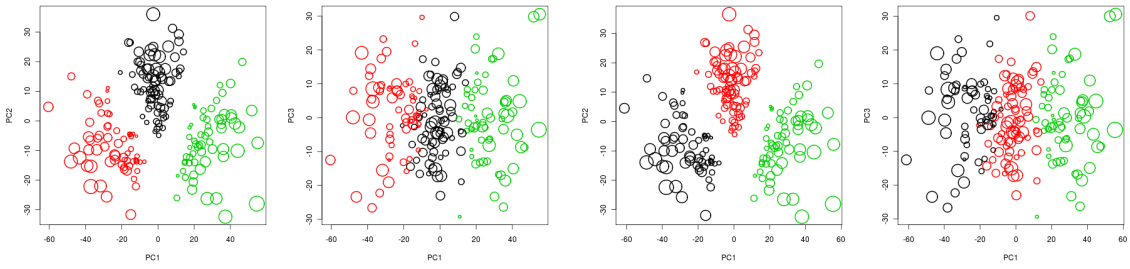
(g) 3-way pull-apart, $a = 0.24$ **(h)** 3-way pull-apart, $a = 0.28$

Figure 3.23: PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 3 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 3 clusters.



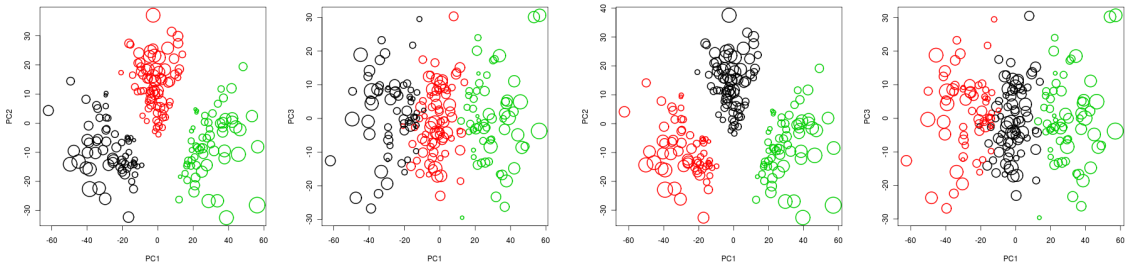
(i) 3-way pull-apart, $a = 0.32$

(j) 3-way pull-apart, $a = 0.36$



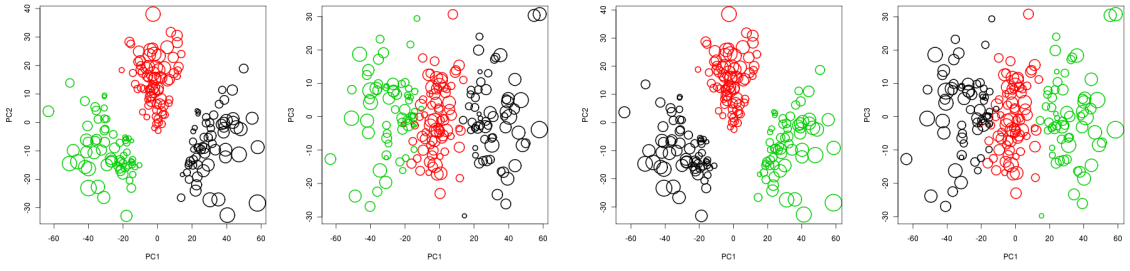
(k) 3-way pull-apart, $a = 0.4$

(l) 3-way pull-apart, $a = 0.44$



(m) 3-way pull-apart, $a = 0.48$

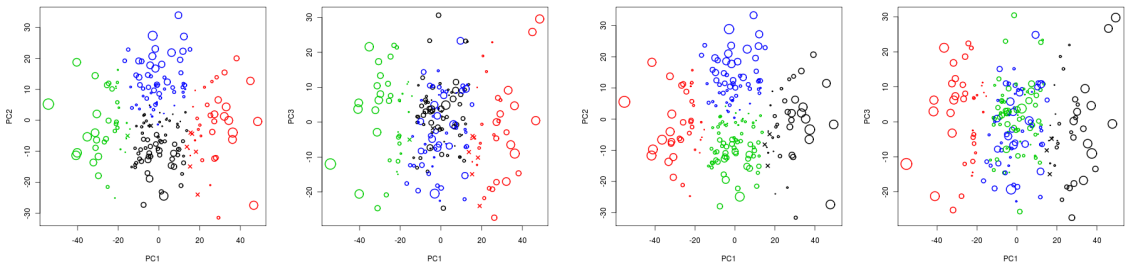
(n) 3-way pull-apart, $a = 0.52$



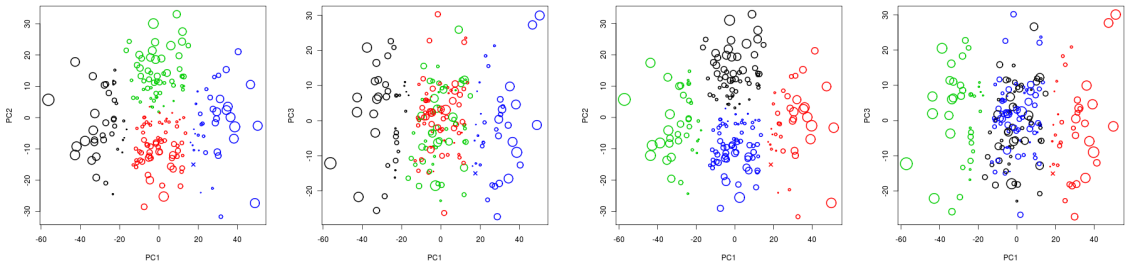
(o) 3-way pull-apart, $a = 0.56$

(p) 3-way pull-apart, $a = 0.6$

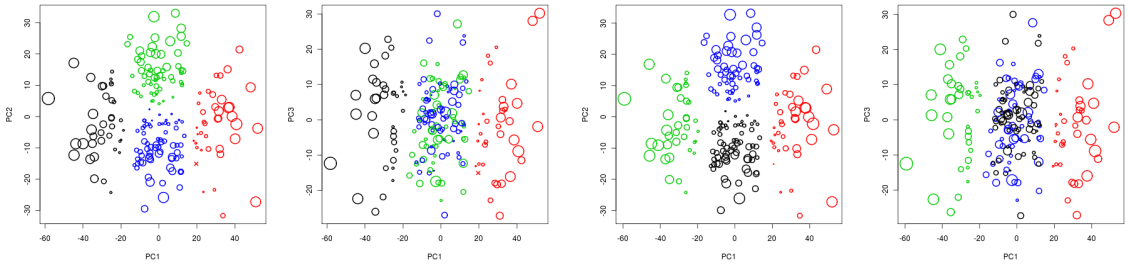
Figure 3.23: (Continued) PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 3 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 3 clusters.



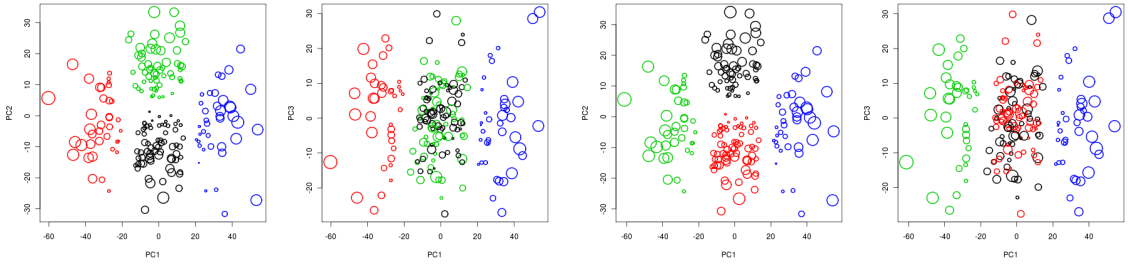
(a) 4-way pull-apart, $a = 0$ **(b)** 4-way pull-apart, $a = 0.04$



(c) 4-way pull-apart, $a = 0.08$ **(d)** 4-way pull-apart, $a = 0.12$

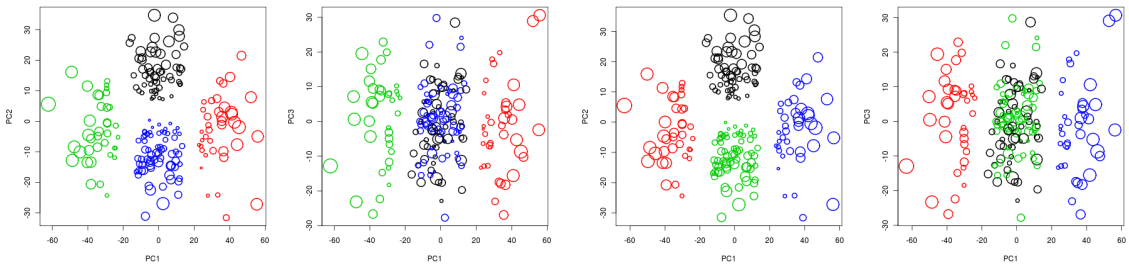


(e) 4-way pull-apart, $a = 0.16$ **(f)** 4-way pull-apart, $a = 0.2$

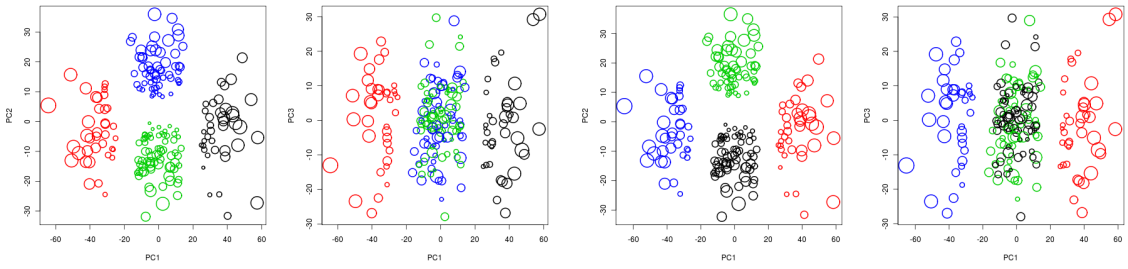


(g) 4-way pull-apart, $a = 0.24$ **(h)** 4-way pull-apart, $a = 0.28$

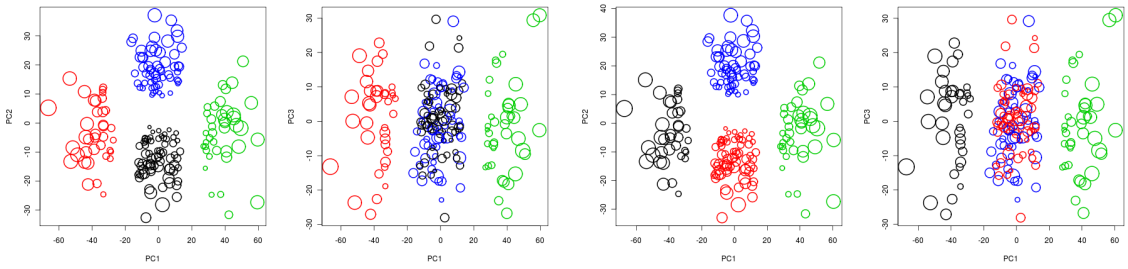
Figure 3.24: PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 4 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 4 clusters.



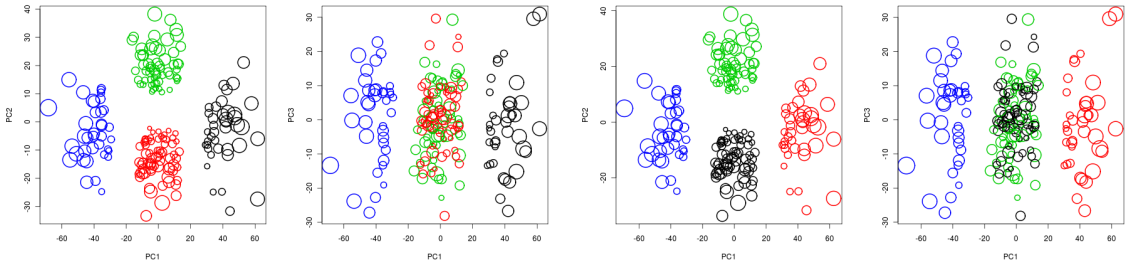
(i) 4-way pull-apart, $a = 0.32$ **(j)** 4-way pull-apart, $a = 0.36$



(k) 4-way pull-apart, $a = 0.4$ **(l)** 4-way pull-apart, $a = 0.44$

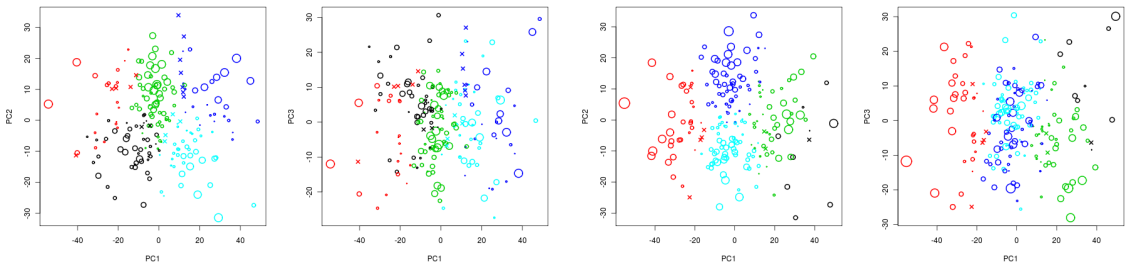


(m) 4-way pull-apart, $a = 0.48$ **(n)** 4-way pull-apart, $a = 0.52$

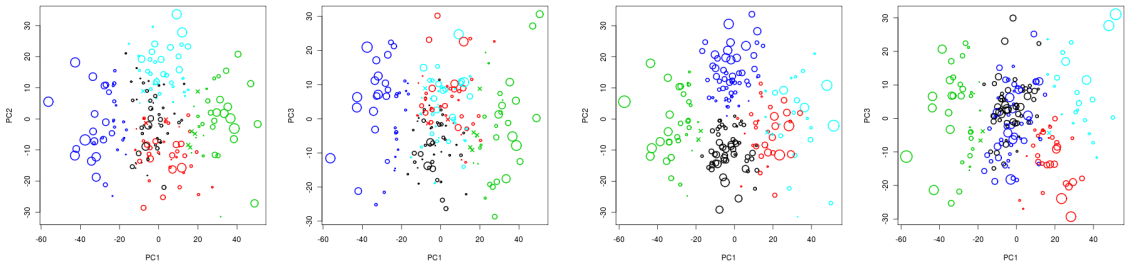


(o) 4-way pull-apart, $a = 0.56$ **(p)** 4-way pull-apart, $a = 0.6$

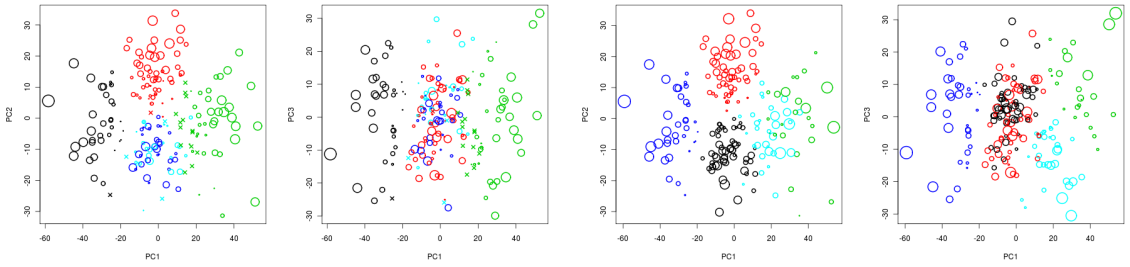
Figure 3.24: (Continued) PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 4 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 4 clusters.



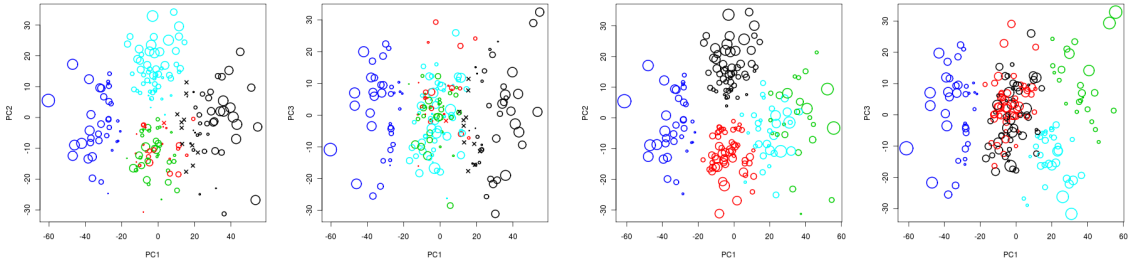
(a) 5-way pull-apart, $a = 0$ **(b)** 5-way pull-apart, $a = 0.04$



(c) 5-way pull-apart, $a = 0.08$ **(d)** 5-way pull-apart, $a = 0.12$

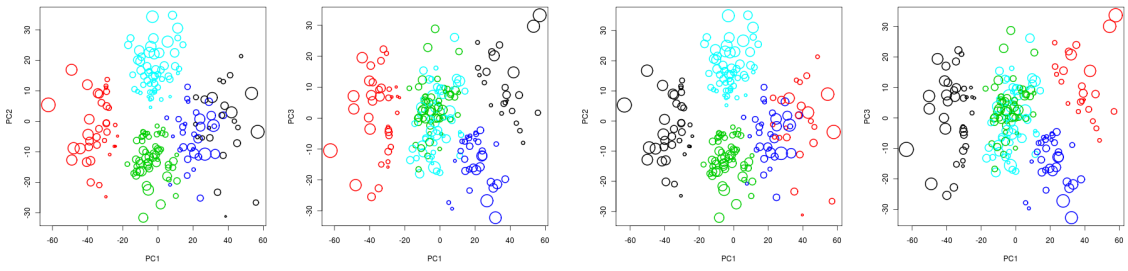


(e) 5-way pull-apart, $a = 0.16$ **(f)** 5-way pull-apart, $a = 0.2$

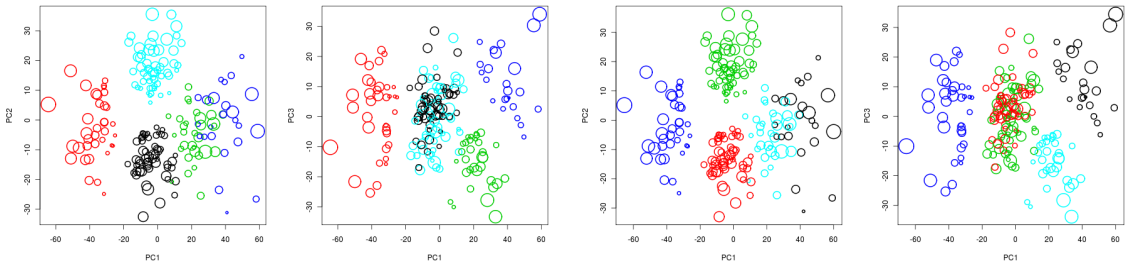


(g) 5-way pull-apart, $a = 0.24$ **(h)** 5-way pull-apart, $a = 0.28$

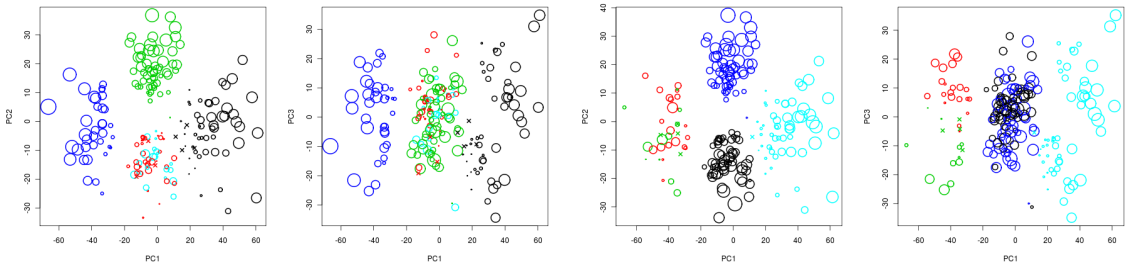
Figure 3.25: PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 5 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 5 clusters.



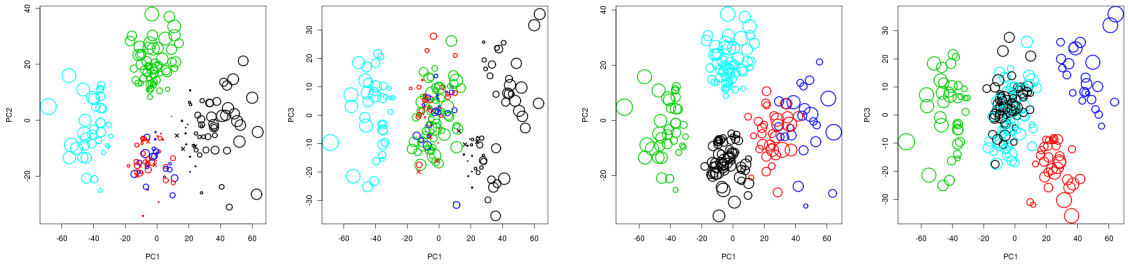
(i) 5-way pull-apart, $a = 0.32$ **(j)** 5-way pull-apart, $a = 0.36$



(k) 5-way pull-apart, $a = 0.4$ **(l)** 5-way pull-apart, $a = 0.44$

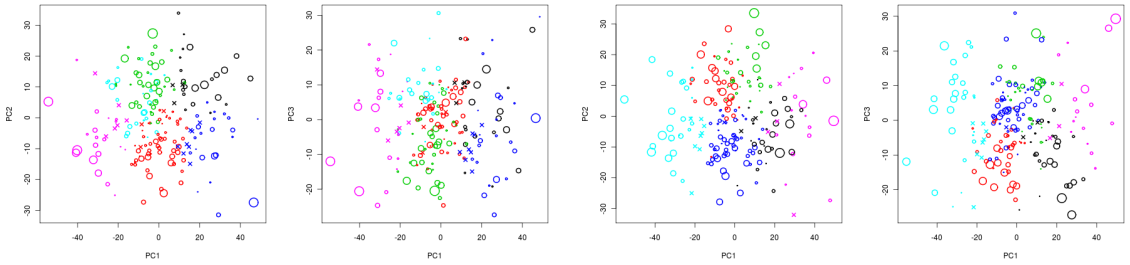


(m) 5-way pull-apart, $a = 0.48$ **(n)** 5-way pull-apart, $a = 0.52$

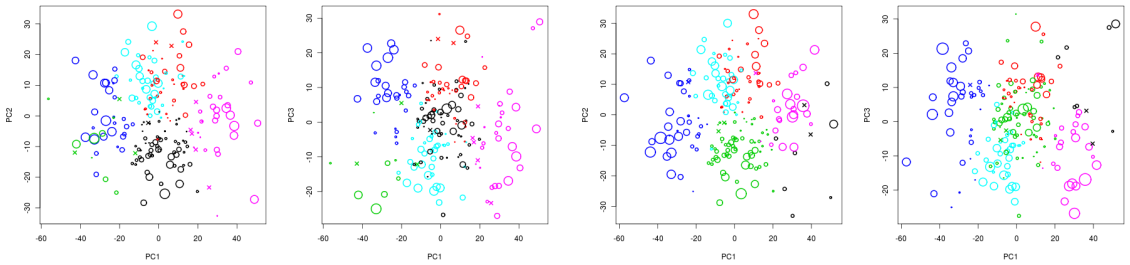


(o) 5-way pull-apart, $a = 0.56$ **(p)** 5-way pull-apart, $a = 0.6$

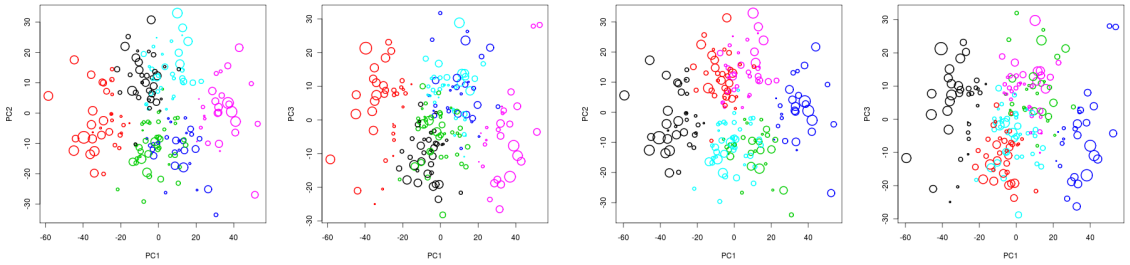
Figure 3.25: (Continued) PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 5 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 5 clusters.



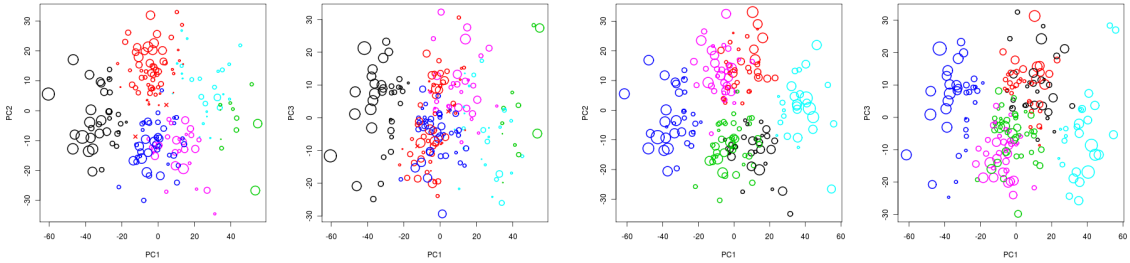
(a) 6-way pull-apart, $a = 0$ **(b)** 6-way pull-apart, $a = 0.04$



(c) 6-way pull-apart, $a = 0.08$ **(d)** 6-way pull-apart, $a = 0.12$

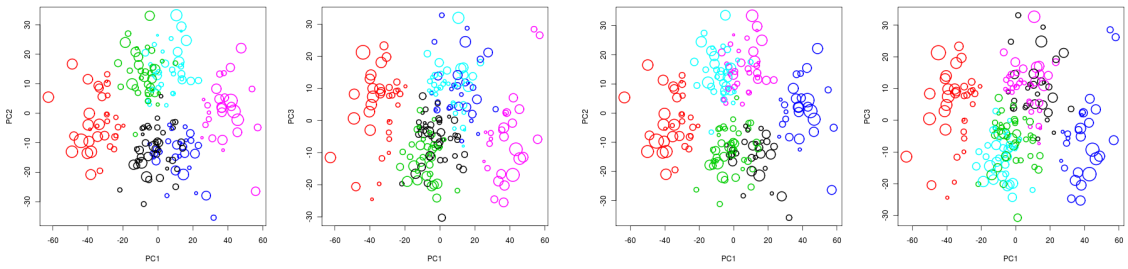


(e) 6-way pull-apart, $a = 0.16$ **(f)** 6-way pull-apart, $a = 0.2$

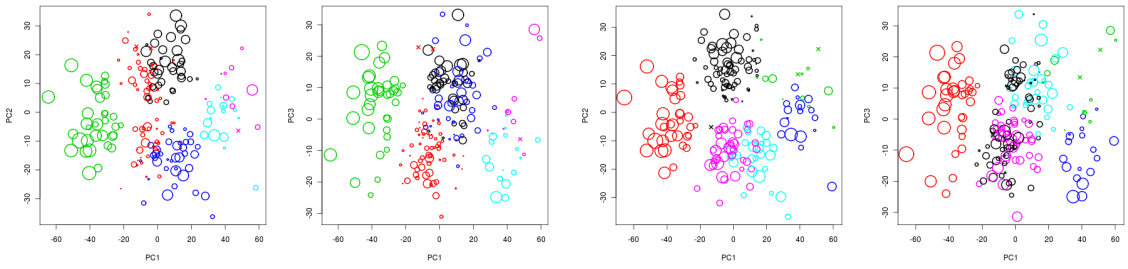


(g) 6-way pull-apart, $a = 0.24$ **(h)** 6-way pull-apart, $a = 0.28$

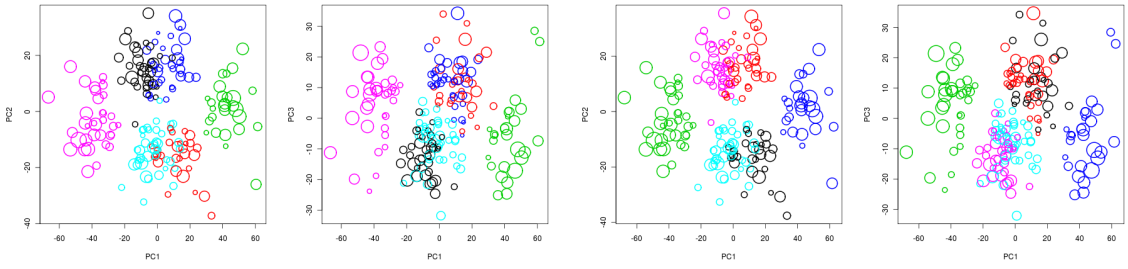
Figure 3.26: PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 6 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 6 clusters.



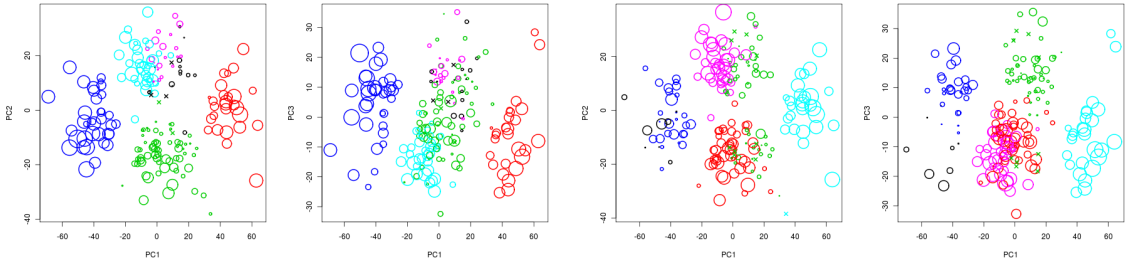
(i) 6-way pull-apart, $a = 0.32$ **(j)** 6-way pull-apart, $a = 0.36$



(k) 6-way pull-apart, $a = 0.4$ **(l)** 6-way pull-apart, $a = 0.44$



(m) 6-way pull-apart, $a = 0.48$ **(n)** 6-way pull-apart, $a = 0.52$

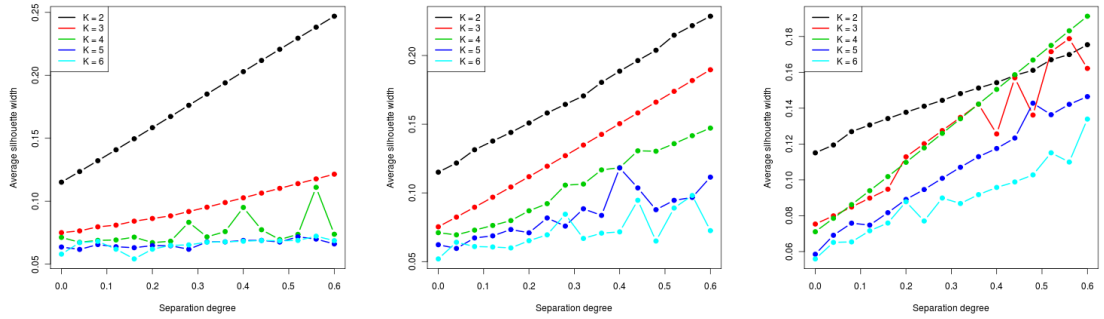


(o) 6-way pull-apart, $a = 0.56$ **(p)** 6-way pull-apart, $a = 0.6$

Figure 3.26: (Continued) PC1 *vs.* PC2, and PC1 *vs.* PC3 plots for a *pcNormal* dataset pulled apart into 6 clusters with separation degree a in [0,0.6]. The coloring scheme is from a K-means classification with 6 clusters.

3.7.2.2 Average silhouette width progression across a.

Average of the silhouette width values of all samples was computed for each separation degree and each number of clusters. The lines track the progression of the average values (y-axis) across the pull-apart separation degree (x-axis). We observe that $K = 2$ usually yields higher results than other K values regardless of the pull-apart number of clusters and separation degree, hence making the true number of clusters unidentifiable most of the time.



(a) 2-way pull-apart **(b)** 3-way pull-apart **(c)** 4-way pull-apart

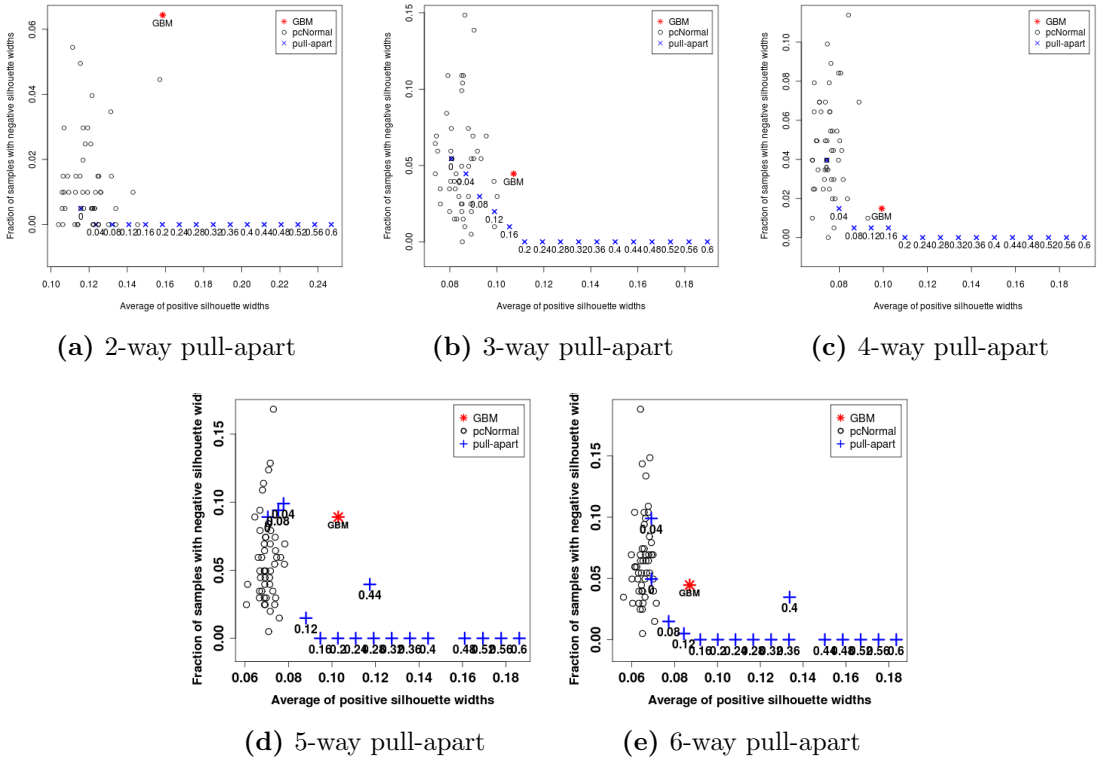
(d) 5-way pull-apart **(e)** 6-way pull-apart

Figure 3.27: Progression of average silhouette width values of pulled-apart datasets across separation degree (a).

3.7.2.3 Pull-apart silhouette scatter plots

Silhouette widths for a data set can also be plotted on a two-dimensional space to analyze the positive and negative values separately. These plots show the average of the positive silhouette widths on the x-axis, and the fraction of negative silhouette widths on

the y-axis. **GBM1**, shown with a red asterisk, is compared with *pcNormal* simulations (hollow circles) and pulled-apart data sets (blue pluses).



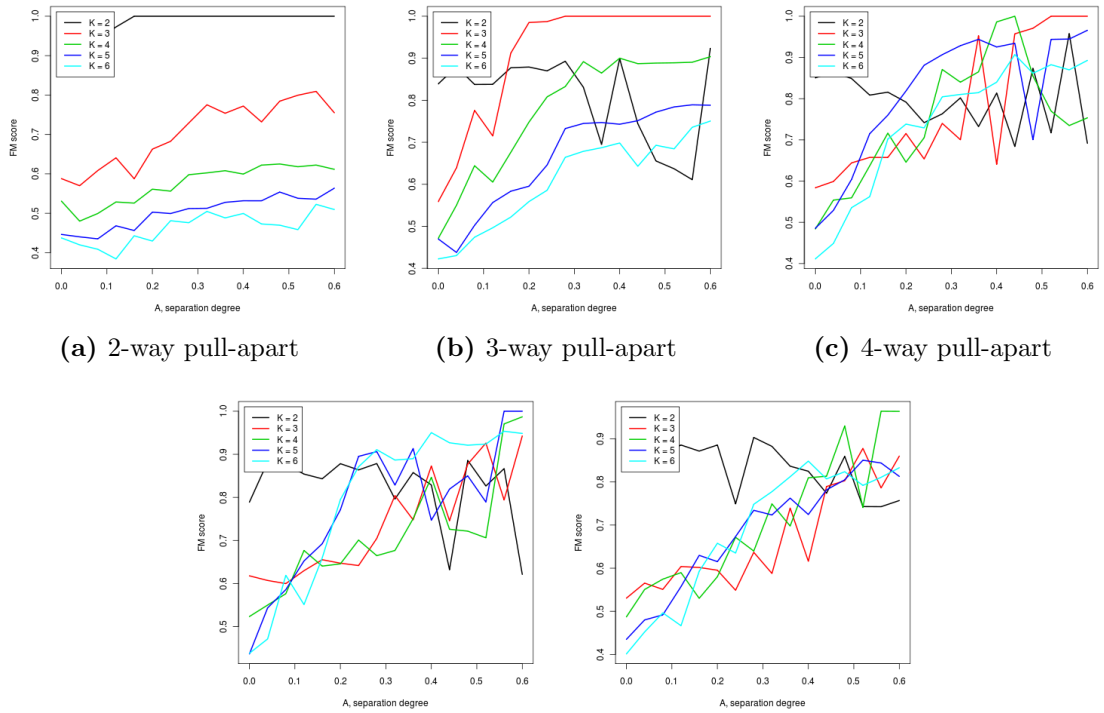
(a) 2-way pull-apart **(b)** 3-way pull-apart **(c)** 4-way pull-apart

(d) 5-way pull-apart **(e)** 6-way pull-apart

Figure 3.28: Silhouette scatter plots for pulled-apart datasets.

3.7.2.4 CLEST progression across a.

The FM-index results from CLEST are plotted on the y-axis against pull-apart separation degree on the x-axis. We observe that it is possible to identify the true number of clusters for 2-way and 3-way pull-apart at larger separation degrees; however CLEST cannot identify the true number for 4,5,and 6-way pull-apart at this given range of separation degrees.



(a) 2-way pull-apart **(b)** 3-way pull-apart **(c)** 4-way pull-apart

(d) 5-way pull-apart **(e)** 6-way pull-apart

Figure 3.29: Progression of CLEST's FM indices for pulled-apart datasets across separation degree (a).

Bibliography

[1] Furnari F.B. et al. (2007) Malignant astrocytic glioma: genetics, biology, and paths to treatment. Genes Dev. 21:2683-2710.

[2] The Cancer Genome Atlas Research Network (2008) Comprehensive genomic characterization defines human glioblastoma genes and core pathways. Nature.455:1061-1068.

[3] Verhaak R.G.W. et al. (2010) Integrated Genomic Analysis Identifies Clinically Relevant Subtypes of Glioblastoma Characterized by Abnormalities in *PDGFRA, IDH1, EGFR,* and *NF1*. Cancer Cell. Vol. 17, Issue 1, pp. 98-110.

[4] Beroukhim R. et al. (2007) Assessing the significance of chromosomal aberration in cancer: methodology and application to glioma Proc. Natl. Acad. Sci. USA 104, 20007-20012.

[5] Phillips H.S. et al. (2006) Molecular subclasses of high-grade glioma predict prognosis, delineate a pattern of diseases progression, and resemble stages in neurogenesis. Cancer Cell. 9, 157-153.

[6] Murat A. et al. (2008) Stem cell-related "self-renewal" signature and high epidermal growth factor receptor expression associated with resistance to concomitant chemoradiotherapy in glioblastoma. J. Clin. Oncol. 26, 3015-3024.

[7] Sun L. et al. (2006) Neuronal and glioma-derived stem cell factor induces angiogenesis within the brain. Cancer Cell. 9, 287-300.

[8] Rousseeuw, P.J. (1987) Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math., 20.

[9] Tibshirani, R., Walther, G., Hastie, T. (2001) Estimating the number of clusters in a data set via the gap statistic. Journal of the Royal Statistical Society, ser. B, vol. 63, part 2, 411-423.

[10] Dudoit, S., Fridlyand, J. (2002) A prediction-based resampling method for estimating the number of clusters in a dataset. Genome Biology 3, research0036.1-research0036.21.

[11] Monti S, et al. (2003) Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. Machine Learning. 52:91-118.

[12] Johnson, S.C. (1967) Hierarchical clustering schemes. Psychometrica 32: 241-254.

[13] McQueen, J.B. (1967) Some methods for classification and analysis of multivariate observations. Proc. Fifth Berkeley Symp. Math. Stat. Probab. 1:281-297.

[14] Kaufman L., Rousseeuw P. (1990) Finding groups in data: An introduction to cluster analysis. New York:Wiley.

[15] Fraley C., Raftery A.E. (2002a) MCLUST: Software for model-based clustering, density estimation and discriminant analysis. Technical Report, Dept of Statistics, University of Washington, WA.

[16] Fraley C., Raftery A.E. (2002b) Model-based clustering, discriminant analysis, and density estimation. J. Am. Stat. Assoc; 97:611-631.

[17] Hayes DN, Monti S, Parmigiani G, Gilks CB, Naoki K, Bhattacharjee A, et al. (2006) Gene expression profiling reveals reproducible human lung adenocarcinoma subtypes in multiple independent patient cohorts. J Clin Oncol; 24:507990.

[18] Monti S, Savage KJ, Kutok JL. (2005) Molecular profiling of diffuse large B-cell lymphoma identifies robust subtypes including one characterized by host inflammatory response. Blood; 105(5):1851-1861.

[19] Wilkerson M.D. et al. (2010) Lung Squamous Cell Carcinoma mRNA Expression Subtypes Are Reproducible, Cinically Important, and Correspond to Normal Cell Types. Clin Cancer Res; 16(19):4864-4875.

[20] Gibbons F. and Roth F. (2002) Judging the quality of gene expression-based clustering methods using gene annotation. Genome Res.; 12, 1574-1581.

[21] Mar J.C., Wells C.A., and Quackenbush J. (2011) Defining an informativeness metric for clustering gene expression data. Bioinformatics; 27(8), 10941100.

[22] Lapointe J et al. (2004) Gene expression profiling identifies clinically relevant subtypes of prostate cancer. Proc Natl Acad Sci USA. Jan 20;101(3):811-6.

[23] Alizadeh, A. A. et al. (2000) Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403, 503511.

[24] Bertucci F, Finetti P, Rougemont J, et al. (2005) Gene expression profiling identifies molecular subtypes of inflammatory breast cancer. Cancer Res; 65:2170-2178.

[25] Handl, J., Knowles, J., and Kell, D.B. (2005) Computational cluster validation in post-genomic data analysis. Bioinformatics, 21: 3201-3212.

CHAPTER IV

Joint Estimation of DNA Copy Number from Multiple Platforms

4.1 Abstract

Motivation: DNA copy number variants (CNV) are gains and losses of segments of chromosomes, and comprise an important class of genetic variation. Recently, various microarray hybridization-based techniques have been developed for high throughput measurement of DNA copy number. In many studies, multiple technical platforms or different versions of the same platform were used to interrogate the same samples; and it became necessary to pool information across these multiple sources to derive a consensus molecular profile for each sample. An integrated analysis is expected to maximize resolution and accuracy, yet currently there is no well formulated statistical method to address the between-platform differences in probe coverage, assay methods, sensitivity, and analytical complexity.

Results: The conventional approach is to apply one of the CNV detection (a.k.a. "segmentation") algorithms to search for DNA segments of altered signal intensity. The results from multiple platforms are combined after segmentation. Here we propose a new method, Multi-Platform Circular Binary Segmentation (MPCBS), which pools statistical evidence across platforms *during* segmentation, and does not require pre-standardization of different data sources. It involves a weighted sum of t-statistics, which arises naturally from the generalized log-likelihood ratio of a multi-platform model. We show by comparing the

integrated analysis of Affymetrix and Illumina SNP array data with Agilent and fosmid clone end-sequencing results on 8 HapMap samples that MPCBS achieves improved spatial resolution, detection power, and provides a natural consensus across platforms. We also apply the new method to analyze multi-platform data for tumor samples.

4.2 Introduction

In recent years, more and more genetic studies have relied on collecting genome-scale data on DNA variants. With the rapid influx of large data sets came the increasingly common problem of data integration when multiple technical platforms (or different versions of the same platform) were used to interrogate the same biological samples. For example, the Cancer Genome Atlas (TCGA) project, an NIH-funded initiative to characterize DNA, RNA, and epigenetic abnormalities in tumors, has adopted three independent platforms for studying DNA copy number variants (CNVs) in its pilot phase: Affymetrix SNP 6.0 arrays, Illumina HumanHap 550K SNP arrays, and Agilent CGH 244K arrays. The conventional approach for analyzing these data is to apply one of the CNV detection (a.k.a. "segmentation") algorithms to search for genomic intervals of altered signal intensity using data from each platform separately. The segmentation results from three platforms are then combined. However, when the platforms disagree on the calling of a CNV, it is difficult to decide what the consensus should be. Furthermore, the reported DNA copy numbers (i.e. the location and magnitude of the changes) are often different in different platforms. At the fundamental level, the three platforms represent three distinct marker panels and different molecular assay methods:

- Illumina arrays produce allele-specific data, Agilent arrays produce only the total intensity, whereas Affymetrix arrays have both allele-resolved SNP probes and invariant CNV probes, thus effectively containing two sub-platforms.
- Agilent arrays produce two-color ratio data in a test/reference format, while the other two measure each sample independently.

- In regions of high-fold amplification, Illumina and Affymetrix tend to have more pronounced signal saturation. In fact, all three platforms estimate the true levels of copy number change with different scaling factors, which may be non-linear and may vary across chromosomes or samples (Bengtsson et al., 2009).

- The three methods produce data values with distinct noise characteristics, with different proportions of low-quality SNPs and distinct local signal trends that are partly due to the sample amplification procedures used.

- For some, such as the Illumina data, the default normalization procedure is not tailored to copy number analysis.

In short, each platform has its advantages and disadvantages, but together they produce a more detailed genomewide survey for each sample. If the data sets from the three platforms are separately segmented, it is difficult to combine their respective segment summaries because, for the same underlying event, they will report different magnitudes, with different boundaries and different degrees of uncertainty. An integrated analysis, where information from all platforms are used at the same time to detect CNVs and to estimate the levels of change, is expected to maximize resolution and accuracy. Currently, however, there is no well formulated statistical method to address the between-platform differences in probe coverage, sensitivity, and analytical complexity. Simply combining the three data series into a single data set without proper normalization will not yield better segmentation results, because when the underlying true copy number is not known, it is difficult to determine how to normalize across platforms given the uneven coverage between the platforms at any genomic region.

In order to tackle the increasingly common problem of data integration across multiple sources, we propose a new method based on a simple multi-platform change-point model. The model extends existing approaches for detecting change-points in a single sequence (14) to the problem of detecting coupled changes in multiple sequences with differing noise and signal intensities. The model gives rise to an efficient algorithm, multi-platform Circular Binary Segmentation (MPCBS), which relies on a weighted sum of t statistics to scan for

copy number changes. MPCBS sums statistical evidence across platforms with proper scaling, and does not require a pre-standardization of different data sources. The statistics are derived through maximizing the likelihood of the multi-platform model, with the dimension of the model (i.e. the number of segments) chosen by maximizing a generalized form of the modified Bayes information criterion (BIC) criterion proposed in Zhang and Siegmund (15). Platform specific quantities such as noise variances and response ratios are also estimated by our method. Importantly, the method provides a single, platform-free consensus profile for each sample for downstream analyses.

4.3 Multiplatform Model and Methods Overview

Let the platforms be indexed by $k = 1, \ldots, K$, with K being the total number of platforms. We observe total intensity data $\mathbf{y}_k = y_{k1}, \ldots, y_{kn_k}$ for the n_k snps/clones on the k-th platform, which have ordered locations $(t_{k1}, \ldots, t_{kn_k})$ along a chromosome. We assume that for each platform, the data has been normalized to be centered at 0 for "normal" copy number and to have Gaussian (or near-Gaussian) noise. Actual data must be transformed with missing values imputed, sometimes with extreme outliers truncated in order to approximate Gaussian noise. In some studies, the "normal" diploid state of the genome is difficult to determine, such as when an entire chromosome has been amplified. When this occurs, other types of information, such as allelic ratios from SNP arrays, or intensity ratios from two-color aCGH experiments, will be needed to help assign the correct absolute copy number to each segment. Such complications are expected to affect all platforms. Here we deal with the integration of multiple platforms in detecting *changes* in CNV and only need to assume that the baseline "normal" state is shared in common across platforms.

The fact that all $\{\mathbf{y}_k : \ k = 1, \ldots, K\}$ are assaying the same biological sample implies that at any genomic location t there is only one true underlying copy number μ_t for all platforms. We define the observed intensity level for the i-th probe of the k-th platform consisting of a signal $f_k(\mu_{t_{k,i}})$ plus a noise term that has platform specific variance σ_k^2.

Specifically, we assume the following model for the data:

$$y_{ki} = f_k(\mu_{t_{k,i}}) + \epsilon_{k,i}, \tag{4.1}$$

where the noise term $\epsilon_{k,i}$ are independently distributed $N(0, \sigma_k^2)$. We call $f_k(\cdot)$, which quantifies the dependence of the observed intensity on the underlying copy number, the response function of platform k.

We model the true copy number as a piecewise constant function, i.e. constant within a segment, and yet may change to a different level at a "change-point". For a chromosome of length T, we assume that there exists a series of change-points $0 = \tau_0 < \tau_1 < \ldots, < \tau_m < T$ such that within each interval,

$$\mu_t = \theta_i, \quad t \in [\tau_i, \tau_{i+1}). \tag{4.2}$$

The magnitude parameters $\theta = (\theta_0, \ldots, \theta_m)$ and change-points $\tau = (\tau_1, \ldots, \tau_m)$ are all unknown and, like the response functions, must be estimated from the data.

For this paper, we assume that the response function is linear, i.e. $f_k(\mu) = r_k\mu$. The parameter r_k, which we call the response ratio, describes the ratio between the change in observed intensity for platform k and the underlying copy number. The linearity assumption allows for simple and intuitive test statistics and fast scanning algorithms.

While the linearity assumption is an oversimplified ideal situation, empirically the platform response functions are often observed to be approximately linear for low-amplitude changes. Response functions are usually nonlinear for high amplitude changes due to saturation effects. However, the high-amplitude changes usually have high statistical significance and are relatively less affected by this simplification in modeling. It is the low amplitude, statistically borderline cases where we expect to boost power through multi-platform integration.

When the platform specific response ratios r_k are known, the breakpoints τ and true copy numbers θ can be estimated through a likelihood based recursive segmentation procedure that builds on the conceptual foundations of Olshen *et al.* (10) and Vostrikova (12), which

we describe in Section 4.4.1. Conversely, when τ and θ are given, f_k can also be easily estimated using the procedures described in Section 4.4.4. Since both are usually unknown, we propose the iterative procedure described in Section 4.4.5.

4.4 Methods

4.4.1 Pooling Evidence by Weighted t-statistics

First consider the case where the goal is to test whether there is a CNV at a window from s to t. Under the *null* hypothesis that there is no CNV, the data within this region should have baseline mean $f_k(0) = 0$, i.e.

$$H_0: \quad y_{ki} \sim N(0, \sigma_k^2) \quad \text{for } k = 1, \ldots, K; \quad \text{and } i: \ s \leq t_{ki} < t. \tag{4.3}$$

If there is a gain (or loss) of magnitude μ, each platform should respond with signal $f_k(\mu) = r_k\mu$. The signal is a mean shift in a *common direction* for all platforms, with the observed magnitude of shift being $r_k\mu$ for platform k, i.e.

$$H_A: \quad y_{ki} \sim N(r_k\mu, \sigma_k^2) \quad \text{for } k = 1, \ldots, K; \quad \text{and } i: \ s \leq t_{ki} < t. \tag{4.4}$$

Since the generalized likelihood ratio statistic maximizes the power over all statistical tests for this model, we will use the likelihood based framework to test this hypothesis. Let $n_k(s,t) = |\{i : t_{k,i} \in (s,t]\}|$ be the number of probes from the k-th platform that falls within the interval $(s,t]$. Let $\bar{y}_{k,(s,t]}$ denote the mean intensity of probes that map within $(s,t]$, and similarly let $\bar{y}_{k,(s,t]^c}$ be the mean intensity of probes that map outside of $(s,t]$. It can be shown (see the Supplementary Appendices) that under this formulation, the log generalized likelihood ratio statistic is a weighted sum of platform specific terms:

$$Z(s,t) = \frac{\left[\sum_{k=1}^{K} \delta_{k,s,t} X_{k,s,t}\right]^2}{\sum_{k=1}^{K} \delta_{k,s,t}^2}, \tag{4.5}$$

where

$$X_{k,s,t} = \frac{\bar{y}_{k,(s,t]} - \bar{y}_{k,[1,n_k]}}{\sigma_k \sqrt{n_k(s,t)^{-1} - n_k^{-1}}}, \tag{4.6}$$

is the t-statistic for testing for a change in segment (s,t) using the data from platform k. The estimate of error standard deviation from platform k, $\hat{\sigma}_k$, can be obtained from the residuals after subtracting the mean within each segment. The weights

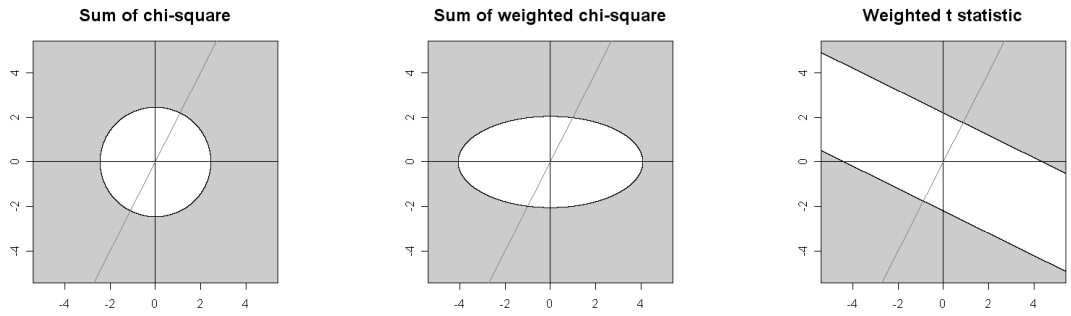
$$\delta_{k,s,t} = r_k \sqrt{n_k(s,t)}/\sigma_k \tag{4.7}$$

are proportional to the response ratio r_k, the square root of the number of probes from that platform that fall into $[s,t)$, and the inverse of the estimated error standard deviation $\hat{\sigma}_k$. When there is only one platform, the statistic (4.5) is equivalent to the chi-square statistic used in the Circular Binary Segmentation algorithm of Olshen *et al.* (10). Theoretical properties of scans using (4.6) and related statistics for a single platform were studied by Siegmund (11) and James *et al.* (5). Usually σ_k is unknown and must be estimated from the data as well, and we replace it with an estimate $\hat{\sigma}_k$ in (4.6) and (4.7). In the simplest case we assume a common variance for all probes of a given platform, the number of data points used to estimate σ is very large and thus $\hat{\sigma}_k$ is very precise and for all practical purposes can be treated as a known quantity. In situations where σ^k is dependent on the underlying copy number or differs between genomic regions, a generlized likelihood ratio statistic similar to (4.5) can also be computed.

Note that the statistic (4.5), which we call the *weighted t-statistic*, is different from the sum-of-chisquares statistic proposed in Zhang *et al.* (16) for multi-sample segmentation, where each sample comes from a different biological source assayed on the same experimental platform. The statistic used in Zhang *et al.* (16) is the sum of chi-square (SC) from N samples,

$$Z^{SC}(s,t) = \frac{1}{N} \sum_{n=1}^{N} X_{n,s,t}^2. \tag{4.8}$$

Intuitively, one may be tempted to extend the above formula to the multi-platform case by



a. b. c.

Figure 4.1: Comparison of the null hypothesis rejection regions between the sum of chi-square statistic (4.8), the weighted sum of chi-square statistic (4.9), and the weighted t-statistic (4.5) on $K = 2$ platforms. In all figures, the axes are the magnitudes of the X variables (4.6) for platforms 1 and 2. A significance level of 0.05 is used to determine the decision boundaries of all three statistics. For Figures (b) and (c), weights of $\delta_1 = 1$, $\delta_2 = 2$ are used. The diagonal line shows the direction of the weight vector $\delta = (\delta_1, \delta_2)$.

proposing a weighted form (SWC)

$$Z^{SWC}(s,t) = \frac{\sum_{k=1}^{K} \delta_{k,s,t}^2 X_{k,s,t}^2}{\sum_{k=1}^{K} \delta_{k,s,t}^2} \tag{4.9}$$

that does not treat all platforms equally. However, this approach has the drawback that it does not reward agreement between platforms. When pooling data across samples, independent biological specimen are not expected to carry the same CNV, and often both deletions and amplifications can be observed between the samples at the same genome location. Thus, the statistic (4.8) is intuitively correct in not "rewarding" agreement in direction of change between samples. For pooling data across platforms, however, the underlying CNV is the same, and the statistic in (4.5) correctly rewards agreement and penalizes disagreement. For example, consider the case of $K = 2$, where (4.5) simplifies to $(\delta_1^2 X_1^2 + \delta_2^2 X_2^2 + 2\delta_1\delta_2 X_1 X_2)/2$. If the signs of X_1 and X_2 agree, this statistic is always larger than (4.8), while if the signs disagree, it is smaller. This makes the weighted t-statistic more suitable for pooling evidence across multiple samples that come from the same biological source (Figure 4.1).

The difference between the three statistics is shown graphically in Figure 1 for the simple

case of two platforms with the response ratio of the second platform being twice that of the first platform. Note that all three statistics are functions of $X = (X_1, X_2)$, which, assuming that σ_k is known, is bivariate Gaussian with mean 0 and identity covariance matrix under the null hypothesis. Figures 1(a-c) show in gray the region in the (X_1, X_2) plane where the null hypothesis will be rejected. That is, X needs to fall in to the gray region to make a CNV call. For example, in Figure 1a, which depicts the situation in (4.8), the gray region is $\{X : \ Z^{SC}(X) > t_\alpha^{SC}\}$, where t_α^{SC} is a threshold chosen for the test to have significance level α. In Figure 1b, which depicts the situation in (4.9) the weights $\delta_2/\delta_1 = 2$ favor evidence from X_2 over evidence from X_1, giving an elliptical boundary. In Figure 1c, which depicts the situation in (4.5), the boundary of the rejection boundary is $\{X : \ \delta' X > t_\alpha\}$, which is perpendicular to the vector δ_2/δ_1. Importantly, note that (c) rewards agreement between the two platforms, while (a,b) treat all quadrants of the plane equally. The statistic (4.5, Figure 1c) also allows one platform to dominate the others: In the case where the directions disagree, e.g. in the upper left or lower right quandrants, the consensus can still be made according to the dominant platform.

4.4.2 Recursive Segmentation Procedure

In the previous section, we described the statistic used to test whether a specific interval $[s, t)$ constitutes a CNV. In reality, there can be multiple change-points in a chromosome copy number. To detect all change-points, we adopted a framework similar to Vostrikova (12), Olshen *et al.* (10), and Zhang and Siegmund (15). Vostrikova (12) proved the consistency of binary segmentation algorithms. Olshen *et al.* (10) proposed an improvement, called circular binary segmentation, that works better in detecting small intervals of change in the middle of long regions. Zhang and Siegmund (15) proposed a BIC criterion for deciding the number of segments. Both Olshen *et al.* (10) and Zhang and Siegmund (15) showed that these types of procedures work well on DNA copy number data. Two independent comparative reviews by Willenbrock and Fridlyand (13) and Lai *et al.* (7) concluded that the CBS algorithm of Olshen *et al.* (10) is one of the best performing single-platform segmentation methods. This motivated us to extend CBS to the case of multiple platforms.

The Multi-platform CBS (MPCBS) algorithm will be described in detail in the Supplementary Materials. Here, we give an intuitive overview using the following notation: Let $\mathcal{R}$ be an ordered set of segments $\{(i,j):\ 0<i<j<T\}$, and $\mathcal{Z}$ be the corresponding likelihood ratio statistics. Let M be the maximum number of change-points tolerated, which is usually determined by computational resources.

The algorithm proceeds as follows: S_k is the list of estimated change-points in the k-th iteration, which is initialized to contain only $\{0,T\}$. The entire dataset is scanned for the window $[s^*,t^*)$ that maximizes $Z(s,t)$, that is, where the evidence for a change is the strongest. This window is added to S_k. Then, the region (1) to the left of s^*, (2) between s^* and t^*, and (3) to the right of t^* are each scanned for a sub-segment that maximizes $Z(s,t)$, these maximum values are called Z_L, Z_C, and Z_R respectively. The corresponding locations of the maximum are R_L R_C, and R_R. These are kept in the ordered lists $\mathcal{Z}$ and $\mathcal{R}$. At each iteration k of the algorithm, the region whose maximum weighted t statistic is the largest, i.e. $i^* = \arg\max_i \mathcal{Z}[i]$, is determined. The change-points from that region that achieve this maximum, i.e. $(s^*,t^*) = \mathcal{R}[i^*]$, are added to S_k. Since s^*,t^* splits a previously contiguous region into three regions, $\mathcal{Z}$ and $\mathcal{R}$ must be updated to include the maximal Z values and maximizing change-points for the new regions to the left, center, and right of the new change points. This process is repeated until S_k has at least M change-points in addition to $\{0,T\}$. Finally, the modified BIC criterion described in the next section is used to determine a best estimate of the number of change-points and the final segmentation. The modified BIC is a theoretically proven method for estimating the true number of change-points based on asymptotic approximations to posterior model probabilities. It is an off-the-shelf method that automatically determines the trade-off between false positive and false negative rates. For users who wish to detect CNV using more or less stringent stopping rules, the software MPCBS allows the option of a user tunable z-score threshold for deciding the fineness of the segmentation.

4.4.3 Estimating the Number of Segments

To estimate the number of change-points, we use a modified form of the classic BIC criterion that extends the approach by Zhang and Siegmund (15). In Zhang and Siegmund (15), it was shown that the modified BIC, when used on top of the CBS procedure of Olshen *et al.* (10), improves its performance for DNA copy number data.

To describe the extension of Zhang and Siegmund (15) to the case of multiple platforms, we first define several quantities. For a given genome position t, let $n_k(t) = |\{i : t_{k,i} < t\}|$ be the number of probes in the region $[0, t)$ for platform k. Let

$$S_{k,t} = \sum_{i=1}^{n_k(t)} y_{k,i}$$

be the sum of the intensities of all probes in this region. For a given set of estimated change-points $\hat{\tau} = (\hat{\tau}_0 = 0 < \hat{\tau}_1 < \cdots < \hat{\tau}_k = T)$, let $\delta_{k,i} = r_k\sqrt{n_k(\hat{\tau}_i)}/\sigma_k$,

$$X_{k,i} = \frac{S_{k,\hat{\tau}_i} - n_k(\hat{\tau}_i)S_{k,\hat{\tau}_{i+1}}/n_k(\hat{\tau}_{i+1})}{\hat{\sigma}_k\sqrt{n_k(\hat{\tau}_i)[1 - n_k(\hat{\tau}_i)/n_k(\hat{\tau}_{i+1})]}},$$

and

$$U_i(\hat{\tau}) = \frac{\sum_{k=1}^K \delta_{k,i}X_{k,i}}{\left(\sum_{k=1}^K \delta_{k,i}^2\right)^{1/2}}.$$

$X_{k,i}$ is the t statistic for testing that the change in mean at $\hat{\tau}_i$ is not zero. $U_i(\hat{\tau})$ is a weighted sum of $X_{k,i}$, just as (4.5) is a weighted sum of (4.6). Let N be the total number of distinct values in $\{t_{k,i} : \ 1 \le k \le K, \ 1 \le i \le n_k\}$, that is, the number of different probe locations from all K platforms. For any natural number n, $n!$ denotes the factorial of n. It is possible to show using arguments similar to Zhang and Siegmund (15) that

$$\frac{1}{2}\sum_{i=1}^m U_i(\tau)^2 - \frac{1}{2}\sum_{i=0}^m \log\left[\sum_{k=1}^K n_k(\hat{\tau}_i, \hat{\tau}_{i+1})\right] - \log\frac{N!}{m!(N-m)!}. \tag{4.10}$$

is asymptotically within an $O_p(1)$ error term of the Bayes factor for comparing the model with k change-points versus the null model. The number of change-points should be selected to maximize the BIC.

The first term of the modified BIC is the maximized likelihood, and is thus the same as the first term of the classic BIC criterion. The second and third terms are penalties that increase with the number of change-points. The second term penalizes the θ parameters by summing up the logarithm of the effective sample size for estimating each θ_i. The third term is the logarithm of the total number of ways to select m change-points from N possible values, which penalizes the change-points parameters τ.

With the modified BIC, there is no need for a user specified p-value threshold. The trade-off between false-positive and false-negatives is automatically decided by the modified BIC.

4.4.4 Estimating the Platform-Specific Response Ratio

In this section we discuss the situation where the segmentation is known, and we would like to estimate the platform specific response ratios $r = (r_1, \ldots, r_K)$, the baseline levels $\alpha = (\alpha_1, \ldots, \alpha_K)$, and the underlying copy numbers $\theta = (\theta_1, \ldots, \theta_m)$. For each $(\hat{\tau}_i, \hat{\tau}_{i+1})$, the data from platform k that fall within the segment can be used to obtain an estimate of $f_k(\theta_i)$:

$$\hat{f}_{k,i} = n_k(\hat{\tau}_i, \hat{\tau}_{i+1})^{-1} \sum_{j: t_{k,j} \in [\tau_i, \tau_{i+1})} y_{k,i}, \tag{4.11}$$

For each i and k, $\hat{f}_{k,i} \sim N(f_k(\theta_i), v_{k,i})$, where $v_{k,i} = \sigma_k^2 / n_k(\hat{\tau}_i, \hat{\tau}_{i+1})$ is proportional to the noise variance of the k-th platform and inversely proportional to the number of probes in that platform that lies in the i-th segment. Thus, the negative log-likelihood of the data is

$$\frac{1}{2} \sum_{i=0}^{m} \sum_{k=1}^{K} v_{k,i}^{-1} (\hat{f}_{i,k} - \alpha_k - r_k \theta_i)^2. \tag{4.12}$$

The unknown parameter vectors r and θ should be chosen to minimize the above weighted sum of squares.

If the variances $v_{k,i}$ were identical across i and k, r and θ can be estimated through the singular value decomposition of the matrix $F = (f_{i,k})$ or through a robust approach such as median polish. This model would then be similar to those proposed in Irizarry *et al.*

(4) and Li and Wong (8) for model-based probe set summary of Affymetrix Genechip data. However, the differences in variances should not be ignored, because segments with less data, for which we are less sure of the mean estimate, should be down-weighted. Similarly, platforms with higher noise variance should also be down-weighted compared to platforms with smaller noise.

There are many ways to modify existing approaches to minimize (4.12). We take the following simple iterative approach: Note that for any fixed value of r, the corresponding minimizer $\hat{\theta}(r)$ can be found through a weighted least squares regression. The same is true if we minimize with respect to r when the value of θ is held fixed. Thus, joint optimization of r and θ is achieved through a simple block update procedure which we detail in the Supplementary Materials.

4.4.5 Iterative Joint Estimation

Sections 4.4.1-4.4.3 detail a method for segmenting the data when the platform-specific signal response functions are known. Then, Section 4.4.4 describe a method for estimating the response functions with the segmentation given. In most cases both the segmentation and the response functions are unknown. The algorithm below is an iterative procedure that jointly estimates both quantities from the data.

Multi-platform Joint Segmentation.

Fix stopping threshold ε. Initialize $f_k^{(0)}(\mu) = \mu$ for $k = 1, \ldots, K$. Set $i \leftarrow 0$.

1. Estimate the segmentation $\tau^{(i)}$ using MPCBS assuming response functions $f^{(i)}$.
2. Estimate $f^{(i+1)}$ as described in Section 4.4.4 assuming the segmentation $\tau^{(i)}$.
3. If $\|f^{(i+1)} - f^{(i)}\| < \varepsilon$, exit loop and report:

$$\hat{\tau} = \tau^{(i)}, \quad \hat{f}_k = f_k^{(i)}, \quad k = 1, \ldots, K.$$

otherwise, set $i \leftarrow i + 1$, and iterate.

In the algorithm above, $f_k^{(i)}$ and $\tau^{(i)}$ are respectively the response function and the segmentation estimated in the i-th iteration. The response functions are initialized to be equal in all platforms, a setting which in most cases already gives a decent segmentation. After the first iteration, the estimated segmentation can be used to obtain a more accurate estimate of the response functions, which can then be used to improve the segmentation. In all of our computations we simply set the stopping parameter $\varepsilon = 0.01$. The estimates of f_k stabilized within a few iterations for all of the HapMap samples analyzed in Section 4.5.1.

4.5 Results

4.5.1 Comparison with Single Platform CBS by Using HapMap Data

We applied our approach to the eight HapMap samples analyzed in Kidd *et al.* (6) using fosmid clone end-sequencing. In addition, we also analyzed the reference genotype data for the same eight samples from an Agilent platform over 5,000 common copy number variants (2). We combined the fosmid and Agilent datasets and collectively referred to them as reference CNVs. The same HapMap samples have both been analyzed by Illumina 1M Duo and Affymetrix 6.0 genotyping chips. We used MPCBS to combine the two platforms in making joint CNV calls, and compared these calls with those made by running CBS on each individual platform separately. We also compared MPCBS results with the union of CBS calls made on both platforms, as well as the intersection of CBS calls made on both platforms. Details of data normalization are described in the Supplementary Materials. For CBS analysis, we show results using a range of p-values from 0.0001 to 0.1. For MPCBS, we show results using both the modified BIC based stopping criterion described in this paper, as well as a range of z-score thresholds from 4.5 to 9. We assessed performance by computing, for each method and stopping threshold, the fraction of calls made by the method that is also reported in Kidd *et al.* (6) or Conrad *et al.* (2) (precision), and the fraction of CNVs reported in these two references that were also detected by the method (recall).

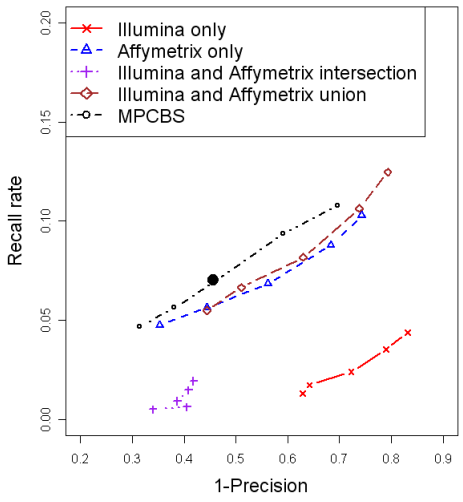


Figure 4.2: Precision-recall curve for detection of CNVs in eight HapMap samples. The methods being compared are (I) CBS on Illumina platform only, (II) CBS on Affymetrix platform only, intersection of (I) and (II), union of (I) and (II), and MPCBS jointly on Illumina and Affymetrix. The solid black dot is the result given by MPCBS using the modified BIC stopping criterion. The horizontal axis is the fraction of calls made by the given method that fails to overlap with a reference CNV (1-precision). The vertical axis is the fraction of all reference CNVs that are discovered by the given method (recall). The curves are obtained by varying the stopping thresholds of CBS and MPCBS.

When the fosmid and Agilent platforms detect a CNV, the boundaries of the CNV are not precisely defined. We therefore defined concordance to be any overlap between a CNV called by CBS/MPCBS and a reference CNV. When multiple calls made by CBS/MPCBS overlapped with the same reference CNV, only one of them was counted as concordant. This guarded against over-segmented CNV regions. This criteria of overlap can be made more or less stringent, but as long as it is applied consistently in the comparison between CBS and MPCBS, the conclusion made would be unbiased. Figure 4.2 shows the curves of 1-precision versus recall. We see from these results that concordance with reference is low across all methods. The low concordance with fosmid detected CNVs has also been reported previously, see, for example Cooper *et al.* (5) and McCarroll *et al.* (9). Importantly, at comparable levels of precision, MPCBS gives higher recall rates than either Affymetrix or Illumina does alone, and higher recall rates than combining calls from the two platforms by intersection or union. In general, Affymetrix discovers many more segments than Illumina,

with many more concordant calls, likely due to having more probes than the Illumina chip.

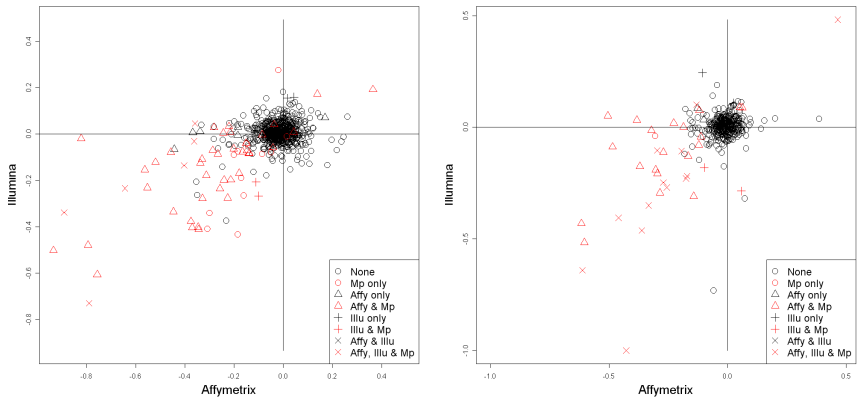


Figure 4.3: Mean probe intensities within reference CNV calls for Affymetrix versus for Illumina in samples NA18956 and NA12878. The points are colored and shaped based on the combination of Affymetrix, Illumina, the integrated method that detected it.

Is the low concordance between Affymetrix, Illumina, and reference CNVs due to inherent disagreement in the raw data, or low sensitivity or specificity of the statistical method? To investigate this issue, for each reference CNV, we computed the mean intensity of the Affymetrix or Illumina probes mapping within each reference CNV. We would expect that if the absolute change in mean probe intensity is high for a given platform, and if the segment spans a sufficient number of probes, the CNV is more likely to be also called by that platform. Alternatively, if the mean probe intensity within the reference CNV is indistinguishable from baseline, it would be missed by that platform. Figure 4.3 shows the Affymetrix versus Illumina mean intensity plot for two of the eight samples. Each point corresponds to a reference CNV. The points colored in red are reference CNVs also detected by MPCBS, i.e. overlapping one of the CNVs called by MPCBS. The shapes of the points reflect whether they are detected by single platform CBS in none of the individual platforms alone, in only Affymetrix, in only Illumina, or in both Affymetrix and Illumina. Most of the reference CNVs do not have a shift in intensity in any platform, suggesting that the microarray based assays are noisy and prone to cross hybridization, especially in repetitive regions or regions with complex rearrangements (5). By combining information from the Affymetrix and Illumina platforms, MPCBS is able to make calls that were not identified

in either platform alone.

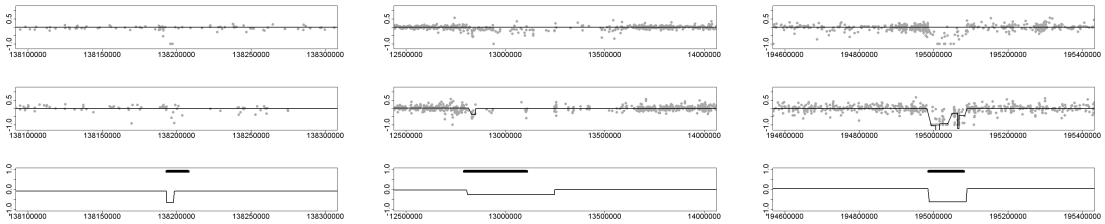


Figure 4.4: Examples of regions detected by MPCBS. For each panel, the top plot shows the Illumina data with CBS fit, the middle plot shows the Affymetrix data with CBS fit, and the bottom plot shows the MPCBS consensus estimate along with thick horizontal lines depicting the reference CNV call. In the left panel, the detected CNV region contains too few probes and is thus missed by CBS on both Affymetrix and Illumina platforms alone. However, by pooling the information from both platforms, MPCBS is able to make a call. Similarly, in the middle panel, neither platform alone has strong signal, but with pooled evidence the MPCBS call has good agreement with the reference. The right panel shows that CBS has the tendency to over-segment CNV regions. The problem is mitigated in MPCBS coupled with the modified BIC stopping criterion.

Figure 4.4 shows four examples of CNV calls made by multi-platform CBS that is missed by one or both of the individual platforms. In the first two examples shown in the top left and top right panels, the number of probes in each platform is too few to make a call. However, combining the two platforms, multi-platform CBS makes a call that partially overlaps with a reference. In the examples on the bottom left and bottom right panels, multi-platform CBS improves on the boundaries of the Affymetrix call.

4.5.2 TCGA Cancer Data

To provide an example of application to somatic CNVs, we analyze a data set from The Cancer Genome Atlas (TCGA) samples. Intensity data from three platforms, Illumina 550 K, Affymetrix 6.0 and Agilent 244K were downloaded from TCGA data portal. The segmentation result for CBS and MPCBS on Chromosome 7 of the data is shown in Figure 4.5. The top three panels show the results for the standard approach, which is to call CNVs for each platform separately. But to integrate the three CBS data sets one is faced with the difficulty that for a true underlying CNV, the three segmentation summaries may not have all detected the CNV, and even when they do, they will report different

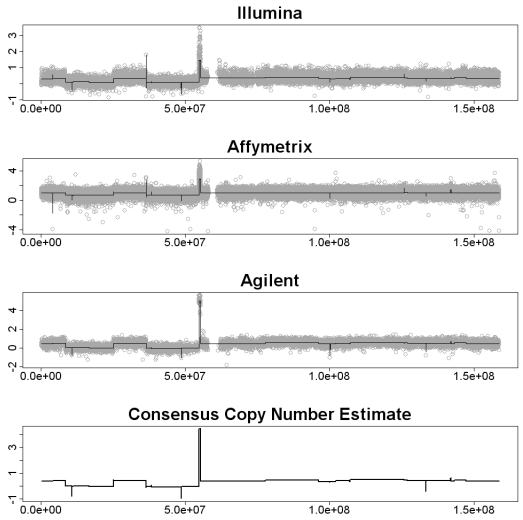


Figure 4.5: Result of MPCBS on a TCGA sample. The top three plots show Illumina, Affymetrix, and Agilent data with CBS fit. Bottom panel shows multi-platform consensus.

magnitudes, different boundaries and different degrees of uncertainty. The MPCBS result in the bottom panel provides a natural consensus estimate without the problem of having to decide how to integrate the three CBS segmentation results. While MPCBS provides a single combined estimate, it remains a statistically constructed best-possible summary, and cannot be automatically taken as evidence of technical replication. We emphasize that crucial results in specific regions still require careful validation in further experiments.

4.5.3 Computing Time

The computation was done on a 1.6 GHz Intel Core 2 Duo processor. In the analysis of this example region, which contains 30,170 Illumina probes, 98,993 Affymetrix probes, and 13,241 Agilent probes, MPCBS took 122 seconds for each iteration of steps 1-3 in Section 3.5. The algorithm converged in two iterations. Extrapolating to the full data set consisting of ~ 2 million Affymetrix probes, > 1 million Illumina probes, and 240K Agilent probes, the computing time is on the order of 1 CPU-hour per sample, and fluctuates according to the number of CNVs detected. In general, computing time scales with the number of samples linearly, and with the number of probes N as $N \log N$. We have implemented additional speed-up algorithms that are documented in the `R` package.

4.6 Discussion

We have proposed a model for the joint analysis of DNA copy number data coming from multiple experimental platforms. Under simplifying assumptions, the maximum likelihood framework can lead to an easily interpretable statistic and a computationally tractable algorithm for combining evidence across platforms during segmentation. By comparing to Agilent and fosmid clone end-sequencing data on eight HapMap samples, we showed that MPCBS gives more accurate copy number calls, as compared to a simple intersection or union of the calls made by CBS separately on each platform. This method has also been applied to TCGA data, where it provides consensus copy number estimates that provide a natural summary of data from Affymetrix, Illumina, and Agilent platforms.

A main feature of MPCBS is that it combines scan statistics from multiple platforms in a weighted fashion, thus without requiring pre-standardization across different data sources. For a given underlying copy number change, platform A may report a higher level of absolute change in signal intensity than platform B, but if A also shows a higher level of noise, or fewer probes in the genomic region in question, the scan statistics of A may not be larger than those of B because such statistics are scaled appropriately within each platform before being combined in MPCBS. However, careful normalization and standardization across platforms are still desirable when running MPCBS. This is because while segmentation per se is not sensitive to absolute signals of different platforms, the mean level of change reported by MPCBS can still be sensitive to the scale of different platforms. Recently, Bengtsson *et al.* (1) proposed a joint normalization method for bringing different platforms to the same scale and for addressing the issue of non-linear scaling between platforms. While the method of Bengtsson et al. is not concerned with joint segmentation, it can be coupled to MPCBS so that the mean level of copy number change reported by MPCBS is an even better approximation of the consensus level of change. We expect that the segmentation result will alter slightly when using data pre-processed by the method of Bengtsson et al. mainly because the current version of MPCBS has not considered non-linear response functions. In short, we recommend pre-standardization of the scale of copy number changes across

platforms before running MPCBS. This would have little impact on segmentation but may improve the mean copy number change reported.

MPCBS can be applied also to the situation when a biological sample is assayed multiple times on the same experimental platform. In our general specification of the model (2.1), we allow the snps/clones from different assays to overlap. When the same platform is used for repeated assays of the same sample, model (2.1) and the MPCBS algorithm still apply without modification. This does not assume that technical replicates using the same platform have the same signal response curves, as there may be differential quality in replicates due to differing handling and hybridization conditions. However, the user can have the option of constraining the response ratios to 1 if the samples have already been preprocessed, e.g. using the method of Bengtsson et al. (2009) to equalize the signal magnitudes.

4.7 Supplementary Materials

4.7.1 Derivation of the Likelihood Ratio Statistic (4.5)

To show that the likelihood ratio statistic gives (4.5): For simplicity of notation we suppress the location indices $[s, t]$. Since this is a Gaussian mean shift model, the log likelihood ratio between H_A and H_0 is

$$l_A(\mu) - l_0 = \sum_{k=1}^{K} [\mu \delta_k X_k / \sigma_k - \mu^2 \delta_k^2 / (2\sigma_k^2)]. \tag{4.13}$$

Differentiating the above with respect to μ and setting the derivative to 0, we get $\hat{\mu} = \tilde{\delta}' X / \tilde{\delta}' \tilde{\delta}$, where $\tilde{\delta} = (\delta_1/\sigma_1, \ldots, \delta_K/\sigma_K)$. Plugging this value back into (4.13), we have $l_A(\hat{\mu}) - l_0$ equals $(\tilde{\delta}' X)^2 / (2\tilde{\delta}' \tilde{\delta})$, which is one-half of (4.5).

4.7.2 Pseudo-code for MPCBS Segmentation Algorithm

Initialize:

Set $k \leftarrow 0$, $S_0 \leftarrow \{0, T\}$,

$$Z_{\max} = \max_{0<i<j<T} Z(i,j), \quad (s^*, t^*) = \arg\max_{0<i<j<T} Z(i,j),$$

Set $\mathcal{Z} \leftarrow Z_{\max}$, $\mathcal{R} \leftarrow (s^*, t^*)$, $BIC(0) \leftarrow 0$.

While $|S_k| - 2 < M$ **repeat:**

1. Let $i^* \leftarrow \arg\max_i \mathcal{Z}[i]$, $(s^*, t^*) \leftarrow \mathcal{R}[i^*]$,

$$s \leftarrow \max\{i \in S_k, i < s^*\}, \quad t \leftarrow \min\{i \in S_k, i > t^*\}.$$

   For each of $(i,j) \in \{[s, s^*), [s^*, t^*), [t^*, t)\}$, compute

$$Z_{\max} = \max_{i<a<b<j} Z(a,b), \quad (s^*, t^*) = \arg\max_{i<a<b<j} Z(a,b).$$

   Let Z_L, Z_C, and Z_R be respectively the value of $Z_{\max}$ computed for the left segment $[s, s^*)$, the center segment $[s^*, t^*)$, and the right segment $[t^*, t)$. Similarly, let R_L, R_C, R_R be respectively the maximizer for the left, center, and right segments.

2. Let $L = |\mathcal{Z}|$, Set:

$$k \leftarrow k + 1,$$

$$S_k \leftarrow S_{k-1} \bigcup \{s^*, t^*\},$$

$$\mathcal{Z} \leftarrow \{\mathcal{Z}[1 : i^* - 1], Z_L, Z_C, Z_R, \mathcal{Z}[i^* + 1, L]\},$$

$$\mathcal{R} \leftarrow \{\mathcal{R}[1 : i^* - 1], R_L, R_C, R_R, \mathcal{R}[i^* + 1, L]\}.$$

   Set $BIC(k)$ to be the BIC criterion (4.10) of the estimated change-points S_k.

Finally, let $k^* = \arg\max_{0 \le k \le M} BIC(k)$**. Return** S_{k^*}**.**

4.7.3 Block-update procedure for estimating platform response ratio

Let K be the number of platforms, m be the number of regions. We are fitting

$$\frac{1}{2}\sum_{i=0}^{m}\sum_{k=1}^{K} v_{k,i}^{-1}(\hat{f}_{i,k} - \alpha_k - r_k\theta_i)^2 \tag{4.14}$$

with the response ratio r_K for platform K constrained to be 1.

Initialize $t \leftarrow 0$,

$$r^0 \leftarrow (1, \ldots, 1)_{1\times K},$$

$$\alpha^0 \leftarrow (0, \ldots, 0)_{1\times K}.$$

Repeat:

1. $t \leftarrow t + 1$
2. Given r^{t-1}, estimate by weighted least squares

$$\theta^t \leftarrow \arg\min_{\theta} \sum_{i=0}^{m}\sum_{k=1}^{K} v_{k,i}^{-1}(\hat{f}_{i,k} - \alpha_k^t - r_k^{t-1}\theta_i)^2.$$

3. Given θ^t, estimate by weighted least squares

$$(\alpha_{1:K-1}^t, r_{1:K-1}^t) \leftarrow \arg\min_{\alpha,r} \sum_{i=0}^{m}\sum_{k=1}^{K-1} v_{k,i}^{-1}(\hat{f}_{i,k} - \alpha_k - r_k\theta_i^t)^2.$$

4. For the K-th platform, keep r_K^t at 1 and set $\alpha_K^t \leftarrow \sum_{i=1}^{m}(\hat{f}_{i,K} - \theta_i)$.
5. If $\|r^t - r^{t-1}\|/m < \epsilon$ exit loop.

Report $r = r^t$, $\theta = \theta^t$, $\alpha = \alpha^t$.

4.7.4 Normalization of Hapmap samples

For Affymetrix data, we requested GeneChip 6.0 CEL files for HapMap samples from Affymetrix, InC. We used the software package Aroma to analyze preprocess the eight

HapMap samples used in this study. The resulting logR values were used in CBS and MPCBS analysis without further normalization. For Illumina Human1M-Duo Beadchip data, we downloaded from Illumina's public FTP site the logR values for the 270 HapMap samples. We first median-centered logR values for each sample and in each chromosome. To correct for the long range oscillations in baseline logR levels (a phenomenon known as "genomic waves", which is related to local GC content and whole-genome amplification conditions prior to array hybridization), we developed a normalization procedure based on a principal component analysis (PCA) in the entire cohort of 270 samples. Each sample is assessed PCA scores that correspond to the magnitude and phase of the "wave" for that sample. For each SNP we performed a linear regression of logR values in all samples against the samples' first three PCA scores. The residuals from the regression were taken as logR values corrected for the "genomic waves", and used for segmentation by CBS or MPCBS.

Bibliography

[1] Bengtsson, H., Ray, A., Spellman, P., and Speed, T. (2009). A single-sample method for normalizing and combining full-resolution copy numbers from multiple platforms, labs and analysis methods. Bioinformatics, 25, 861867.

[2] Conrad, D. F., Pinto, D., Redon, R., Feuk, L., Gokcumen, O., Zhang, Y., Aerts, J., Andrews, T. D., Barnes, C., Campbell, P., Fitzgerald, T., Hu, M., Ihm, C. H., Kristiansson, K., MacArthur, D. G., MacDonald, J. R., Onyiah, I., Pang, A. W., Robson, S., Stirrups, K., Valsesia, A., Walter, K., Wei, J., Tyler-Smith, C., Carter, N. P., Lee, C., Scherer, S. W., and Hurles, M. E. (2009). Origins and functional impact of copy number variation in the human genome. Nature, advance online publication.

[5] Cooper, G. M. M., Zerr, T., Kidd, J. M. M., Eichler, E. E. E., and Nickerson, D. A. A. (2008). Systematic assessment of copy number variant detection via genome-wide snp genotyping. Nature Genetics, 40, 11991203.

[4] Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level data. Biostatistics, 4(2), 249264.

[5] James, B., James, K., and Siegmund, D. (1987). Tests for a change-point. Biometrika, 74, 7183.

[6] Kidd, J. M., Cooper, G. M., Donahue, W. F., Hayden, H. S., Sampas, N., Graves, T., Hansen, N., Teague, B., Alkan, C., Antonacci, F., Haugen, E., Zerr, T., Yamada, A. N., Tsang, P., Newman, T. L., Tu zu n, E., Cheng, Z., Ebling, H. M., Tusneem, N., David, R., Gillett, W., Phelps, K. A., Weaver, M., Saranga, D., Brand, A., Tao, W., Gustafson, E., Mckernan, K., Chen, L., Malig, M., Smith, J. D., Korn, J. M., Mccarroll, S. A., Altshuler, D. A., Peiffer, D. A., Dorschner, M., Stamatoyannopoulos, J., Schwartz, D., Nickerson, D. A., Mullikin, J. C., Wilson, R. K., Bruhn, L., Olson, M. V., Kaul, R., Smith, D. R., and Eichler, E. E. (2008). Mapping and sequencing of structural variation from eight human genomes. Nature, 453(7191), 5664.

[7] Lai, W. R., Johnson, M. D., Kucherlapati, R., and Park, P. J. (2005). Comparative analysis of algorithms for identifying amplifications and deletions in array cgh data. Bioinformatics, 21, 37633770.

[8] Li, C. and Wong, W. H. (2001). Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection. PNAS, 98(1), 3136.

[9] McCarroll, S. A. A., Kuruvilla, F. G. G., Korn, J. M. M., Cawley, S., Nemesh, J., Wysoker, A., Shapero, M. H. H., de Bakker, P. I. W. I., Maller, J. B. B., Kirby, A., Elliott, A. L. L., Parkin, M., Hubbell, E., Webster, T., Mei, R., Veitch, J., Collins, P. J. J., Handsaker, R., Lincoln, S., Nizzari, M., Blume, J., Jones, K. W. W., Rava, R., Daly, M. J. J., Gabriel, S. B. B., and Altshuler, D. (2008). Integrated detection and population-genetic analysis of snps and copy number variation. Nature Genetics, 40, 11661174.

[10] Olshen, A. B., Venkatraman, E. S., Lucito, R., and Wigler, M. (2004). Circular binary segmentation for the analysis of array-based dna copy number data. Biostatistics, 5, 557572.

[11] Siegmund, D. O. (1988). Approximate tail probabilities for the maxima of some random fields. Annals of Probability, 16, 487501.

[12] Vostrikova, L. (1981). Detecting disorder in multidimensional random process. Soviet Mathematics Doklady, 24, 5559.

[13] Willenbrock, H. and Fridlyand, J. (2005). A comparison study: applying segmentation to arraycgh data for downstream analyses. Bioinformatics, 21, 40844091.

[14] Zacks, S. (1983). Survey of classical and bayesian approaches to the change-point problem: Fixed sample and sequential procedures in testing and estimation. In Recent Advances in Statistics, pages 245269. Academic Press.

[15] Zhang, N. and Siegmund, D. (2007). A modified bayes information criterion with applications to the analysis of comparative genomic hybridization data. Biometrics, 63, 2232.

[16] Zhang, N., Siegmund, D., Ji, H., and Li, J. Z. (2009). Detecting simultaneous change- points in multiple sequences. Biometrika, in press.

CHAPTER V

Conclusion

5.1 High-throughput array platforms and next-generation alternatives

In this thesis, I investigated how biological information can be derived from high-dimensional datasets such as gene expression microarrays and array comparative genomic hybridization in *multiple-sample*, *multiple-time-point*, and *multiple-platform* settings. Even though these large-scale datasets have fueled advances in various biomedical contexts in the last decade, it is important to acknowledge their limitations alongside their strengths.

- **Top-down vs. bottom-up approaches:** Before the advent of high-throughput technologies, traditional 'bottom-up' approaches in biology focused on individual functional units. This approach had the potential to uncover greater details about the functional unit under study compared with the global 'top-down' approaches of large-scale datasets. However, it also bore a misinterpretation risk due to the lack of a wider context that can only be provided by global information. Thus, the top-down approaches that are able to provide comprehensive information about biological status and function in a time and cost-efficient manner have increased appeal. However, the top-down approaches also have limitations since a global view can miss important details that might be essential to build the '*story*' about the underlying biology. Hence, it is essential to improve the ability to integrate bottom-up and top-down

approaches (1). The number of large-scale datasets is increasing at an accelerated pace, but these datasets provide little value unless they are of sufficient quality and are interpreted in the richest possible context.

- **Noise structure and resolution:** Raw intensity values from gene expression microarrays and array CGH have two sources of noise: (1) biological and (2) technical. Technical noise can be defined as all noise that do not arise due to the biological sample under study. Inaccuracies related to technical equipment, variability in the nucleic acid hybridization and amplification procedures can all be sources of technical noise.

  In gene expression microarrays, the commonly encountered lack of reproducibility in prognostic signatures across platforms and laboratories is frequently attributed to the noise levels in this technology (2). The Microarray Quality Control (MAQC) Consortium generated data from the Affymetrix GeneChip platform to more carefully investigate the technical noise level in this type of data. Reports show that the variability in microarray data caused by technical noise is low and its role in statistical methodology of data analysis is far from critical (3). However, it is not possible to ensure that this conclusion applies to all biological contexts and questions of interest studied with microarray technology. The variability in hybridization efficiencies and binding propensities of nucleic acids, cross-hybridization background among millions of array probes, as well as the saturation of probe sequences can prevent the platform from reflecting an accurate representation of the specimen's biological properties. Moreover, similar studies on other gene expression platforms have yet to be carried out.

  More recent transcriptome sequencing (RNA-Seq) technologies provide an appealing alternative to microarray platforms. Initial studies show that, in terms of overall technical performance, RNA-Seq is the technique of choice for studies that require accurate estimation of absolute transcript levels (4). Microarrays can still be preferable when a lower sensitivity is acceptable since they provide results in a faster and

less expensive fashion.

Array CGH data are also subject to variability due to technical noise. We have shown in Chapter 4 with an example in Figure 4.5 that the noise structure is different across platforms. We have also shown in Figure 4.3 that most of the reference CNVs are not discovered by the array CGH platforms. These results underline the fact that microarray based assays are noisy and prone to cross hybridization, especially in repetitive regions or regions with complex rearrangements (5).

Next-generation high-throughput sequencing methods (DNA-Seq) are providing an alternative for CNV detection as well (7). Two variants of CNV detection methods using high-throughput sequencing are based on paired-end read mapping and event-wise testing using read-depth of coverage, with examples in (6) and (8) respectively. The authors found that, compared to microarrays, both sequencing-based approaches are much more sensitive for detecting copy number variations smaller than 1,000 base pairs, which make up the majority of CNVs (8).

5.2 DynBoost for reverse engineering of gene regulatory networks

In Chapter 2 of this thesis, I introduced DynBoost, a new method for reverse-engineering of gene regulatory networks using time-series gene expression data. DynBoost is a fast and flexible algorithm based on nonlinear autoregressive models and outperforms competing algorithms. However, I showed in Tables 2.2 and 2.3 that network predictions of 100-node networks have considerably lower performance that those of 10-node networks. Even though competing algorithms exhibit the same type of decline in performance for 100-node networks, it is not obvious whether this low performance is due to the *data* or the *algorithms*.

- **Algorithms:** At a community level, algorithms for reverse-engineering gene regulatory networks may not be competent enough in correctly modeling the interactions between genes and gene products. (9) have identified certain types of *systematic* errors in network predictions that continue to challenge modeling efforts. These errors,

namely the fan-in, fan-out, cascade, and feed-forward loop (FFL) errors, arise from co-regulation or chain-regulation among a set of nodes in the network. Even if the data values have no technical noise, correctly sieving out pairwise regulations from such joint regulation settings may not be possible without introducing conditional distributions into the models. Assuming a model includes proper conditional distributions, the identifiability of pairwise regulations would remain to be questionable without a large enough sample size.

Further, DynBoost as well as many other algorithms, rely on interaction assumptions that may be biologically over-simplistic. For instance, DynBoost models interactions between time t and $t + 1$, but we do not have sufficient reason to believe that all regulatory interactions in a cell become identifiable in the same short-term interval. The effect of a regulatory interaction can become apparent over a wider time span due to the particular regulation mechanism or certain constraints in the complex environment of the cell. While modeling nonlinear interactions with first-order interactions is important, it can miss longer-term interactions. A second pass of the algorithm with higher-order interactions would provide valuable information about regulatory mechanism that generate transcripts over a wider time span.

- **Data:** The technical noise associated with gene expression arrays as mentioned above may be a limiting factor in the success of reverse-engineering methods. Because of the challenge of having to identify pairwise interactions from a complex system, even a small noise level can hinder the correct inference of regulatory mechanisms. Data from next-generation sequencing technologies (RNA-Seq) carry promise in this regard for increasing the performance of existing methods. It would be partiularly interesting to ask whether gene expression data with lower noise levels than microarrays would help solve some of the challenges related to fan-in, fan-out, cascade, and FFL errors. Competitive methods in the community should be tested with transcript levels obtained from RNA-Seq platforms that are known to be less noisy than microarray platforms.

Knock-out and knock-down data that entail removing or reducing the effects of single genes respectively have greater power to detect regulatory interactions. In these cases, it is possible to observe the effect of changing one variable on the whole system. Since only one variable is knocked-out or knocked-down, the changes in expression levels have large enough magnitude to surpass the noise associated with the microarray platform. However, obtaining knock-out and knock-down data from real biological systems is expensive and time-consuming. Community-wide initiatives to generate RNA-Seq data from model organisms with knocked-out or knocked-down genes will be highly valuable for increasing the success with reverse-engineering methods. This type of large-scale data can also be reused a high number of times to understand various biological mechanisms beyond gene regulatory networks.

Potential future directions for DynBoost include modifications on the use of (1) residuals and (2) the subnetwork S_m in the algorithm. As the first point, we note that the residuals of genes are given equal weights during the optimization procedure. However, the algorithm can be modified to use residuals with different weights as in the spirit of AdaBoost. Each iteration of DynBoost would then adapt to discover interactions between genes whose expression time series is poorly estimated by the past iterations.

DynBoost also has the limitation that potential regulators for a gene at iteration m can only be chosen from the subnetwork S_m. Since S_m is chosen to be small in the spirit of a weak learner, the algorithm itself significantly restricts the number of potential regulators that will be explored. As a remedy, the number of boosting iterations can be increased, but this is not desirable as it would lead to overfitting of the data. In future work, the construction of S_m can be modified to allow all genes to be explored as potential regulators.

5.3 Consensus clustering and unsupervised class discovery

Asking an investigator how many classes a genomic dataset has can be analogous to asking a museum-goer how many parts a painting has. The answer certainly depends on the person's own understanding of what a part is, and also on how close he or she is viewing

the painting. In unsupervised class discovery, real high-dimensional datasets are often times like a painting in that classes are not readily discernible or well-defined, and their number depends on the investigator's own preferences for the modeling scheme, the partitioning method and pertinent parameters.

Even before a partitioining method is chosen and applied, the investigator needs to specify what modeling scheme he or she is going to use. One of the main decisions to be made in this regard is whether to use **flat** *vs.* **nested (hierarchical)** models. If prior knowledge about the underlying biology implies substructure within classes, a nested model should be preferred. (10) presents an example of a nested classification for gene expression data from cancers of unknown primary (CUP) origin. Nested models have also been proposed for single cancer types such as glioma (11).

Another modeling question that the investigator needs to address before the implementation of partitioning is whether to use soft *vs.* hard clustering. In soft clustering, also known as *fuzzy* clustering, objects that are clustered do not belong to a single cluster, but rather have partial membership in multiple clusters. In contrast, hard clustering algorithms assign each object to only one of the clusters. Partial membership models in situations with poorly separated clusters is not only a viable alternative, but also one that needs to be carefully examined alongside hard clustering methods. The correlation of the clustered objects with phenotypic outcomes may prove to be better explained by a gradient of partial memberships rather than hard cluster assignments.

In both hard and soft clustering techniques, it is possible to categorize the objects as **core** *vs.* **non-core**. The objects closest to cluster centroids in hard clustering, and the objects with partial membership coefficients above a certain threshold in soft clustering can be classified as core objects, which means they constitute a set of objects that best characterize the cluster they are in or have strongest membership to. The profiles of these core objects can then be used to train parameters for a supervised classification scheme. Future objects with unknown cluster membership can be tested with these parameter estimates to infer the probabilities with which they belong to each cluster. These probabilities, in turn, can be used either directly as partial membership coefficients, or to find the maximum-probability

cluster in a hard clustering setting. Consensus matrix values in consensus clustering can be used as a tool to separate core objects from non-core ones. Objects with high values can be clustered separately to group similar samples in the same class.The parametric nature of supervised classification would allow for significance estimates and higher interpretability of classes, as well as comparebility of results across different datasets. A combination of unsupervised and supervised approaches in this way would also be expected to increase power in class discovery efforts.

Integrating '*significance*' estimates into unsupervised methods is an important point that has not received sufficient attention in the class discovery literature. Comparing results from the real dataset with those from a null distribution should be a fundamental component of every unsupervised class discovery attempt. In this context, class discovery needs to be viewed as a two-part question. The significance levels for (1) the existence of structure and (2) the number of clusters should be investigated separately with null datasets that are specific to each one of these two questions. The first part of the question, namely the existence of structure in the dataset, should be explored with a global null distribution whose sole function is to allow the decision of *structure vs. no-structure*. The second part of the question, the number of clusters in the dataset, requires a set of null datasets that are study-specific and lead to the most significant results for the optimal number of clusters. Significance for the first question should be viewed as a visa to move on to the second question, not as a guarantee for the significance of the second question. In the case of significance for the first question, the null datasets for the second question still need to be able to distinguish the number of clusters that attain the highest significance levels. A visual inspection of partitioning results for a set of number of clusters, and choosing the one that looks best should not be sufficient to report an optimal number.

The contribution of gene-gene correlations in a dataset to the level of structuredness should also always be born in mind during class discovery. Most real gene expression datasets have a significant gene-gene correlation structure due to functionally similar genes being expressed at similar levels. This correlation structure can affect the high-dimensional localization of the samples as a source of bias for partitioning methods. Therefore, gene-gene

correlations should be included in global null datasets when determining the significance level for the structuredness of the data. Validation of classes in external datasets is another point where this correlation structure has potential to mislead investigators. The particular placement in high-dimensional space of the most discriminant genes for a certain number of classes can be significantly affected by the gene-gene correlations in the data. Thus, these genes will have a similar localization in an external dataset simply due to gene-gene correlations, rather than sample structuredness. It is important to account for gene-gene correlations as much as possible in class discovery so as to discover the structure in the data that stems only from samples.

Consensus clustering was developed to be able to visualize the stability of classes as well as infer the optimal number of classes in gene expression data. I applied consensus clustering to random data and observed that many unsupervised partitioning methods are able to divide homogeneous data into prespecified numbers of clusters; and consensus clustering shows apparent stability of such *chance partitioning of random data*. The results I presented suggest that consensus clustering needs to be applied with caution as it is prone to over-interpretation. If samples are not well-separated, consensus clustering could lead one to conclude apparent structure when there is none, or declare cluster stability when it is subtle. In the presence of genuine structure, this method may be a powerful tool for keeping false negatives at low levels, but in the exploratory phase of many studies can lead to false positives if not compared to a suitably formed null distribution. When partitioning poorly separated samples from a real dataset, it is necessary to objectively evaluate the evidence of clustering, and if appropriate, consider alternative models such as nested models, partial membership models or continuous distributions.

5.4 MPCBS for joint estimation of DNA copy number from multiple platforms

MPCBS presents a solution for the challenge of joint DNA copy number estimation from multiple array CGH platforms. It can be applied also to the situation when a biological

sample is assayed multiple times on the same experimental platform. Due to the advent of high-throughput sequencing technologies, investigators now have more alternatives to interrogate samples for DNA copy number. Even though the application of high-throughput sequencing methods for CNV detection is also divided into multiple categories such as paired-end read mapping and event-wise testing, these do not differ from each other as fundamentally as the different array CGH platforms. In that sense, the introduction of high-throughput sequencing as an alternative for CNV detection is exptected to lead to less variable calls for a given copy number event, hence unifying results in effect. The necessity to make joint estimation of DNA copy number from multiple platforms may be diminished, however, array CGH platforms should co-exist with high-throughput sequencing technologies for an extended amount of time either as complementary or only validation techniques.

5.5 Closing remarks

The data deluge we are experiencing today is revolutionizing the world. There is an abundance of data in virtually all sectors of life, be it private or government, corporate or individual, consumption or production. Biology is not an exception to this trend. In fact, biology is one of the fields at the forefront of this revolution. The rapid advent of new technologies one after the other has facilitated the generation of bulk amounts of data in a fast and cheap way. These large-scale data fill databases in an unprecedented pace. However, the value of these datasets will not be realized until they are interpreted effectively in the richest possible context. In this thesis, I focused mainly on two technologies: gene expression microarrays and array comparative genomic hybridization (aCGH). I showed how the different dimensions of gene expression datasets can be analyzed to understand the relationships among genes and among biological specimens. I also showed how different platforms for the same technology, namely aCGH, can be used jointly to make an inference for the DNA copy numbers in the genome. The more recent high-throughput sequencing technologies RNA-Seq and DNA-Seq provide a potentially more sensitive alternative to

the gene expression and copy number detection analysis respectively. However, both gene expression microarrays and array CGH are expected to co-exist with the newer technologies for the time being as a source of validation and/or complementary information.

Bibliography

[1] Pinkel, D., Albertson, D.G. (2005) Array comparative genomic hybridization and its applications in cancer. *Nat Genet.* 37 Suppl:S11-7. Review.

[2] Marshall E. (2004) Getting the noise out of gene arrays. *Science*;306:630631. doi: 10.1126/science.306.5696.630

[3] Klebanov L, Yakovlev A. (2007) How high is the level of technical noise in microarray data? *Biol Direct*; 2:9. doi: 10.1186/1745-6150-2-9.

[4] Fu X, Fu N, Guo S, Yan Z, Xu Y, Hu H, Menzel C, Chen W, Li Y, Zeng R, et al. (2009) Estimating accuracy of RNA-Seq and microarrays with proteomics. *BMC Genomics*, 10:161.

[5] Cooper, G. M. M., Zerr, T., Kidd, J. M. M., Eichler, E. E. E., and Nickerson, D. A. A. (2008). Systematic assessment of copy number variant detection via genome-wide snp genotyping. Nature Genetics, 40, 11991203.

[6] Bickhart DM, Hou Y, Schroeder SG, Alkan C, Cardone MF, Matukumalli LK, Song J, Schnabel RD, Ventura M, Taylor JF, Garcia JF, Van Tassell CP, Sonstegard TS, Eichler EE, Liu GE. (2012). Copy number variation of individual cattle genomes using next-generation sequencing. *Genome Res* Feb 2. [Epub ahead of print]

[7] Xie, C. and Tammi, M.T. (2009) CNV-seq, a new method to detect copy number variation using high-throughput sequencing. *BMC Bioinformatics*, 10:80

[8] Yoon, S., Xuan, Z., Makarov, V., Ye, K., Sebat, J. (2009) Sensitive and accurate detection of copy number variants using read depth of coverage. *Genome Res.*1586-92.

[9] Marbach, D., Prill, R.J., Schaffter, T., Mattiussi, C., Floreano, D., Stolovitzky, G. (2010) Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286-6291

[10] Shedden, K.A., et al. (2003) Accurate molecular classification of human cancers based on gene expression using a simple classifier with a pathological tree-based framework. *Am. J. Pathol.*;163:19851995.

[11] Li, A. et al. (2009). Unsupervised Analysis of Transcriptomic Profiles Reveals Six Glioma Subtypes. *Cancer Res.* 69(5): 20912099.