Understanding Pathways

Donny Soh

Imperial College London

2009

UNDERSTANDING PATHWAYS

DONNY SOH
(B.Eng. (Hons.), NUS)

PROFESSOR YIKE GUO
PROFESSOR LIMSOON WONG

A THESIS SUBMITTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTING
IMPERIAL COLLEGE LONDON

2009

Abstract

The challenge with todays microarray experiments is to infer biological conclusions from them. There are two crucial difficulties to be surmounted in this challenge:(1) A lack of suitable biological repository that can be easily integrated into computational algorithms. (2) Contemporary algorithms used to analyze microarray data are unable to draw consistent biological results from diverse datasets of the same disease.

To deal with the first difficulty, we believe a core database that unifies available biological repositories is important. Towards this end, we create a unified biological database from three popular biological repositories (KEGG, Ingenuity and Wikipathways). This database provides computer scientists the flexibility of easily integrating biological information using simple API calls or SQL queries.

To deal with the second difficulty of deriving consistent biological results from the experiments, we first conceptualize the notion of "subnetworks", which refers to a connected portion in a biological pathway. Then we propose a method that identifies subnetworks that are consistently expressed by patients of he same disease phenotype.

We test our technique on independent datasets of several diseases, including ALL, DMD and lung cancer. For each of these diseases, we obtain two independent microarray datasets produced by distinct labs on distinct platforms. In each case, our technique consistently produces overlapping lists of significant nontrivial subnetworks from two independent sets of microarray data. The gene-level agreement of these significant subnetworks is between 66.67% to 91.87%. In contrast, when the same pairs of microarray datasets were analysed using GSEA and t-test, this percentage fell between 37% to 55.75% (GSEA) and between 2.55% to 19.23% (t-test). Furthermore, the genes selected using GSEA and t-test do not form subnetworks of substantial size. Thus it is more probable that the subnetworks selected by our technique can provide the researcher with more descriptive information on the portions of the pathway which actually associates with the disease.

Keywords: pathway analysis, microarray

Acknowledgement

I thank my supervisors, Prof Limsoon Wong and Prof Yike Guo for their guidance, patience and faith. I am grateful to my colleagues Cen Lee, Guang Hui Seah, Difeng Dong, Chek Tien Tan and Boon Kiat Quek for their help and guidance. I thank my family members and friends for their support. I am also appreciative of A*STAR for funding my studies under a A*STAR-Imperial College Programme scholarship and a A*STAR SERC grant 072 101 0016.

Contents

List of Figures

List of Tables

List of Abbreviations

3NF	Third Normal Form
ALL	Acute Lymphoblastic Leukemia
AML	Acute Myelogenous Leukemia
ANOVA	ANalysis Of VAriance between groups
API	Application Programming Interface
cDNA	complementary DeoxyriboNucleic Acid
DAG	Directed Acyclic Graph
DMD	Duchenne Muscular Dystrophy
DNA	DeoxyriboNucleic Acid
FC	Fold Change
FCS	Functional Class Scoring
FDR	False Discovery Rate
GNEA	Gene Network Enrichment Analysis
GO	Gene Ontology
GPML	GenMAPP Pathway Markup Language
GSEA	Gene Set Enrichment Analysis
JSON	JavaScript Object Notation

KEGG	Kyoto Encyclopedia of Genes and Genomes
MSE	Mean Squared Error
MLL	Mixed Lineage Leukemia
NCBI	National Center for Biotechnology Information
NEA	Network Enrichment Analysis
ORA	OverRepresentation Analysis
PCR	Polymerase Chain Reaction
SAM	Significance Analysis of Microarrays
SBML	Systems Biology Markup Language
SCC	Significant Connected Components
SOAP	Simple Object Access Protocol
SPIA	Signaling Pathway Impact Analysis
SSE	Sum of squares
SSTR	Treatment Sum of Squares
XML	eXtensible Markup Language

CHAPTER 1

Introduction

This dissertation is concerned with making consistent biological inferences from microarray experiments and prior biological information. Although technological limitations in such experiments have been considerably reduced over the past decade, there still exists challenges in providing proper diagnosis and understanding, part of the challenges arising from our incomplete knowledge of the functions of biological systems (Berger and Iyengar, 2009). This dissertation first argues that contemporary biological repositories seriously limit the availability of such information to computational methods. Such limitations will be all the more evident as microarray experiments evolve in size and complexity. This dissertation next advocates incorporating biological information into algorithms to support microarray analysis, and demonstrates this concept with a working algorithm.

1.1 Motivation

The challenge with todays microarray experiments is to be able to infer biological conclusions from them. This challenge has traditionally been tackled as a pure computational problem. Hence solutions (Choe et al., 2005; Tusher et al., 2001) generally entail running experimental data through algorithms and selecting statistically significant genes. This approach, albeit mathematically sound, ignores the biological motivations behind the microarray experiment. Such an analysis usually leaves the investigator with a large list of genes and information correlated mathematically but unwieldy for biological inferences. In addition, most of these techniques strongly rely on an arbitrary p-value cutoff used to select differentially expressed genes (Sivachenko et al., 2005).

To obtain results both of high accuracy as well as biological relevance, there is therefore a need to incorporate insights (Pavlidis et al., 2002; Subramanian et al., 2005) from additional biological repositories into our investigations. This allows us to complement the mathematical results with biological background, providing the clinician with relevant biological results.

However even these contemporary techniques (which integrate biological information) share a number of limitations. First, many of these techniques rely only gene sets present within biological processes and pathways, ignoring the different intricate connections and topology within pathways. For example (Tarca et al., 2008) showed that such topological details can be extremely crucial. In particular, the authors stated that within the insulin pathway, if the insulin receptor (INSR) is not present, the entire pathway will be shut off. Hence in the scenario where some of the genes appear to be differentially expressed within the insulin pathway but INSR is not present, this might mean that the genes differentially expressed are not connected to one another and might not affect the pathway significantly.

Some algorithms (Liu et al., 2007) utilize global protein connections which ignore the fact that majority of proteins participate in multi-domain processes (Liu et al., 2009) and such global process may not translate similarly locally to smaller pathways (Sivachenko et al., 2007).

Lastly, these algorithms only consider individual pathways or gene sets as a whole. This ignores the fact that many times, only a portion of a gene network might be significantly perturbed (Sohler et al., 2000) and this portion might be ignored if its proportion within the network is not large enough. This happens especially when the gene set is extremely large.

1.2 Goal of Thesis

The goal of this thesis is to make consistent biological inferences from microarray experiments and prior biological information. Intuitively, the thesis appears viable for the following reasons:

- The biological repository created is currently being used in a research environment for tasks such as extracting genes or gene relationships from pathways.
- The designed algorithm has been tested with four different diseases (eight unique datasets, two datasets for each disease) and we are able to find consistent biological information independently derived from the dataset pair from each disease.

These observations lead directly to the thesis statement:

> **Incorporating accurate and detailed biological information can improve the consistency and reproducibility of biological inferences for similar diseases across datasets.**

1.3 Contributions

Our thesis has the following two main contributions:

1. Creation of an aggregated biological pathway database, providing the database with a unified access method (through API calls) and standardizing data formats. This aggregation of data reduces the scenario of conflicting or missing data from solitary pathway databases. The unification of access methods and data formats allows easy access to researchers who wish to access data from diverse databases.
2. Conception and development of a technique for microarray analysis which provides descriptive biological analysis to microarray data. Results from our microarray technique provides the researcher not only with the genes or pathways that are differentially expressed, it directly gives a clear representation of the intricate

relationship between such differentially expressed genes within their biological pathways. In addition, results from our techniques have been verified empirically, showing consistent results across different databases of the same disease. This gives us additional confidence that our technique is able to identify relevant pathways because they are consistently significant across diverse datasets.

1.4 Publications

We had the honor of working with diverse research groups during the course of this dissertation. These collaborative efforts have been significant contributions to this thesis and are listed as publications below. Publication I has been heavily expounded in Chapters 2 and 3. Much of the work in Chapters 3, 4 and 5 revolves around Publication III while Publication IV contains work from Chapters 6 and 7. Publication II involves a real life clinical example where we employed some microarray analysis on patients suffering from NPC (Nasopharyngeal Cancer) (we mention this briefly in Chapter 2). Finally Publication V stands out independently as a theoretical framework that could be employed in the future for subnetwork analysis.

I Donny Soh, Difeng Dong, Yike Guo, Limsoon Wong. Enabling More Sophisticated Gene Expression Analysis for Understanding Diseases and Optimizing Treatments. ACM SIGKDD Explorations, 9(1):3-14, June 2007.

II Wen-Son Hsieh, Ross Soo, Bee-Keow Peh, Thomas Loh, Difeng Dong, Donny Soh, Limsoon Wong, Simon Green, Judy Chiao, Chun-Ying Cui, Yoke-Fong Lai, Soo-Chin Lee, Benjamin Mow, Richie Soong, Manuel Salto-Tellez, Boon-Cher Goh. Pharmacodynamic Effects of Seliciclib, an Orally Administered Cell Cycle Modulator, in Undifferentiated Nasopharyngeal Cancer. Clinical Cancer Research, 15(4):1435–1442, February 2009.

III Donny Soh, Difeng Dong, Yike Guo, Limsoon Wong. Consistency, Comprehensiveness, and Compatibility of Pathway Databases, 23 September 2009 (2nd revision).

IV Donny Soh, Difeng Dong, Yike Guo, Limsoon Wong. Finding Consistent Disease Subnetwork Across Datasets. Manuscript.

V Jinyan Li, Haiquan Li, Donny Soh, Limsoon Wong: A Correspondence Between Maximal Complete Bipartite Subgraphs and Closed Patterns. PKDD 2005: 146-156.

1.5 Organization of the Thesis

This thesis focuses on developing techniques whereby consistent biological analysis can be made. To achieve this, we subdivide the task into two smaller portions. (1) the modeling, implementation and realization of a unified biological database and (2) a technique that uses this unified database to draw consistent biological conclusions from independent datasets. This thesis is organized as:

- Background: Algorithm

  Microarrays have made it technologically feasible for researchers to measure the expression levels of thousands of genes simultaneously. Through microarray analysis techniques, researchers are able to understand the behavior of individual genes. This chapter concerns itself with scrutinizing some popular analysis techniques often employed to analyze microarrays and review their ability of providing consistent and descriptive biological results to microarray experiments. We conclude the chapter by suggesting that contemporary methods are unable to generate consistent and descriptive biological results because (1) they place a greater emphasis on computational models rather than descriptive biological analysis and (2) whatever biological information used is either too fine (individual genes) or too coarse (entire gene sets) for use in biological descriptive reasonings.

- Background: Pathway API

  One of the critical assumptions underlying microarray techniques which integrates biological information (biological pathways) as a priori data into their computational analysis is the availability of a collection of well-curated biological repositories (for example, Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa and Goto, 2000), Ingenuity (Ingenuity, 1998)). We review several representative biological databases and analyze their suitability to be used as a priori biological information for computational microarray analysis. Our survey findings conclude that contemporary biological databases are very diverse, lacking in complete data and not suitable for integration into computational algorithms. These reasons lead us to create our own unified database cache and its API for easy integration into computational algorithms.

- Pathway API — Methods

  Because of the lack of consistency, comprehensivity and compatibility of current databases (shown in the previous chapter), we decided to create our own unified database cache and its API. This chapter explains the technical challenges and techniques we employed in designing and creating our own biological repository. We pay special attention to achieving data comprehensiveness by integrating biological data from three representative biological repositories, achieving data consistency through the standardization of nomenclature, data formats and lastly compatibility through the easy data accessibility of a web http API.

- Pathway API — Evaluation

  This chapter evaluates our database (both quantitatively and qualitatively) to ensure that it is consistent, comprehensive and compatible for integration into computational microarray algorithms. Quantitatively we show that, as we expected, there is a low level of gene and gene pairwise consistency among the three different biological databases (KEGG, Ingenuity, Wikipatways). Qualitatively, because we (1) unified pathway data from three independent biological sources, (2) created consistent data access methods and formats, and (3) standardized nomenclature such as gene references and pathway key features, researchers will be able to access complete information (from different databases) easier by using a single access method (PathwayAPI) and standardized nomenclature.

- Disease and Drug-Response Pathway Identification — Algorithm

  Here we demonstrate the different decision making steps taken to ensure that the results obtained from our technique will provide proper biological analysis to microarray experiments. We first explain how we create the foundation to allow biologically descriptive results by deciding on a proper level of granularity required within the biological pathways of the unified database. Following which we explain how we arrange genes into their respective pathway components to take advantage of the gene-gene relationships within pathways. This is further substantiated with an explanation of statistical testing (which is required to test for significant pathway components). Finally, these are strung together with a detailed explanation and example of our algorithm.

- Disease and Drug-Response Pathway Identification — Results

  We compare our technique with several popular methods of microarray analysis such as SAM (Tusher et al., 2001), t-test (Cui et al., 2005) and GSEA (Subramanian et al., 2005) in this chapter. This comparison is made on four different disease types and eight different datasets (Leukemia (Armstrong et al., 2002; Golub et al., 1999), Leukemia Subtypes (Ross et al., 2004; Yeoh et al., 2002), DMD (Haslett et al., 2002; Pescatori et al., 2007), Lung Cancer (Bhattacharjee et al., 2001; Garber et al., 2001)).

  We show that our technique generates significant subnetworks and genes that are more consistent across datasets as compared to the other methods (GSEA, t-test and SAM). The large size of subnetworks which we generate indicates that these subnetworks are more biologically significant (less likely to be spurious). To validate our results, we show that most of our genes from the generated subnetworks have also been considered significant by the t-test. In addition, we have chosen two sample subnetworks and validated them with references from biological literature. This shows that our algorithm is capable of generating descriptive biologically conclusions.

Chapter 2

Background Study on Microarray Analysis

Chapter Synopsis

Summary

Microarrays have made it technologically feasible for researchers to measure the expression levels of thousands of genes simultaneously. Through microarray analysis techniques, researchers are able to understand the behaviour of individual genes and widely adopted in many areas to detect changes in gene activity. Traditionally, microarray data is analysed via individual genes, identifying changes in specific genes against certain conditions. While this approach works well in identifying genes which react the most significantly, it neglects the underlying biological mechanisms causing such gene activity changes. Such biological mechanisms are the result of interactions between multiple genes within a biological pathway. Hence the challenge with todays microarray analysis is being able to infer such biological conclusions consistently. Here we scrutinise some popular analysis techniques often employed to analyse microarrays and review their ability of providing consistent and descriptive biological results to microarray experiments.

Conclusions

Our literature review seems to suggest that contemporary methods are unable to generate consistent and descriptive biological results because (1) they place a greater emphasis on computational models rather than descriptive biological analysis and (2) biological information used is either too fine (individual genes) or too coarse (entire gene sets) for use in biological descriptive reasonings. For ease of discussion, we have segregated contemporary techniques into three different categories (based on their methodology): (1) individual gene testing, (2) gene pathway testing and (3) gene class testing.

Techniques (Tusher et al., 2001) which belong to our first category of algorithms (individual genes testing) often ignore biological motivations behind the microarray experiment, usually leaving the investigator with unwieldy gene lists which makes it difficult to draw biological conclusions. Gene pathway testing techniques attempt to organise and infer genetic networks from microarray data. This process is challenging because of the large number of samples as compared to patients making such inferences suffer from a high incidence of false positives, creating biological pathways with limited biological foundations.

The last category of techniques, gene class testing techniques have attempted to alleviate these issues by integrating biological information within the analysis. Yet most of these techniques merely integrate biological information on a level of granularity which is too coarse for descriptive biological inferences (Sivachenko et al., 2007; Subramanian et al., 2005). For example, the output of these algorithms is normally a set of gene groups (pre-divided based on their functions) and selected based on the overexpression of its individual genes. This manner of selecting overexpressed pathways ignores the presence of intricate connections and topology. Such rigidity also causes these algorithms to be unable to detect overexpressed gene groups if they occur only within a portion of the gene group.

2.1 Introduction

Humans have tens of thousands of genes, and the development of DNA microarrays by Patrick O. Brown, Joseph DeRisi, David Botstein, and colleagues in the mid-1990s made it possible to examine the expression of thousands of genes at once (Hoopes, 2008). Since then, microarray technology has evolved and we are now able to embed the entire human genome unto a single microarray.

In a similar fashion, many techniques for making microarray analysis have emerged. From this wealth of techniques for identifying significant differential gene expression, we categorise them into three approaches; viz., individual genes, gene pathways and gene classes approaches. In this chapter we present the approaches below:

1. **Individual Gene Testing**

   These techniques highlighted in (Baldi and Long, 2001; Cheng and Church, 2000; Golub et al., 1999; Tusher et al., 2001) search for individual genes that are differentially expressed. Traditionally, this approach consists of running the experimental values either through statistical clustering or probabilistic techniques, such as the fold change, t-test and Significance Analysis of Microarrays (SAM) (Tusher et al., 2001). The SAM test is currently the most prevalent test for testing of differential expressed genes within microarrays. The output of such algorithms is a list of genes that are deemed differentially expressed.

2. **Gene Pathway Testing**

   Methods of this genre attempt to infer biological information from data without using pre-existing biological information. Bayesian learning (Friedman et al., 2000) and Boolean network learning (Lähdesmäki et al., 2006) are representatives of this approach. In this approach, the researcher obtains a set of connected gene networks inferred solely from the gene expression data. While possible pathway relationships can be obtained from such algorithms, it is more probable that false positives – relationships which correlates statistically but have zero biological relevance – might be present. This refers to Type I (false positive) errors. It happens when the expression level of totally unrelated genes display high correlation with each other. Such spurious correlations occur mainly due to the large number of genes (relative to the number of patient samples available for analysis) that are being tested.

3. **Gene Class Testing**

   These techniques test how gene classes behave as a whole. These techniques either pre-process or post-process their information with existing biological background knowledge to guide their analysis of the microarray data. Examples include over-representation analysis (ORA) (Khatri and Draghici, 2005), Functional Class Scoring (FCS) (Goeman et al., 2004), GSEA (Subramanian et al., 2005), GNEA (Liu et al., 2007) and ErmineJ (Pavlidis et al., 2004b). Results from such methods are normally a list of pathways or gene groups that are differentially expressed according to the algorithms.

The commonly acknowledged challenge of these techniques is obtaining replicable results. For instance, in differentially expressed gene discovery, there should be a substantial overlap in the gene lists from different datasets of the same disease. This is inferred from the premise that similar underlying conditions cause the onset of certain diseases. However it has been shown that there is little concurrence among such gene lists (Ein-Dor et al., 2005; Michiels et al., 2005; Zhang et al., 2009).

For example, (Zhang et al., 2009) demonstrated this inconsistency using SAM. For a pair of datasets involving prostate cancer (Lapointe et al., 2004; Singh et al., 2002), he calculated the percentage overlap of differentially expressed genes between them. The top 10 genes had a percentage overlap of 30% while the top 100 genes had a percentage overlap of 15%. The same calculations were repeated for lung cancer (Bhattacharjee et al., 2001; Garber et al., 2001) and DMD (Haslett et al., 2002; Pescatori et al., 2007) datasets, yielding similar low percentages.

In addition, the functional gene lists, pathways or classes determined by such methods do not provide sufficient descriptive information about the interplay and relationship of genes (Soh et al., 2007). Hence the generated hypotheses are usually too general, rendering them ineffective in guiding further research and treatment (Dong et al., 2009).

The remainder of this section explores the individual algorithms from the three classes in greater detail.

2.2 Individual Gene Testing

Many methods exist for the identification of differentially expressed genes between conditions. The interested reader can refer to (Allison et al., 2006; Madeira and Oliveira, 2004) for a large range of different techniques. Yet despite such a large plethora of methods, biologists show a particular affinity for the two earliest approaches: fold change (FC) and t-statistics. Their pervasiveness can be largely attributed to their simplicity. Here we provide details of how the FC, modified FC, t-test, modified t-test algorithms function. We will also provide details on SAM, one of the popular contemporary algorithms for finding differentially expressed genes from microarray results. The algorithms reviewed in detail here are:

1. Fold Change
2. t-test
3. SAM
4. LIMMA

2.2.1 Fold Change

There are two definitions of fold change in literature. The standard formula for fold change is simply the number of times the expression level of a gene has increased/decreased by (Tusher et al., 2001). It is given as follow:

$$FC_{ratioi} = \frac{x_i}{y_i} \tag{2.1}$$

where FC_{ratio_i} is the fold change for gene i, x_i and y_i are the raw expression values of the gene i in the control and treatment samples respectively. The second definition defines fold change from a difference in expression values point of view (Choe et al., 2005). The formula is:

$$FC_{diff_i} = x_i - y_i \tag{2.2}$$

It is noted that these two forms of fold change analyse slightly different forms of changes. The FC_{ratio_i} emphasises more on percentage changes whereas FC_{diff_i} emphasises absolute changes. The usage of either metrics will depend largely on the data. In general, if the data has extremely low values, the latter formula might be more useful because a small increase in the absolute raw value might indicate too large a percentage increase. Conversely, if the data has a good range of values, using the former one would seem relevant because it would be able to show us the significance of the change in value.

2.2.2 t-test

There are three popular variants of the statistical t-test generally used in microarray experiments. The basic two-sample t-test is given by the formula:

$$T_i = \frac{\hat{x_i} - \hat{y_i}}{s_i} \tag{2.3}$$

where T_i refers to the t-statistic, $\hat{x_i}$ and $\hat{y_i}$ refers to the mean log_2 expression of genes i in the control and treatment respectively. s_i thus refers to the standard error of these replicates for gene i.

However we find that such a calculation will yield a large variance for the values for T_i. This is especially so if the values of the expression values are very small. This is exacerbated due to the large number of genes compared to the small sample size. Hence to ensure that the variance of T_i is independent of the gene expression, a small positive constant is usually added to s_i. This is the technique used in the SAM algorithm calculation and in SAM, the value s_o is chosen to minimize the variance value.

$$T_i = \frac{\hat{x_i} - \hat{y_i}}{s_i + s_o} \tag{2.4}$$

The last variant of the t-test involved tweaking the value of s_i, a technique known as variance shrinkage (Baldi and Long, 2001; Cui et al., 2005). It yields the formula below:

$$T_i = \frac{\hat{x_i} - \hat{y_i}}{\sqrt{Bs^2 + (1-B)s_i^2}} \tag{2.5}$$

where s is the predicted variance of T_i based on all genes in the array and s_i is the estimated variance of T_i based only on that single gene. Thus when B is equal to 0, it will be the standard t-test statistic. Using this weighted combination of variance allows information to be borrowed across genes.

2.2.3 Significance Analysis of Microarrays (SAM)

The different variants of the t-tests above provide a t-score to test the probability of that gene being significant by coincidence. For instance, if we assign a p-value threshold of 0.05, all genes with a p-value of less than 0.05 will have their null hypothesis rejected. This means that these genes will be declared as significant. The value of 0.05 means that these genes have a 5% chance of being a false positive, known also as a Type I error.

Seen in this light, the t-test basically argues that because the probability of a Type I error happening by chance is so low, the experiment has confidence that the event did not occur by coincidence. Indeed, a p-value of 0.01 or even 0.05 is very suitable for such usage in normal circumstances. However, in the context of microarray experiments, because we are doing multiple hypothesis testing of each gene within the microarray experiment, we would probably be doing the hypothesis test 10,000 times (once for each gene). Using a p-value cutoff of 0.01, it would mean that we are expected to get 100 Type I errors / false positives, which would probably be too large a number to ignore.

This issue with the t-test leads to the development of Significance Analysis of Microarrays, or SAM (Tusher et al., 2001), where genes are tested based on the change in gene expression relative to the standard deviation of repeated measurements for that gene and genes greater than a certain threshold are deemed to be significant. We will describe how the number of Type I errors can be controlled using a technique known as False Discovery Rate (FDR) (Benjamini and Yekutieli, 2001).

1. Firstly, the calculation of the relative difference $d(i)$ of the gene i is given by the expression:

   $$d(i) = \frac{\bar{x}_I(i) - \bar{x}_{II}(i)}{s(i) + s_o} \tag{2.6}$$

   where $\bar{x}_I(i)$ and $\bar{x}_{II}(i)$ are the average levels of the expression values in Phenotypes I and II respectively for gene i. The term $s(i)$ in the denominator is known as the gene-specific scatter and the term s_o is a positive constant to prevent genes whose expression is near zero (and hence unreliable) from having large scores $d(i)$. The gene specific scatter is given by the formula:

   $$s(i) = \sqrt{a\{\sum_{m}^{M}[x_m(i) - \bar{x}_I(i)]^2 + \sum_{n}^{N}[x_n(i) - \bar{x}_{II}(i)]^2\}} \tag{2.7}$$

   where M refers to the total number of patients in phenotype I, N refers to the total number of patients in phenotype II and $a = (1/m + 1/n)/(m + n - 2)$. $\sum_m^M$ and $\sum_n^N$ are thus the summation of the calculations in phenotypes I and II respectively.

2. We calculate the value of the relative difference $d(i)$ for all genes p in the microarray experiment and order these statistics according to the magnitude of the $d(i)$ values such that $\hat{d}(1)$ was the largest relative difference, $\hat{d}(2)$ was the second largest

relative difference, and $\hat{d}(i)$ was the i^{th} largest relative difference. This is shown by the equation below in Equation 2.8

$$\hat{d}(1) < \hat{d}(2) < ... < \hat{d}(p) \tag{2.8}$$

3. Next, we generated a large number of controls by first making B number of permutations of the phenotype labels within the microarray experiment, and computing the relative differences from these permutations by repeating the steps above. The corresponding order statistics is therefore computed for each single permutation. This means that we will permute the M and N labels between all the patients from phenotypes I and II. (Hence during the each permutation sequence, some of the patients from phenotype I will be labeled as from phenotype II and vice versa. However we maintain the portion of labels for each phenotype, M labels for phenotype I and N labels for phenotype II).

4. From each permutation b, we compute the order statistics $\hat{d}^b(i)$ such that $\hat{d}^b(1) < \hat{d}^b(2) < ... < \hat{d}^b(p)$. From this set of permutations, we estimate the expected order statistics of each single gene by calculating the expected mean of the gene rank by the formula in Equation 2.9

$$\hat{d}^B(1) = (1/B)\sum_{1}^{B}\hat{d}^b(i) \tag{2.9}$$

5. We plot the values of $\hat{d}(i)$ against $\hat{d}^B(i)$ and a threshold Δ which will be used to detect genes above or below a certain threshold. An example of this plot is seen in Figure 2.1 (image reproduced from (Tusher et al., 2001)). Specifically, we will begin from the origin of the graph, move up to the right to find the first gene j_1 where $\hat{d}(i) - \hat{d}^B(i) > \Delta$. All genes past j_1 will then be considered as significantly positive. We call this list of genes G_p (p for positively significant genes).

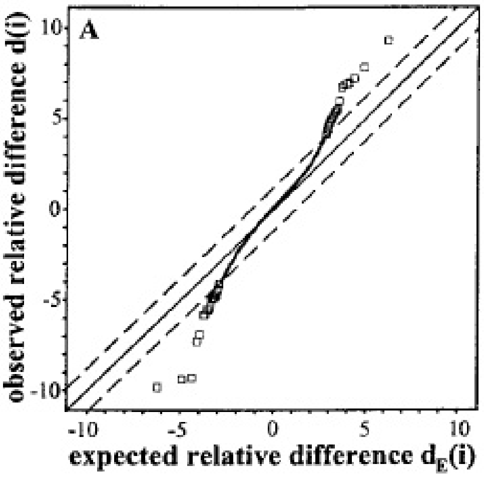


Figure 2.1: Sample plot of the observed relative difference. The solid line represents the expected mean of the gene rank while the dotted lines refer to upper and lower thresholds. The circles refer to the corresponding genes which fall above/below the threshold. Image reproduced from (Tusher et al., 2001).

6. To find the significantly negative genes we will start from the origin, move down to the left and find the first gene k_1 such that $\hat{d}^B(i) - \hat{d}(i) > \Delta$. All genes past k_1 will then be considered as significantly negative. This list of genes will be considered as G_n (n for negatively significant genes).

7. Therefore the combined lists, G_p and G_n from this technique gives us the list of genes which are deemed significant as compared to the rest of the genes.

8. Finally, we can calculate the false discovery rate (FDR) by first finding the number of average genes which are deemed significant from each of the single B permutations using Δ. This value will be considered as the average number of false positives.

9. The FDR will then be computed as this average number of false positives divided by the number of genes called significant. For instance, if the average number of false positives is calculated to be 8.4 from the B permutations, and the number of genes deemed as significant is found to be 46, then FDR in this case will be 18%

2.2.4 LIMMA

The LIMMA (Linear Models for Microarray Data) package (Smyth and Smyth, 2004) is a popular tool written in R that is often used for gene expression analysis. It consists of algorithms ranging from preprocessing to finding differentially expressed genes. One of the useful methods in LIMMA for locating differentially expressed genes is the Empirical Bayes analysis (Efron et al., 2001) which is described below.

We assume that there are two classes of genes — viz., differentially expressed and non-differentially expressed genes. Suppose Z is the distribution of the (normalized) expression level of the gene, we let:

$$p_1 = \text{probability that a gene is affected} \tag{2.10}$$

$$p_0 = 1 - p_1 = \text{probability that a gene is unaffected} \tag{2.11}$$

and

$$f_1(Z) = \text{probability density of Z for affected genes} \tag{2.12}$$

$$f_0(Z) = \text{probability density of Z for unaffected genes} \tag{2.13}$$

The mixture density for the two populations is then given as:

$$f(Z) = p_0 f_0(Z) + p_1 f_1(Z) \tag{2.14}$$

We next estimate the value of f(Z) directly from the expression scores of the genes. Applying Bayes rule, we obtain:

$$p_0(Z) = \text{Prob (gene not affected | z)} \tag{2.15}$$

$$p_0(Z) = \text{Prob (gene not affected) * Prob (z | gene not affected)/Prob (z)} \tag{2.16}$$

$$p_0(Z) = p_0 f_0(z)/f(z) \tag{2.17}$$

Similarly, we obtain by Bayes rule:

$$p_1(Z) = 1 - p_0 f_0(Z)/f(Z) \tag{2.18}$$

Given that p1 must always be positive, it must satisfy the rules that

$$p_0 <= minf(Z)/f_0(Z) \tag{2.19}$$

and

$$p_1 >= 1 - minf(Z)/f_0(Z) \tag{2.20}$$

f_0 is in fact the null density and can be obtained by carrying out a random permutation of the phenotype labels. This allows us to estimate the value of ratio $f_0(Z)/f(Z)$ directly from the empirical Z and null density distributions. With this ratio we are able to calculate $p_1(Z)$, the probability that $gene_i$ is differentially expressed given the Z score.

2.2.5 Misc Techniques in Individual Gene Testing

There had also been other algorithms such as Conditional Independence (Akutsu et al., 1999), (D'Haeseleer, 2000), (Dhaeseleer et al., 2000), (Liang et al., 1998), (Yoo et al., 2002), Spearman Rank Correlation (D'Haeseleer, 2000), (Murphy, 2001), mutual information (D'haeseleer et al., 1998), (Butte and Kohane, 2000), Lasso (Ghosh and Chinnaiyan, 2004), silhouette (Rousseeuw, 1987), statistical P-values (Sohler et al., 2000), (Rogers and Girolami, 2005) and Pearson Correlation (D'haeseleer et al., 1998). They are basically similar in strengths and weaknesses and hence will not be elaborated further in this section.

In addition to these miscellaneous statistical techniques, a clinical microarray project (Wen-Son et al., 2009), which we were involved in, demonstrated that even basic analysis (we used the basic concept of counting highly expressed genes to determine their significance) are also capable of providing interesting results.

Because these algorithms give researchers lists of genes that are differentially expressed, often they will have to do further processing techniques to such gene list to obtain the biological conclusions that they seek. Thus these algorithms are very much lacking in providing descriptive biological inferences. The next two sections deal with algorithms attempting to provide the researcher with biological inferences from experimental data.

2.3 Gene Pathway Testing

This class of algorithms attempts to provide descriptive biological inferences from the microarray experiments. These algorithms (Auliac et al., 2008; Cooper and Herskovits, 1992; Dilip and Pankaj, 2005; Djebbari and Quackenbush, 2008; Friedman et al., 2000; Henegar et al., 2006; Ideker et al., 2002; Kauffman, 1993; Segal et al., 2003) usually work by recreating possible pathways from the experimental data and thereby finding out possible new associations between genes.

We provide these examples of such algorithms in this section:

1. K2 algorithm
2. Sparse Candidate
3. REVEAL Algorithm

2.3.1 K2 algorithm

The most basic algorithm available is that of the K2 algorithm (Cooper and Herskovits, 1992). This algorithm is essentially an algorithm to learn a DAG from data. The K2 algorithm assumes that the variables are first ordered according to the parentage. The pseudocode for the K2 algorithm is as follows:

```
function K2_ALGORITHM(dataSet D)
    For each node
        Pa(node) = null;
        old_accuracy = score(Pa(node), node, D);
        findMore = true
        while (findMore)
            Z = node where score(Pa(node) U Z, node, D) is MAX
            new_accuracy = score(Pa(node), node, D);
            if (new_accuracy > old_accuracy)
                old_accuracy = new_accuracy
                Pa(node) = Pa(node) U Z
            else
                findMore = false;
            end if
        end while
    end for
end function K2_ALGORITHM
```

In this algorithm, Pa(node) denotes the parent of the node. Score(Pa(node), node,D) denotes the score, of how well the network with node having Pa(node) as its parents would work out as applied to dataset D. Therefore, what this algorithm does is that for every node x, it searches through the other nodes looking for a possible candidate y as its parent. If the addition of node y causes the score to improve, the algorithm includes y into the parents of x. This continues until no new parents for node x can be found.

This node ordering is provided by an expert with extensive domain knowledge. Such information can be readily obtained as a DAG from Gene Ontology and Reactome. However as this technique builds upon knowledge that is pre-existing, the amount of new discoveries made will be limited.

2.3.2 Sparse Candidate

An improvement to the K2 algorithm is the sparse candidate algorithm. This algorithm omits the usage of a node ordering and caters to databases that are very sparse in nature, making it very suitable for biological databases. This algorithm was first proposed in (Friedman et al., 2000). Instead of carrying out exhaustive searches, heuristics are used to determine the topology of the network. One of the crucial points of this algorithm is that edge scores are calculated via a Bayesian framework. It was used in (Friedman et al., 2000) to analyze expression data. There are two stages in this algorithm, called the RESTRICT stage and the MAXIMIZE stage. The pseudo-code is shown below followed by its explanation.

```
function SPARSE_CANDIDATE(dataSet D, initial network B0)
    while non-convergence
        For each node
            select parents(node) based on D and Bi
        //Restrict Stage
            Find network Bn that maximizes the Score(Bn|D).
        //Maximize Stage
    end while
end function SPARSE_CANDIDATE
```

In the **restrict** stage, the algorithm will assign a score to variable pairs and forming a child-parent relationship. This score assigned can be a metric like correlation or mutual information.

In the **maximize** stage, certain heuristics are used to prune the variable orderings and pairs obtained earlier. The paper first restricted the in-degrees of each node followed by a simple greedy hill climbing search.

There were three criterion proposed for the convergence of the algorithm. The algorithm can converge either when the score does not increase, the candidates for the parents do not change, or if the algorithm goes into a non-terminating cycle, when a number of iterations with no improvement in the score is breached.

However the sparse candidate algorithm is very much limited to small datasets because the calculation of the score in the maximize stage will be too expensive. To simplify this calculation, the assumption of uniform data sparseness is usually made (which might not be the case).

2.3.3 REVEAL Algorithm

Boolean networks was first studied by (Kauffman, 1969) and (Kauffman, 1993) as a tool for studying dynamics of complex natural systems. One of its main algorithms is that of the REVEAL algorithm, which we will review here.

```
function REVEAL(networkNodes)
    tempNodes = networkNodes;
    for i = 1 to k
        for all node in tempNodes
            for all permutations of networkNodes pNodes, where
                |pNodes| == i
                if pNodes directly affect Node
                    node.output = pNodes;
                    tempNode.remove (node)
                end if
            end for
        end for
    end for
end function REVEAL
```

Where *networkNodes* is the full set of nodes in the network, *k* is the maximum of input edges in the graph for a node. The algorithm is based on concepts from information theory (Shannon and Weaver, 1963) to calculate a metric to determine if will pNode directly affect Node. However the algorithm becomes impractical when *k* becomes large (> 4).

2.4 Gene Class Testing

The final class of algorithms (Hanisch et al., 2002; Khatri and Draghici, 2005; Liu et al., 2009; Subramanian et al., 2005) involves analysing microarray data together with existing biological information to obtain biological inferences. The algorithms reviewed in detail here are:

1. OverRepresentation Analysis (ORA)
2. Functional Class Scoring (FCS)
3. Gene Set Enrichment Analysis (GSEA)
4. Gene Network Enrichment Analysis (GNEA)
5. Signaling Pathway Impact Analysis (SPIA)

2.4.1 OverRepresentation Analysis (ORA)

ORA (Khatri and Draghici, 2005) was one of the earlier and more established algorithms which made good usages of gene classes in finding out the decisions on genes. This technique allows us to locate exactly specific biological mechanisms which change within a biological pathway. There are three main steps for this method:

1. All genes are grouped according to their respective Gene Ontology (GO) classes. To obtain a good classification, the algorithm removes classes with less than 8 genes within the class. At the same time, it removes classes with more than 150 genes. The reason for doing so is because too small classes would imply a class that is too specific, and too large a class would be too general.

2. Based on the raw expression values, the ORA algorithm then segregates the genes into two groups, "selected" and "non-selected". The selected group refers to the genes which are seen to be overexpressed. We decide if a gene is placed in the expressed group simply via a fold or t-test as explained in the earlier sections.

3. Hence for each group, we actually have a number of genes that are present within the group. We need to test if the number of genes present within the group is statistically significant. This score implies the probability of observing at least a

particular number of genes in a class among the selected groups. This score is much easier to explain using the following statistical example.

Suppose that we are given a set of n genes of which k belong to a category C and a reference set of m genes of which l belong to C. Since l elements of the reference set belong to C, by proportion, we expect to find $k' = ln/m$ elements in the test set. If k is larger than k', C is said to be enriched, if k is smaller than k', C is said to be depleted. To estimate the statistical significance, P-values are computed. The P-values can be calculated in two ways: either through applying a hypergeometric test to compute a one tailed P-value or through permutation analysis. These two methods are described below.

a Hypergeometric test: The hypergeometric test is based on the sampling a fixed population. Assume that 20 balls out of 100 balls in a basket are white and we wish to calculate the probability of drawing 7 or more white balls of out 10 balls given the distribution of balls in the basket. Hence in this case, the 100 balls is the total m genes of reference set, 20 balls l genes which belong to C, 10 balls the sample set of n genes and finally the 7 balls would refer to the k genes in the sample set which belong to C. The hypergeometric probability can be calculated by the formula as in Equation 2.21.

$$p - value = 1 - \sum_{i=0}^{k} \frac{\binom{l}{i}\binom{m-l}{n-i}}{\binom{m}{n}} \tag{2.21}$$

b Permutation analysis: However, calculating Equation 2.21 might be intractable for large values of l, m or n. Hence the other alternative would to calculate the P-value via empirical means. For each permutation, we would pick up n genes from the gene list m, with r_i being the number of genes from n_i which is in C. This permutation procedure is repeated for $I = 10,000$ iterations. The P-value can be calculated by the formula in Equation 2.22.

$$p - value = \frac{Num\ permutations\ where\ r_i > k}{I} \tag{2.22}$$

Thus we will be able to calculate a score for each GO set and this score implies the probability of the class being overrepresented.

There are several limitations associated with this method. Mainly, it firstly predivides the genes into two groups, selected and non selected. The method of division is usually based on an arbitrary threshold. The choice of this threshold thus will have an effect on the final classes of genes analyzed as "overrepresented". Another important limitation is that after the genes are deemed as selected or unselected, they are considered as equal regardless of their raw expression values. However, it would make more sense to treat them differently depending on the strength of the individual raw values.

2.4.2 Functional Class Scoring (FCS)

FCS (Pavlidis et al., 2002) was created to address some of the limitations of ORA as it is able to obtain overrepresented groups within the GO ontology without having to predivide the genes into "selected" or "unselected". Moreover, the raw microarray values of the genes within the experiment is also taken into consideration. The details of the algorithm are as follows:

1. The p-values of the individual genes is simply the negative log of the raw microarray value, given by $-log(P_k)$.

2. All genes are next grouped according to their GO classes (similarly to the FCS algorithm).

3. The arithmetic mean – considered as a raw value for each of the individual GO class – is calculated as $\sum_{k=1}^{n} -log(P_k)$.

4. Permutation testing is lastly carried out to obtain a final p-value to each GO class. We illustrate the technique for permutation testing for each GO group with this example. For instance, the size of a particular class is k, and its raw score is r. The algorithm will next draw out random samples of size k from the entire list of genes, and calculate the raw score of this randomly generated group. This is repeated q times, and the resulting distribution is next stored. The p value for the GO group would then be the fraction of these random trials greater than the raw score r.

Hence at the end of the FCS algorithm, we get those gene classes which contain genes are overexpressed on average. Two years later, the group at Broad Institute introduced GSEA. Although GSEA addresses many of the limitations introduced by ORA, it is slightly different to FCS. FCS ranks the gene groups according to the expression values of the individual genes, while GSEA does it via the differential expression between the gene as compared to that of a fixed phenotype. How this is done is explained in the next portion.

2.4.3 Gene Set Enrichment Analysis (GSEA)

Of all algorithms reviewed so far, GSEA (Subramanian et al., 2005) seems to be the most complete and rigorous in its analysis. It is one of the first algorithms that is able to provide insights by focusing on particular gene sets, unifying genes statistically related with an unifying theme. For GSEA, we first have to rank the genes according to its differential correlation with a phenotype or profile. This ranking is based on the correlation $r(g_j) = r_j$ of their expression profiles with a phenotype of profile C. This list will be known as $L = g_1, ..., g_N$. For every gene set, we will also have predefined a set of genes, which we call S. The purpose of GSEA is to determine how this set of genes S is distributed across the main set L. There are basically three steps to GSEA.

1. It goes through the ranked lists of genes L. With each gene it calculates an enrichment score. We increase a running-sum statistic, Enrichment Score (ES)

when we encounter a gene in a set S and decrease it when we encounter genes not in S. The formula is as follows:

$$P_{hit}(S,i) = \sum_{g_j \in S j \leq i} \frac{r_j}{N_R}, where N_R = \sum_{g_j \in S} r_j \tag{2.23}$$

$$P_{miss}(S,i) = \sum_{g_j \notin S, j \leq i} \frac{1}{N - N_H} \tag{2.24}$$

The enrichment score will be the maximum deviation from zero of $P_{hit} - P_{miss}$.

2. The p-values are next calculated using a technique known as permutation testing. This technique estimates the significance via the set of scores of ES_{NULL}. This method corrects the often incorrect method of using statistical tables, which assumes the data follows the normal distribution. The P value is estimated via the following steps:

    (a) Randomly assign the original phenotype genes to the samples in the microarray experiment, thus getting a reordered gene list L. Recompute ES(S).

    (b) Repeat step 1 for 1,000 permutations π to obtain a distribution of enrichment scores, ES_{NULL}.

    (c) Estimate the P value for S, known as $ES(S,\pi)$ from the distribution ES_{NULL}.

3. The enrichment scores, $ES(S,\pi)$ and $ES(S)$ are next normalised by dividing by the mean of $ES(S,\pi)$. This yields the normalised scores $NES(S,\pi)$ and $NES(S)$.

4. The FDR is lastly computed. This will place a cut off to the genes which reject the null hypothesis. This cut off is determined by the estimated probability that a set with a given NES value contains a false positive.

The FDR is defined as the probability that a given gene identified as differentially expressed is a false positive. Here we describe the following procedure for calculating

FDR as introduced in (Benjamini and Yekutieli, 2001) and implemented in GSEA in (Subramanian et al., 2005). We first define a null hypothesis for each single gene g_i known as H_i. For each H_i we first obtain their t-values T_i followed by their corresponding p values p_i. Let $P_{(1)} \leq P_{(2)} \leq P_{(3)} \leq P_{(m)}$ be the ordered p-values with the hypothesis $H_{(i)}$ corresponding to $P_{(i)}$. For FDR, let k be the largest i for which $P_{(i)} \leq \frac{i}{m}q$. We would reject all $H_{(i)}$ where $i = 1, 2, ..., k$. For FWER, let k be the largest i for which $P_{(i)} \leq \frac{q}{m}$. q here refers to the threshold for the level of significance. We would reject all $H_{(i)}$ where $i = 1, 2, ..., k$. The value of q is often given the value of 0.05.

2.4.4 Gene Network Enrichment Analysis (GNEA)

The next technique we describe (which is very similar to GSEA) is known as Gene Network Enrichment Analysis (GNEA) (Liu et al., 2007). The GNEA consists of the following steps:

1. Aggregate a list of gene sets associated with the biological process of interest. These gene sets were taken from the Human Protein Reference Database, HPRD (Prasad et al., 2009) as well as Gene Ontology (Pavlidis et al., 2004b). (HPRD is a result of an international collaborative effort between the Institute of Bioinformatics in Bangalore, India and the Pandey lab at Johns Hopkins University in Baltimore, USA. HPRD contains manually curated scientific information pertaining to the biology of most human proteins.)

2. Importing a global protein-protein network from biological literature, we create a subnetwork within this global protein-protein network based on the individual gene perturbation for each patient. This global protein-protein network is obtained from HPRD.

3. For each gene set, we evaluate if this subnetwork is significantly expressed within the gene set. We repeat Steps (2) and (3) for every patient.

4. Order the gene sets by the number of subnetworks where they appear enriched.

5. Evaluate the p-value of each of these gene sets by this ranking. Gene sets with a significant p-value are taken as transcriptionally affected across the phenotypes of that disease.

GNEA is very similar to the GSEA (described in the preceding section). The difference being GNEA scores only segments of pathways instead of GSEA which scores entire pathways.

2.4.5 Network Expression Analysis (NEA)

The goal of NEA is to find a set of transcription regulators (TR) which are responsible for driving the differential expression of other genes. Here, they define a TR as significant if the downstream genes of its regulatory network exhibits a pattern of differential expression significantly deviating from the distribution expected by random chance (Sivachenko et al., 2005).

The actual distribution is measured in this manner: the absolute log-ratio value of the gene is replicated according to the number of TR-s (in-degrees) regulating this gene. This technique allows us to observe the absolute change in log-ratio as well as its adherence to the biological regulatory relationship. Hence if the relationship is one of activation (suppression) and the gene is activated (suppressed), this log ratio will be given a positive score. If conversely the relationship is one of activation (suppression) and the gene is suppressed (activated), the log ratio is given a negative score. This creates a signed edge distribution.

To approximate the null distribution expected from random chance, we allow the TR to randomly pick edges bearing effect signs. A signed significant test (for example t-test) can be applied to find the TRs that are significant.

2.4.6 Identifying regulatory modules

The method of (Segal et al., 2003) relies on inferring regulatory modules from gene expression data. This method takes in a gene expression data set and a large precompiled set of candidate regulatory genes. Next, the method partitions all the genes into modules and searches for a regulation program for each module. The regulation program of a module specifies the set of regulatory genes that control the module and a set of rules that determine the gene expression profile of the genes in the module in terms of the expression of the module's regulators.

A brief description of the method is as follows:

1. A set of candidate regulators are first compiled from the microarray experiment and clustered into three categories—viz., downregulated, no change and upregulated.

2. The aim is to assign a regulation program to as many of these candidate regulators as possible. By a regulation program, it means a set of rules determining the behaviour of candidate regulated genes (downregulated, no change and upregulated) based on the behavior of these candidate regulators. This set of rules is derived using a regression tree (analogous to a decision tree). Hence, the nodes of the regression tree are thresholds on relevant candidate regulator genes and the leaves are the behavior of the relevant candidate regulated genes. The candidate regulated genes are next clustered into candidate modules. Then an iterative expectation-maximization (EM) algorithm is applied to refine the modules and find regression trees that encode the corresponding regulation programs of these modules. Basically, in the M step, a regression tree is learned for each module. In the E step, genes whose behaviours are best explained by the regulation program of a module are re-assigned to the module. Care is taken such that a gene which is part of the regulatory input to a module is not assigned to that module as a regulated gene of the module.

2.4.7 Signaling Pathway Impact Analysis (SPIA)

The final technique we describe is known as Signaling Pathway Impact Analysis (SPIA) (Tarca et al., 2008). SPIA attempts to provide better biological results by calculating significance based on the over-representation of genes within a pathway as well as the amount of perturbation measured in each pathway. SPIA thus comprises of two separate probabilities: P_{NDE} and P_{PERT}. P_{NDE} captures probability that a given pathway is significant based on the over-representation analysis of the number of differentially expressed genes within the pathway. P_{PERT} captures the probability that the pathway will be significant based on the connections that exhibit a differential behaviour within the pathway. P_{NDE} can be calculated using one of the previous mentioned techniques such as ORA and FCS.

The second probability P_{PERT} is calculated based on the amount of perturbation experienced within each pathway. We define the perturbation factor for each gene g_i as:

$$PF(g_i) = \delta E(g_i) + \sum_{j=1}^{n} \frac{PFg_j}{N_{ds}(g_j)} \tag{2.25}$$

where PFg_j refers to the perturbation factors of the upstream genes, and it is normalised by N_{ds}, the number of downstream genes. We next calculate the net accumulated perturbation by subtracting $\delta E(g_i)$ from $PF(g_i)$. This subtraction is required so that genes that are solitary will not affect the probability P_{PERT} (since they have already been considered in P_{NDE}). This accumulated perturbation is hence given by:

$$Acc(g_i) = PF(g_i) - \delta E(g_i) \tag{2.26}$$

The total net accumulated perturbation for a pathway is simply the sum of the accumulated perturbation of the genes within a pathway A, $t(A) = \sum_i Acc(g_i)$ and the probability P_{PERT} is given by:

$$P_{PERT} = P(T_A \geq t_A | H_0) \tag{2.27}$$

T_A is a null distribution of t_A values, obtained empirically through a bootstrapping approach of randomising the relationship of genes within a pathway. H_0 stands for the null hypothesis, that the genes that appear as differentially expressed on a given pathway are completely random. Finally, we calculate the probability of a pathway with the formula:

$$P_G(i) = c_i - c_i * In(c_i) \tag{2.28}$$

where $c_i = P_{NDE}(i) * P_{PERT}(i)$. From the list of $P_G(i)$ values calculated, the FDR algorithm as proposed in (Benjamini and Yekutieli, 2001) was used to control the false discovery rate at 5%.

2.5 Discussions

This literature survey of the different classes of analysis techniques suggests that contemporary algorithms are unable to provide the researcher with sufficient biological descriptions. For instance, techniques such as the fold change (Choe et al., 2005), t-test (Cui et al., 2005) and SAM (Tusher et al., 2001) generate gene lists containing large number of genes, requiring the researcher to carry out additional processing to draw biological information from them. Furthermore since the number of samples available for analysis is usually very small relative to the number of genes to be considered, such techniques always suffer from the having too much false positives (due to multiple hypothesis testing). In addition, they require specific thresholds to be set in order to determine genes which are significantly expressed.

Techniques such as GSEA (Subramanian et al., 2005) and NEA (Sivachenko et al., 2007) have attempted to solve these issues, for instance, incorporating prior biological information by finding differentially expressed functional gene sets (hence providing more biological descriptive information), introducing false discovery rate (FDR) (Benjamini and Yekutieli, 2001) (hence alleviating the issue of multiple hypothesis testing).

However, these methods share some limitations. For instance, most of these techniques (Subramanian et al., 2005) ignores the intricate protein connections and protein topology within pathways. Although techniques such as (Liu et al., 2007; Sivachenko et al., 2007) take into account protein interactions, they are considered in a general sense, not sufficiently detailed for biological interpretation. Finally, most of these algorithms locate entire pathways or gene sets which are differentially expressed, ignoring the situation when only a portion of a pathway is significantly expressed (which might be prevalent when the gene set is extremely large).

One major group of algorithms which centres on time series clustering/analysis of microarray analysis have not been reviewed here as they are not the focus of this thesis. However we recognise that such algorithms can be very interesting when our concept is extended to time series data. The interested reader can refer to various key algorithms ranging from traditional algorithms like hierarchical clustering (Eisen et al., 1998), K-means algorithm (Tavazoie et al., 1999) to specialised clustering methods involving time series microarray data using expression profiles, slopes (Wen et al., 1998), linear splines (de Hoon et al., 2002) and just recently, a gene rank based approach (Yi et al., 2009).

CHAPTER 3

Background Study on Biological Databases

Chapter Synopsis

Summary

One of the critical assumptions underlying microarray techniques which integrates biological information (biological pathways) as a priori data into their computational analysis is the availability of a collection of well-curated biological repositories (for example, Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa and Goto, 2000), Ingenuity (Ingenuity, 1998)). Specifically, we focus on three criteria: compatibility of database (if the database is capable of merging seamlessly into the analysis technique), comprehensivity of database (if the database contains sufficient information) and consistency of database (if the database provides information consistent with other biological databases). We review several representative biological databases and analyse their suitability to be used as a priori biological information based on the three criteria listed above.

Conclusions

Based on our criterion, our survey findings conclude that contemporary biological databases are very diverse. In terms of compatibility, most repositories still relies heavily on human intervention when manipulating biological pathways (which is impractical *when analysing thousands of genes across hundreds of pathways for each microarray experiment). With such little support for programming interfaces, integration of their biological pathways into computational algorithms becomes difficult. In addition, a popular biological pathway (the apoptosis pathway) was chosen from three representative biological databases to investigate the level of comprehensiveness and consistency of each database. Alarmingly, results show an extremely low level of similarity in the genes quoted by the three databases (*32% *to* 46%*) for the apoptosis pathway. This shows that none of the three databases are comprehensive (because other databases have information which they do not have) and neither are they consistent (because the level of similarity between them is consistently low). These reasons lead us to create our own unified database cache and its API for easy integration into computational algorithms.*

3.1 Introduction

As described, the current trend of microarray analysis is for prior biological information to be integrated/used as part of the input to the analysis method. As such, it is important that we look at the typical sources of biological information used/needed by these modern analysis methods. In our study and search for databases (which are suitable for integration into microarray analysis), we will be concentrating on the following criteria:

+ Are they sufficiently comprehensive individually?

  Because the output analysis from algorithms depend significantly on the pre-existing biological data, care must be taken to ensure that the data within the biological repository is comprehensive. Otherwise the results may be highly inconsistent as they depend largely on the biological database being used. Hence we check if there is any single database sufficiently comprehensive to represent most of the pathway data that is available.

+ Are the databases easily accessible to researchers who wish to use the data for their analysis?

In order to integrate data from these databases into algorithms, databases must have proper software interfaces for computer scientists to connect to. Without such an interface, integration will be highly impossible.

+ Are various biological pathway data sources consistent with each other?

  In the event where we require data from different biological sources to communicate with one another, we require that they follow consistent molecular representations, pathway naming conventions and exchange formats. This allows for the inter-communication of biological databases.

3.2 Review of Contemporary Databases

In this section, we review if current biological databases are suitable for use in computational algorithms which incorporate biological information. This review of databases will be conducted based on the criteria listed in the previous section.

3.2.1 KEGG, Reactome

Data

KEGG (Kanehisa and Goto, 2000) and Reactome (Joshi-Tope et al., 2003) are well established pathway databases. Their pathway data is first obtained from literature and accurately curated and approved by experts. The focus of these two databases is to provide pathway data as easily and seamlessly to their users as possible, they also boast the most flexible methods of data access. For instance, both databases allow their users to search for individual pathways and visualise them in pictorial formats. For KEGG, they have an API (Kawashima et al., 2003) which allows computational scientists to connect to their databases and extract pathway information from them. Hence to access all the data from KEGG, one just executes the following API commands:

1. Execute an API call to obtain all the available pathways (for the homo sapiens species) in KEGG.

2. For each individual pathway, execute another API to obtain all the genes and gene relationships for that particular pathway from KEGG.

In contrast, Reactome provides a sql data dump and also access through a SOAP api. These data access methods allow information to be easily downloaded from their website, allowing easier integration to computational algorithms.

Data Format

Pathways from KEGG are represented either in SOAP (returned when using API calls) or BioPax (downloaded individually) (Kotecha et al., 2008). However we realized that the API is not well updated and is no longer supported by the latest versions of python. In addition the data stored in BioPax does not seem to be well maintained. Information from KEGG remains stored in a BioPax level 1 format when the latest BioPax stands at level 3. For Reactome, because they allow the download of the entire database in sql, it is the most flexible and easily integrated into computational algorithms.

3.2.2 Ingenuity

Ingenuity (Ingenuity, 1998) is a commercial company which provides biological pathway information to their subscribers. The company obtains such information using manual curation by experts from published literature. Ingenuty provides data only in image pictorial format. This is probably to control the rapid duplication of its proprietary information to the different sources. Hence to obtain information from Ingenuity, one must painfully extract the pathways details by extracting individual gene and gene relationship manually.

3.2.3 NCBI

One of the main sources of biological information available online is NCBI (NCBI, 2009). Founded more than twenty years ago, it now houses one of the largest set of free online resources for biological information. As NCBI caters more for the users who wish to search for online information, they focus mainly on UI design and visualisation tools which allow one to easily search for the genes / pathways required and to drill down to various details of genes or relationships within pathways. Yet these databases are inconvenient when it comes to programmatically obtaining information from them. To illustrate this point, NCBI allows one to easily search for genes / pathways of various species for numerous parameters within many different datasets. Executing such a search provides one with details like which pathways the gene appears in, the list of literature the gene is mentioned and even related pathways to the gene. However such information is not suitable for use and integration within computational algorithms. Indeed, NCBI has in fact even implemented steps to prevent such misuses of its data. Specifically, it limits each IP address to 2,000 Internet connections on any single day, hence preventing such scenarios from occurring.

3.2.4 Cytoscape

Cytoscape (Shannon et al., 2003) allows for the complex visualisation and editing of biological pathways. The purpose of these tools is to help display the experimental data on detailed diagrams of the relationships between genes/proteins in known pathways. Thus, Cytoscape serves more as a visualisation tool and the biological information fed into the tool depends on data generated by the user. Being an established software tool, it has the capability of reading multiple types of input files including Graph Markup Language, SBML and BioPAX.

3.2.5 BioCyc

BioCyc is a collection of databases where each database in BioCyc describes the genome and metabolic pathways of a single organism. Other than the provision of data, one of its unique propositions is the many tools within Biocyc that allow the user to navigate, visualise and analyse the different databases. Relevant to this thesis, BioCyc allows for the download of its data in various formats to be analysed using its tools. In addition to comprehensive tools and visualisers (like NCBI), BioCyc boasts its own Pathway Tools API which allows a user to query the database in Lisp, Java and Perl.

3.2.6 SBML

We review one of the commonly used data formats in this subsection, the Systems Biology Markup Language (SBML). This initiative was started in 2000 to work on developing better software infrastructure for computational modelling in systems biology. It has garnered popular usage and is supported by at least 200 biological packages. As SBML was designed for a different use in mind (for simulation of systems biology reactions) most of the features within SBML might be redundant in our experiments. However, when the current scope is expanded to include systems biology simulations, such a markup language will prove to be very relevant.

3.2.7 Wikipathways

Data

Wikipathways (Wikipathways, 2004) is a community effort which invites professionals to aid in providing pathway data to its repository voluntarily. In addition to curating the data from academic professionals, Wikipathway also obtains data from the other repositories and incorporates it into its database. This has allowed it to store a substantial number of pathways.

Data Format

Pathway information from Wikipathways is usually in the pictorial format. However even though Wikipathways does not provide any API or software interface, Wikipathways supports a markup format known as the GPML format. This format allows a person to plot out the exact details of the pathways because it contains the spatial location of the genes and relationships on the pictorial format. It also allows a direct download of all its pathways in the GPML format (Dahlquist et al., 2002). Understanding the pathway GPML format from Wikipathways is possible, although non-trivial and difficult. To do so, we are required to parse through the format and carry out numerous image processing techniques to properly extract the genes and their relationships.

3.3 Review Findings

3.3.1 Incomprehensive Data

Although many commonly used pathway databases (eg NCBI (NCBI, 2009), GO (Pavlidis et al., 2004b), Reactome (Joshi-Tope et al., 2003), HumanCyc (Romero et al., 2005), HPD (Chowbina et al., 2009), Panther Pathways (Thomas et al., 2003), etc) were reviewed, we have selected three representative data sources (KEGG, Ingenuity and Wikipathways) for our analysis on data comprehensiveness among different databases. These sources are chosen because they are representatives of three very different kinds of curation effort. For instance, Wikipathways is maintained by a community of professional users via the free and open wiki platform. KEGG database is curated independently by a single lab from published literature. Ingenuity is a commercial product.

We demonstrate the lack of comprehensiveness (incomprehensiveness) of current databases with a manual comparison on the agreement of the apoptosis pathway across these three databases. By the term incomprehensiveness, we mean that each single biological database is not a comprehensive representation of biological knowledge that are acknowledged by experts to be accurate.

We define the following metrics to illustrate the diversity across databases. The first metric, the “Gene Agreement Count” of a pathway, counts the number of genes that are common to that pathway in all the databases. The second metric, the “Gene Pair Agreement Count” of a pathway counts the number of “interacting gene pairs” that are common to that pathway in all the databases. An interacting gene pair is a pair of genes (or their products) that are directly interacting in a pathway. When calculating the “Gene Agreement Percentage” of a pathway, we first find the total number of genes within that pathway for each individual database. We next select the gene count from the database that has the least number of genes for that pathway. Finally we divide the Gene Agreement Count by this minimum gene count to obtain the Gene Agreement Percentage. The same technique is employed to calculate the Gene Pair Agreement Percentage.

Results indicate an overlap range of 11%-16% for gene pairs and an overlap percentage of 32%-46% for genes. This is an extremely low level of agreement given a pathway as pervasive as the apoptosis pathway. This highlights the fact that using any single database alone is not sufficiently comprehensive. Full results are seen below in Table 3.1.

Table 3.1: Table showing data overlap for Apoptosis Pathway. This table shows the manual calculation of the gene/gene pair differences between the different repositories for the apoptosis pathway.

Apoptosis Pathway			
	KEGG x Ingenuity	KEGG x Wiki	Ingenuity x Wiki
Gene Pair Count:	151 vs 3374	151 vs 133	3374 vs 133
Gene Count:	89 vs 169	89 vs 82	169 vs 82
Gene Overlap:	33	38	26
Gene % Overlap:	37%	46%	32%
Gene Pair Overlap:	21	21	15
Gene Pair % Overlap:	14%	16%	11%

It should be highlighted that Ingenuity has a disproportionate high number of gene pairs/genes. This is because in Ingenuity, entire pathways are being annotated as nodes, which are expanded by us into its many genes/gene pairs. However, our test remains fair as the overlap percentage count between databases is obtained by dividing the number of genes/ gene pairs overlap by the minimum number of genes or gene pairs from either database. Hence the large number of gene / gene pairs from Ingenuity does not affect the fairness of the test.

3.3.2 Incompatible Methods for Data Acquisition / Data Formats

3.3.2.1 Incompatible methods for data acquisition

All databases release their pathway information via some non-standard graphical format. Such a graphical representation is useful for visual manual analysis. However, it is inconvenient for large-scale computational analysis.

The primary preferred method of data analysis for pathway information for clinicians is still the pictorial form. However, having pathway information in pictorial formats would mean that the clinician first has to download the data in the pictorial format from the repository. Next the clinician will have to refer to the visual representation of the pathway individually while making his analysis manually.

While such visual representation is useful on a small scale analysis, this visual representation is virtually useless if one is required to incorporate such information within large scale computational analysis.

Different databases use different methods of data access. Some databases only allow data to be downloaded via web access. Others provide flexible access to their databases through their API.

This means that a lot of human effort and intervention is required to download the required information for databases. For databases with no API (take for example Ingenuity), this creates tedious challenges for software to obtain information from such databases.

For databases whose API is public, there is no guarantee that all such API would use the same programming languages. This causes developers to incorporate clumsy wrappers within their applications to adhere to the API of the databases.

Fortunately most databases offer a secondary form of data acquisition which allows one to download such pathway data in a format more friendly to algorithms than humans.

Yet this secondary form of data acquisition differs largely from database to database. For instance, if one wishes to draw data from KEGG, he/she would use the KEGG API to download the required data. On the other hand, if Wikipathways is used, a spider must be written to scan and download data from its website. This means that if a computational scientist wishes to incorporate pathway information within his algorithm, he will have to customise an entirely new method for pathway extraction for each of individual data repository!

3.3.2.2 Incompatible Data Formats

Some repositories do release their data in formats such as their proprietary markup languages or API data structures. These are more convenient for large-scale analysis. However such formats are always unique to their originating database.

This lack of a consistent data format means that different databases use different formats to represent their data. Hence dedicated codes have to be written to parse, understand and integrate data from each individual database.

For instance, KEGG allows download of the information via their proprietary SOAP data structures while Wikipathways allow for the download of data via their format known as the GPML format. The two formats are very different from each other and would require the computer scientist to write dedicated parsers for each database.

We would like to point out that there have been efforts to make data exchange formats consistent with one another. An example of such an effort is BioPax (Kotecha et al., 2008). However, the BioPax format was first designed in 2002. A cursory look would show that it is cumbersome for computational manipulation. An updated API which allows fast and simple manipulation of pathways is wanting. In addition, BioPax updates its data in a very manual fashion. For instance, the links to pathway information in BioCyc is as old as 2005 and links to KEGG data in BioPax are dead. Thus such an exchange format might no longer be technologically suitable.

3.3.3 Inconsistent Data / Molecular Representations / Pathway Naming

3.3.3.1 Inconsistent Data

We define inconsistent data to mean data which differs as compared from one data repository to another. These inconsistencies occur in gene-gene relationships where different repositories provide different or even contradictory relationship information between genes for the same pathway ie, Pathway I of Database 1 states that Gene A activates Gene B but Pathway I of Database 2 states otherwise (Gene A inhibits Gene B, Gene B activates Gene A or Gene B inhibits gene A). An example of such an instance occurs within the KEGG's Cell Cycle Pathway, gene RB1 activates gene TFDP1. Ingenuity's Cell Cycle Pathway however states that gene RB1 inhibits gene TFDP1.

Because of the inconsistency and incomprehensiveness of data (demonstrated above), cross database queries are the intuitive solution to harness the required information across these databases. Doing so is crucial else the microarray analysis results would be too dependent on the databases. Meaning the researcher running his experimental data on Database 1 will end up with a totally different set of results if he runs it on Database 2 due to the fact that the data within the pathways are already inherently different. The latter portion of this report will demonstrate quantitatively how different the figures are across the different databases. However incompatibilities between different databases makes cross database accesses extremely challenging to execute. We investigate these incompatibilities across different databases and present them here.

3.3.3.2 Inconsistent Molecular Representations

Different repositories assign different naming conventions to their pathway nodes. These nodes can be described as proteins, genes or symbols depicting protein families. For example, KEGG describes most of their elements as genes, Ingenuity describes them as proteins, while Wikipathways uses a combination of both.

We have gene names / IDs described inconsistently across different data repositories. The problem here is heavily compounded because different databases have a different way of referring to genes. One of such instance where this inconsistency was particularly compounded was when the Ingenuity database used a single protein symbol "PAKC"to represent 50 different genes on a single relationship within a pathway. This will greatly increase the difficulty of both the computer scientist as well as the researcher in grabbing information which will result in incomplete pathways and incomplete analysis.

Hence it is possible to miss crucial genetic relationships because of such inconsistent representation. To obtain all relationships represented within pathways, algorithms are required to convert all nodes to a common representation.

3.3.3.3 Inconsistent Referrals to Pathway Names

Common biological pathways in different databases are often given names with no indication of how pathways are related to one another. For instance, KEGG may refer to a pathway as “Wnt signaling and pluripotency” and the Wikipathways might refer to it simply as “Wnt signaling”. Other than the fact that both pathways have the common terms “Wnt signaling”, there is no way of knowing if the “Wnt signaling” pathway is a subset of the “Wnt signaling and pluripotency” pathway other than through human intervention.

This makes it difficult to determine pathways that refer to similar biological processes (albeit sporting different pathway names). It is difficult to match and compare similar pathways across different repositories.

3.4 Motivation for Pathway Aggregation

The prior section discussed the incomprehensiveness, incompatability and inconsistency of current databases. This illustrates that current databases are not ready for allowing seamless integration of their data into biological algorithms, and certainly not equipped to cater to the needs of newer generation of computational algorithms which relies heavily on prior biological information.

Thus if we require that our algorithm works seamlessly with currently available databases, we have the following options:

1. To utilize only the biological information from a single database which supports a software interface

2. To create a separate software wrapper for each individual database that we use, and integrate this wrapper into the algorithm

3. To create an integrated database cache and API which connects to the other databases, allowing the researcher and algorithm to access to up-to-date from the diverse data sources at all times.

The first option is not advisable because it only provides incorporates data from a single data source. Hence the analysis obtained might not be totally accurate and may well be skewed.

The second option provides us with access to multiple databases within the algorithm. However because it depends heavily on wrapper classes within the algorithm itself, the entire algorithm will probably go offline when one day the parent databases decide to change part of their protocols or formats.

Hence we are left with the third and best option of providing an integrated database with a common API. This common API will always maintain a version of the data stored in the parent databases. At the same time it allows one to obtain the information from the other databases from a single point of access. Because all data stored will be stored using a standard nomenclature, accessed with a standard protocol, accessed in a standard format, it solves the problems of incompatabilities and inconsistencies. In addition, several databases will be supported, to ensure that the information stored is comprehensive.

This common API works in this manner: A local database serves as a cache, storing data from the other repositories. Requests for information from the different repositories are directed to this cache to obtain the required information. This is achieved also via software wrapper classes. To ensure that our interface is always kept up to date, automatic incremental updates are run periodically to extract the latest information from the different repositories. This process creates a unified interface for the different databases, as well as a unified database where graphs of the same pathway are merged.

3.5 Discussions

In spite of the advancements in microarray analysis explained in the previous chapter, it is still challenging to draw biological conclusions from today's microarray experiments. The main source of the difficulty is that the number of samples available for analysis is usually very small relative to the number of genes to be considered. It is often the case that many genes are statistically significant according to the wide variety of computational and statistical analysis algorithms. Yet there is little concurrence between the genes selected by different algorithms. Furthermore, the genes selected by these algorithms do not always provide an insight that is biologically consistent or biologically interpretable.

In order to obtain results that are more biologically meaningful, it is important to incorporate information from biological repositories into the analysis of microarray data (Soh et al., 2007). Indeed, most of the new generation of algorithms (under the category of "Gene Class Testing" in the previous chapter) incorporate information from biological pathways into microarray data analysis (Goeman et al., 2004; Khatri and Draghici, 2005; Subramanian et al., 2005).

Examples of the new generation of microarray data analysis algorithms that incorporate biological pathway information into the analysis process include ORA (Over Representation Analysis) (Goeman et al., 2004; Pavlidis et al., 2004a), FCS (Functional Class Scoring) (Goeman et al., 2004; Pavlidis et al., 2004a), GSEA (Gene Set Enrichment Analysis) (Subramanian et al., 2005), ErmineJ (Lee et al., 2005) and Pathway Express (PathwayExpress, 2009).

Examples of databases which these algorithms reference are: NCBI (NCBI, 2009), KEGG (Kanehisa and Goto, 2000; Kanehisa et al., 2006; Ogata et al., 1999), Ingenuity (Ingenuity, 1998), GO (Gene Ontology) (Pavlidis et al., 2004b) and Wikipathway (Wikipathways, 2004). In terms of source authority, both KEGG and Ingenuity derive their data from published work while Wikipathways first derive their's from several established databases (eg KEGG, Netpath) and are subsequently curated by the research community.

However these biological databases are very diverse, making it extremely laborious to carry out even simple queries across databases. To make matters worse, inconsistencies and incompatibilities between different repositories render the individual databases less effective for collaborative purposes.

This inconsistency is worsened because the boundaries of signaling pathways are not that clearly defined scientifically. For example, the pathway "MAPK Cascade" probably has no clear consistent definitions in the literature hence making the question of exactly which genes to include quite subjective (Green ML, 2006).

ORA (Khatri and Draghici, 2005), FCS (Pavlidis et al., 2002) and GSEA (Subramanian et al., 2005) are all examples of algorithms that incorporate information from biological databases. Both ORA and FCS use the GO database to select relevant genes according to their GO classes. GSEA uses their proprietary database (curated from various sources) for gene selection.

The importance of the accuracy and comprehensiveness of the biological pathway information used should be clear from the review above of modern microarray data analysis algorithms. For instance, clinicians may potentially end up with different results and conclusions depending on the database they group their genes by! Hence it is crucial that we did a review of some of the popular biological databases to determine their suitability to algorithms.

This shows that current databases are not suitable for seamless integration into microarray analysis because of incompatible, inconsistency and incomprehensive issues. Hence it was determined that the best option was to create an aggregated database cache with its own API, henceforth allowing different algorithms to utilize biological information from different databases from a location. Doing so solves both the problems of incompatabilities and inconsistencies. Because a few databases will be supported, we will ensure that the information stored within the databases be as comprehensive as possible. The implementation details of this database cache will be outlined in the next chapter.

Chapter 4

Pathway API — Methods

Chapter Synopsis

Summary

The previous chapter (Chapter 3) argued for the need to create our own biological repository (for integration within computational analysis) because of the lack of consistency and comprehensivity and compatibility of current databases. This chapter explains the technical challenges and techniques we employed in designing and creating our own biological repository. We focus on three main aspects of the design: (1) How we formalise key biological features within pathways to computational database entities. (2) How we extract, store and update these key features from publicly available pathway databases (in particular, we focus on three representative databases, KEGG, Ingenuity and Wikipathways) (Soh et al., 2009). (3) Finally we describe and provide a short specification of the application programmable interface (API) to the database such that others will be granted easy access protocols for integration into their computational algorithms.

Conclusions

The techniques presented in this chapter aim to create our unified database which is comprehensive, consistent with other databases and compatible for integration into computational microarray algorithms. We attempt to achieve data comprehensiveness *by integrating biological data from three representative biological repositories. Data consistency is achieved through the standardisation of nomenclature and data formats. We ensure that data from this unified biological resource is both compatible and easily accessible through simple web based http API calls.*

4.1 Introduction

The previous chapter showed how current databases are ineptly equipped to be easily integrated into computational algorithms which require biological information. Faced with this situation, the solution proposed was to create our own database. This chapter shows the methods adopted in creating such a database cache supported with an API.

To create such an API, we first have to define the type of information which is crucial for use for the database, followed by obtaining such biological data from respective data sources. The key components behind such an API are:

+ Pathway Formalisation: Key features within pathways

+ Database Cache: How we store, extract and update the data

+ API Implementation: Short specification of the API

4.2 Pathway Formalisation

Pathway databases supply many informative features that are useful for the purposes that these databases were originally intended for. However, for use in gene expression analysis algorithms such as ORA, FCS and GSEA it is sufficient to capture only two key features in these pathway databases.

One feature defines all the genes within the pathway while the other defines gene-gene relationship within the pathway. Here we only consider two relationships between genes: activation and inhibition. (Gene relationships in metabolic pathways are formalised in the same manner based on how they catalyse adjacent steps within the pathway. For metabolic pathways, relationships between adjacent proteins are indicated as neutral, meaning neither activating or inhibiting. This approach for metabolic pathways is similar to the approach adopted by KEGG.) This formalisation helps to organise and streamline information within pathways.

For illustration, we redraw a KEGG pathway in Figure 4.1. The original pathway is in Figure 4.2. The component depicting genes within a pathway refers to the individual genes MDM2, TP53, etc. The other component depicting gene-gene relationships refer to relationships (eg MDM2 inhibits P53, ATM activates CHK1) in the pathway diagram.

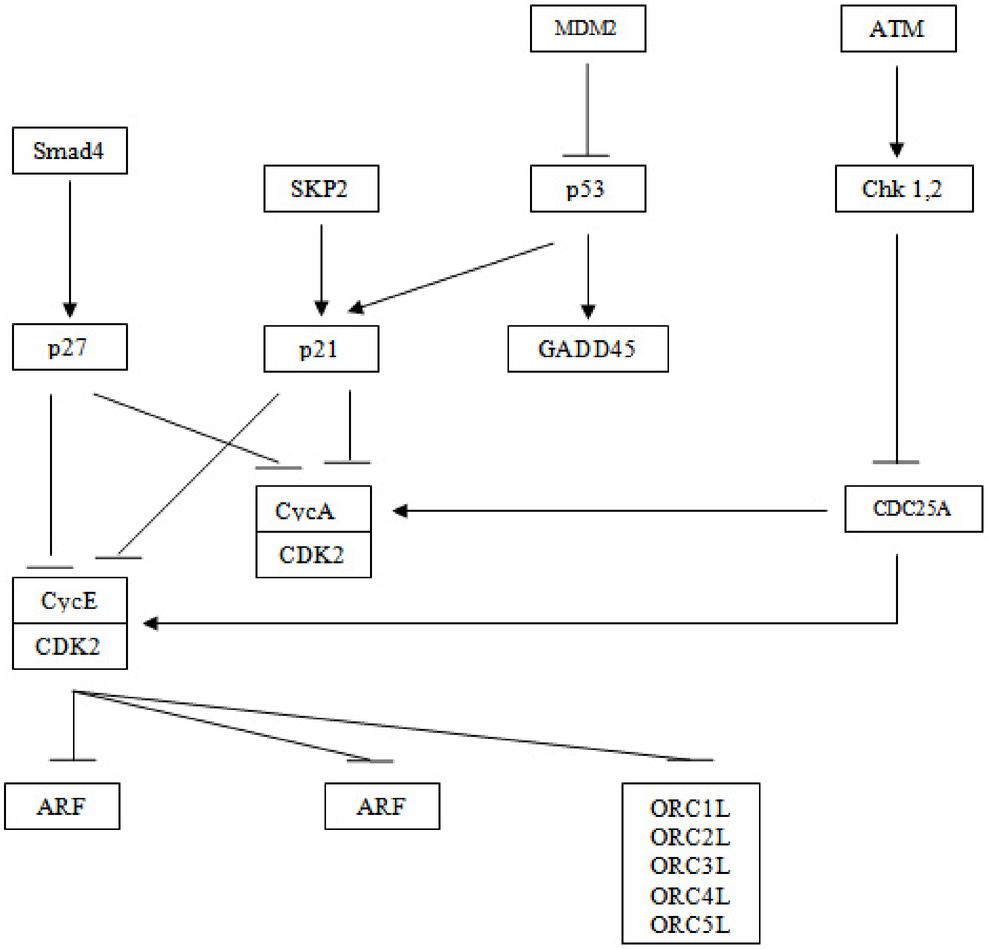


Figure 4.1: A selected portion extracted from the Cell Cycle KEGG Pathway (from Figure 4.2) and redrawn here.

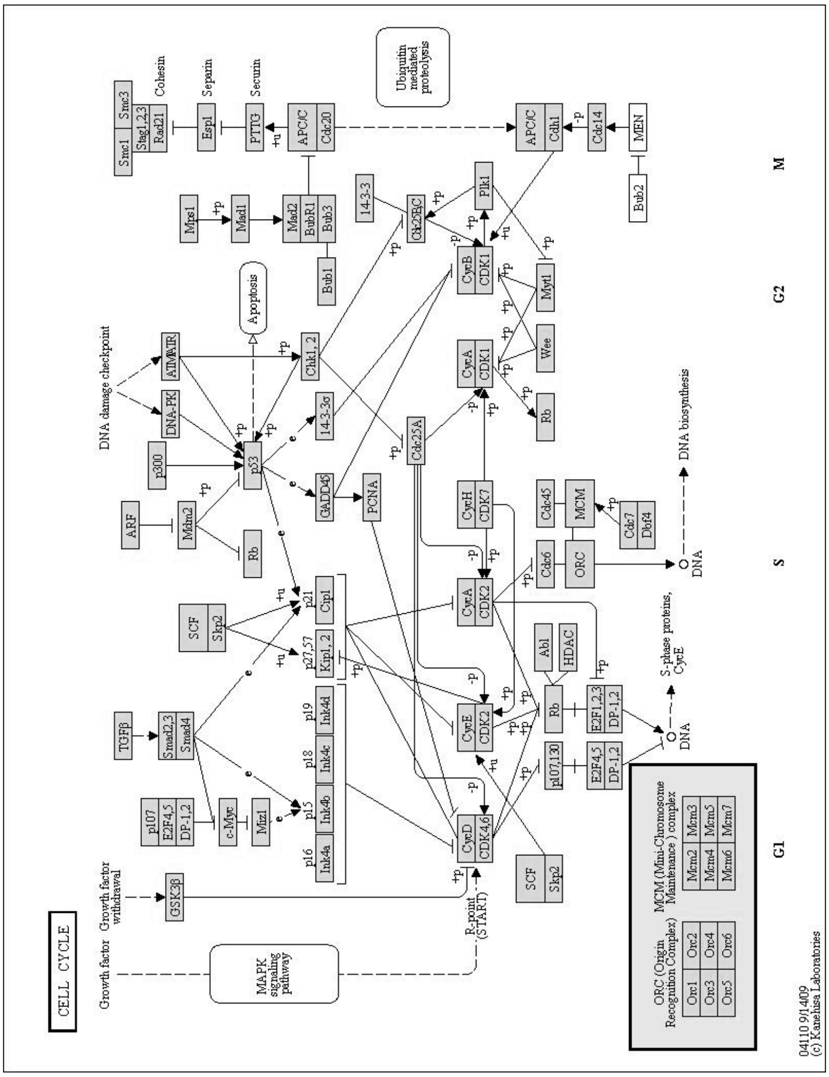


Figure 4.2: The original Cell Cycle KEGG Pathway (Kanehisa and Goto, 2000), extracted from the KEGG database and reproduced here.

4.2.1 Representation of Genes

As mentioned earlier, one of the inconsistencies across different databases is the inconsistent usage of proteins, genes or protein lists within pathway data. To address this issue, all gene or protein representations are converted to their corresponding NCBI Gene ID. The NCBI Gene ID is obtained by issuing and parsing the results of the query:

http://www.ncbi.nlm.nih.gov/entrez/

query.fcgi?db=gene&cmd=search&term=Y+homo+sapiens

Where the symbol Y refers to the gene name. Executing this query iteratively across all the genes/proteins within the pathway provides us with the Gene IDs within the pathway. This common terminology reconciles gene naming inconsistencies across the different repositories.

4.2.2 Representation of Genes-Gene Relationships

There are only two types of relationships present between genes: inhibition and activation. These two relationships are illustrated in Figure 4.3 where we see ATM activating CHK1, CHK2 and MDM2 inhibiting p53. Figure 4.4 shows the explicit relationship between MDM2 (as the inhibitor) and p53 (as the inhibitee).

By constructing such inhibitor-inhibitee/activator-activatee relationships, investigators explicitly know the exact relationship of genes within pathways. This allows them to analyse the adherence of these relationships in their experimental data.

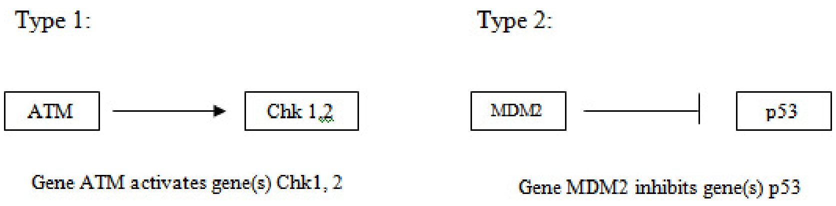

Type 1:
ATM
Chk 1,2
Gene ATM activates gene(s) Chk1, 2
Type 2:
MDM2
p53
Gene MDM2 inhibits gene(s) p53

Figure 4.3: Images depicting the two types of gene relationships considered. Left: Type 1 relationship where gene ATM activates both genes Chk 1 and Chk2. Right: Type 2 relationship where gene MDM2 inhibits gene p53. (Image reproduced from (Soh et al., 2007))

Inhibitor	Inhibitee
MDM2	p53

Figure 4.4: Figure showing distinctly the relationship between genes p53 and MDM2. MDM2 is explicitly referred to as the inhibitor and p53 referred to as the inhibitee.

4.3 Database Cache

In the following sections, we will be discussing on the storage, extraction and updating the data for our unified database.

4.3.1 Data Storage

We maintain a database cache to store information from the other pathway repositories. To ensure fast response to users, all queries submitted are directed to this database cache. Our database cache is kept up to date with a set of automated scripts written to do periodic incremental updates from the other databases.

The 3NF database schema used to store the captured data is shown in the Figure 4.5. Note that the fields in bold are the names of the table. The underlined fields are the primary keys of the table.

4.3.2 Data Extraction

4.3.2.1 Wikipathways

Data from Wikipathways are publicly available via their proprietary file format known as the GPML format (Dahlquist et al., 2002). Hence we first obtain the pathway IDs of all the pathways present within the Wikipathways database. The next step involves iterating through these ids to obtain the GPML file associated to each pathway ID. The final step parses the GPML format to obtain the pathway genes and associations.

All pathways within Wikipathways are obtained by issuing and parsing the query:

http://www.wikipathways.org/index.php/Special:BrowsePathwaysPage

A simple parsing of this page gives one the list of all the pathway names present within the Wikipathway database. The following webquery is issued to obtain the pathway information in a text format. Here, Wikipathways term it as a GPML file. The corresponding GPML file for each pathway is obtained with this query:

http://www.wikipathways.org//wpi/wpi.php?action=downloadFile
&type=gpml&pwTitle=Pathway:Homo%20sapiens:X

Where X refers to the name of the pathway.

The GPML format is designed towards the visual display of pathway information. Hence it contains detailed coordinate information about the spatial location of genes and arrows/t-bars (which depict activating/inhibiting relationships). Yet how these genes are related is not described in the GPML specifications. A parser is therefore needed to understand these spatial descriptions and extract the relevant genes and associations. The different components of the parser are:

+ **Gene Extraction**: Extraction of genes from the GPML file requires the identification all occurrences of the GPML DOM attribute name: “DataNode”. This enables the parser to obtain the Gene Name, Gene NCBI ID and the spatial coordinate locations associated to this datanode.

+ **Spatial Clustering**: Activating/inhibiting relationships are described across gene clusters spatially. Therefore the genes have to be spatially clustered to determine spatially the activator/activatee or inhibitor/inhibitee relationships.

  An example is reproduced in Figure 4.6. Here the genes CDK2, CYCE inhibits the entire cluster of ORC genes. Hence we group the genes CDK2, CYCE together as one cluster, and the ORC genes as another.

  Using the coordinates from the genes obtained above, a nearest neighbour technique is employed to organise the genes into their respective clusters. Basically this nearest neighbour algorithm groups genes together if their distance apart is below a threshold (empirically determined as 100 pixels). A short pseudo-code of this algorithm is as follows:

```
function nearest_neightbour(nodelist n)
    set nodecluster = n
    while (change)
        change = false
        For each cluster1 in nodecluster
            For each cluster2 in nodecluster
                   if dist(cluster1, cluster2) < threshold
                   cluster1 = combine(cluster1, cluster2)
                   change = true
                endif
            end for
        end for
    loop
    return nodecluster
end function nearest_neightbour
```

We basically treat each individual gene node as a node cluster. All the node clusters are cycled with two nested for loops. If we find within the nested loop that there are two gene clusters whose distance are below a certain threshold (100 pixels here), we combine these two clusters. The algorithm continues to run until the clustering of the genes remains constant.

The potential running time for this algorithm is $O(n^4)$ but the average running time is $O(mn^2)$ where m refers to the number of clusters present.

This gives us the exact spatial coordinate information of each gene cluster. This allows us to form relationships between gene clusters (explained in the next section).

- **Relationship Extraction**: Relationships within the GPML files are represented by the attribute keywords: “Arrow” for activating and “T-Bar” for inhibiting. These attributes provide their spatial coordinate information of activating and inhibiting relationships. The challenge here is associating the correct gene clusters to each relationship.

  We can thus easily locate all activating and inhibiting relationships within a pathway. The challenge then becomes associating the correct inhibitee / inhibitor or activatee / activator gene clusters to each relationship. This we carry out via image processing means.

  By representing relationships as a straight line, this relationship line in the spatial space is extended until it intersects with the nearest gene clusters on both sides of it. This technique assigns the activator/activatee or the inhibitor/inhibitee gene clusters to both sides of the relationship.

  We refer to an earlier diagram in Figure 4.6. Extending the relationship line, the relationship spatially intersects gene clusters CDK2 and CYCE and the ORC gene clusters. This assigns gene cluster CDK2 and CYCE as inhibitors and the ORC gene cluster as the inhibitee.

  The equation of the relationship line, can be calculated using the geometry equation

$$y = mx + c \tag{4.1}$$

  We next test each gene cluster (found in the previous section) to assign it to its proper relationship. The pseudocode is as follows:

```
function finding_relationships(relationships rs, nodecluster n)
    For each relation in rs
        For each cluster1 in nodecluster
             if intersection(cluster1, relation)
                [dist, tag] = distance(cluster1, relation)
                relation.insert(dist, tag)
            end if
        end for
    end for
    return rs
end function finding_relationships
```

The innermost loop takes note that we assign the most suitable gene cluster to the relationship (By most suitable, here we refer to the cluster spatially nearest to the relationship).

For metabolic pathways (because the gene relationship is neither activating or inhibiting), the GPML attribute keyword is simply a “Solid” line attribute. In such instances, the relationship type attribute to the gene pair would be “neutral”.

4.3.2.2 KEGG

Data is obtained from KEGG via a series of API calls and processing the data (SOAP format) returned. An API call is issued to obtain all the pathways first. This returns all the relevant pathway IDs stored within KEGG. Separate API calls are made for each pathway IDs to obtain the genes and gene pairs present for each specific pathway.

The API call to obtain all the pathways for homo sapiens is: serv.list_pathways(“hsa”) where “serv” refers to the created wdsl object to communicate with KEGG and hsa refers to the “homo sapiens species”.

The API call for gene and gene pair extraction from a KEGG pathway are:

+ Gene Extraction: serv.get_genes_by_pathway(X)

+ Gene Pair Extraction: serv.get_element_relations_by_pathway(X)

where X refers to the pathway ID within KEGG.

The data returned from the API calls is parsed, extracting the critical components and populating them into the database.

4.3.2.3 Ingenuity

All pathway repositories allows users to download pathway data in pictorial formats. However, such pictorial formats are usually useless to automated computational analysis. Hence most repositories would support downloading of their data in a text format too. Ingenuity though supports downloading of pathway data in a pictorial format. This forces our pathway data extraction from Ingenuity to be done manually, an extremely painful and time consuming process.

In fact, as Ingenuity is a commercial company and earns its revenue stream from charging the download of data, it is not surprising that Ingenuity legally disallows batch download of data from Ingenuity (hence only releasing information in the pictorial format). We have taken this legal issue into consideration and disallow the download of Ingenutiy's information from our servers.

4.3.3 Data Updates

An expiry date is assigned to all information stored within our database cache. Upon reaching the expiry date, scripts are triggered to run, automatically extracting information from the reference databases (KEGG and Wikipathways) and populating it into our database cache.

4.4 API Implementation: Short specification of the API

We provide a single common API to access the different databases. This common API works in this manner: A local database serves as a cache, storing data from the other repositories. Requests for information from the different repositories are directed to this cache to obtain the required information. To ensure that our interface is always kept up to date, automatic incremental updates are run periodically to extract the latest information from the different repositories. This process creates a unified interface for the different databases, as well as a unified database where graphs of the same pathway are merged.

The database is available at www.pathwayapi.com. The recommended requirements are: 1 Mbps internet connection, 1GHz Processor, 512MB Memory. The database cache currently stores a total of 397 gene pathways, 21,314 genes and 60,900 gene pairs. From this API, access to pathways from both the integrated and the individual sources are provided. Further details of this interface can be found at (PathwayAPI, 2009).

The API was written in PHP, and data transfer in JSON format. We have chosen JSON over SOAP or XML because:

+ JSON is lighter in weight, transmitting less information over the internet. Client applications therefore executes faster.
+ JSON has the ability to easily represent most general data structures such as records, lists and trees.
+ With SOAP or XML, dedicated parsers are always required on the client. JSON is innately supported by most programming languages, eliminating the need for client parsers.

Some implemented functions of the API include:

+ GetDatabase: Returns all repositories supported by our API. No parameters are required for this function. The usage example is: www.pathwayapi.com/api/API_GetDatabase.php and the sample results returned is:
["KEGG","Ingenuity","Wiki"].

+ GetGene: Returns the NCBI GeneID of the gene. This function takes the name of the gene as the parameter. An usage example is: www.pathwayapi.com/ api/API_GetGeneID.php?SearchGene=MDM1. The format returned is: [[“MDM1”,“252867”],[“MDM1”,“56890”]]. In this case, there are two separate gene ids that are returned.

+ GetDBPathways: Returns the all pathway names and IDs of a specific repository. Only the database name needs to be submitted to the function. For instance, www.pathwayapi.com/api/API_GetDBPathways.php?DatabaseName=KEGG. Here, the following will be returned: [kegg{“1”: {“DatabaseName”: “KEGG”, “PathwayName”: “Glycolysis Gluconeogenesis - Homo sapiens (human)”}, “2”:{“DatabaseName”: “KEGG”, “PathwayName”: “Citrate cycle (TCA cycle) - Homo sapiens (human)”}, etc...] where “1” refers to the Pathway ID, “KEGG” refers to the name of the database and “Glycolysis Gluconeogenesis...” refers to the name of the pathway.

+ GetPathway: Returns the pathway ID of a specific pathway of a repository. Posting the name of the pathway in this manner:
www.pathwayapi.com/ api/API_GetPathway.php?Pathway=Apoptosis will return the jason format like: [[“Apoptosis - Homo sapiens (human)”,“KEGG”,"140”], [“Apoptosis Signaling”,“Ingenuity”,“210”]].

  In this instance, this implies that there are at least two pathways with the “Apoptosis” keyword witin their pathway names. The two pathways occurs in the KEGG databases and in the Ingenuity databases. The pathway id associated to each is 140 and 210 respectively.

+ GetPathwayGenes: Returns all the GeneID of a specific pathway of a repository. Providing the pathway ID to this function returns the user all the genes within this pathway in this manner:

www.pathwayapi.com api/API_GetPathwayGenes.php?Pathway=7 Resulting in: [“231”:“AKR1B1”,“2538” :“G6PC”,“2548”:“GAA”,“2582”:“GALE”] where “231” refers to the gene ID and “AKR1B1” refers to the name of the gene.

+ GetGenePathways: Returns all the pathways which a gene occurs. In the opposite note, this function returns all the pathways which a supplied gene occurs in. http://www.pathwayapi.com/ api/API_GetGenePathways.php?SearchGene=7157 We obtain the following database pathway pairs: [“128”:“MAPK signaling pathway - Homo sapiens (human)”,“134”:“Cell cycle - Homo sapiens (human)”,“135”:“p53 signaling pathway - Homo sapiens (human)”] In this example, “128” refers to the Pathway ID and "MAPK signaling pathway - Homo sapiens (human)" refers to the name of the pathway.

+ GetPathwayInteractions: Returns all interactions within a pathway of a database. Passing in the ID of the pathway, the API returns all the interactions within the pathway. www.pathwayapi.com/ api/API_GetPathwayInteractions.php?Pathway=7 will result in [[“231”,“AKR1B1”,“2584”,“GALK1”,“Activate”] ,[“231”,“AKR1B1”,“2585”,“GALK2”,“Activate”]] In the example above: “231” and “2584” refers to the IDs of the gene pair “AKR1B1” and “GALK1” refers to the corresponding genes of the ID.

+ GetPathwayDiff: Get the differences in genes and gene interactions across pathways. This function requires the user to supply the IDs of the two pathways he/she wish to check on the difference in. The call below shows the difference in genes and gene interactions between pathway 7 and pathway 8. www.pathwayapi.com/ api/API_GetPathwayDiff.php?Pathway1=7&Pathway2=8

  This gives the following results where: [[“AKR1B1”,“G6PC”,“GAA”,“GALE”,“GALK1”], [“ALDH2”,“ALDH3A1”], [“AKR1B1_GALK1”,“AKR1B1_GALK2”,“AKR1B1_GLA”],[]]

Where [“AKR1B1”,“G6PC”,“GAA”,“GALE”,“GALK1”] refers to the genes within pathway 7 not in pathway 8.

[“ALDH2”,“ALDH3A1”] refers to the genes within pathway 8 not in pathway 7.

[“AKR1B1_GALK1”,“AKR1B1_GALK2”,“AKR1B1_GLA”] refers to the gene interactions within pathway 7 not in pathway 8.

[] refers to the gene interactions within pathway 8 not in pathway 7. This set is empty because all interections in pathway 8 are in pathway 7.

4.5 Example Usage of API

Here we show an example on how we use this API to find out the pathways with the most number of differentially expressed genes within a microarray experiment.

1 Find out the genes within the microarray experiment which are differentially expressed. This step can be easily achieved by using the t-test or any other statistical test mentioned in the earlier chapters.

2 Find out which pathways these genes belong to. For each gene found differentially expressed in the previous step, we carry out iteratively the following API call: http://www.pathwayapi.com/api/API_GetGenePathways.php?SearchGene= individual_gene_id where individual_gene_id refers to the gene ID for each individual gene within the microarray experiment

This gives us a whole list of pathways where the gene is found to be differentially expressed.

3 Depending on the number of occurrences each pathway appears, as well as the number of genes present within the pathway, we can assign a score to each individual pathway. The pathway with such highest score can be deemed as the pathway that is most significant (due to the large number of differentially expressed genes within that pathway) for the microarray experiment.

4.6 Discussions

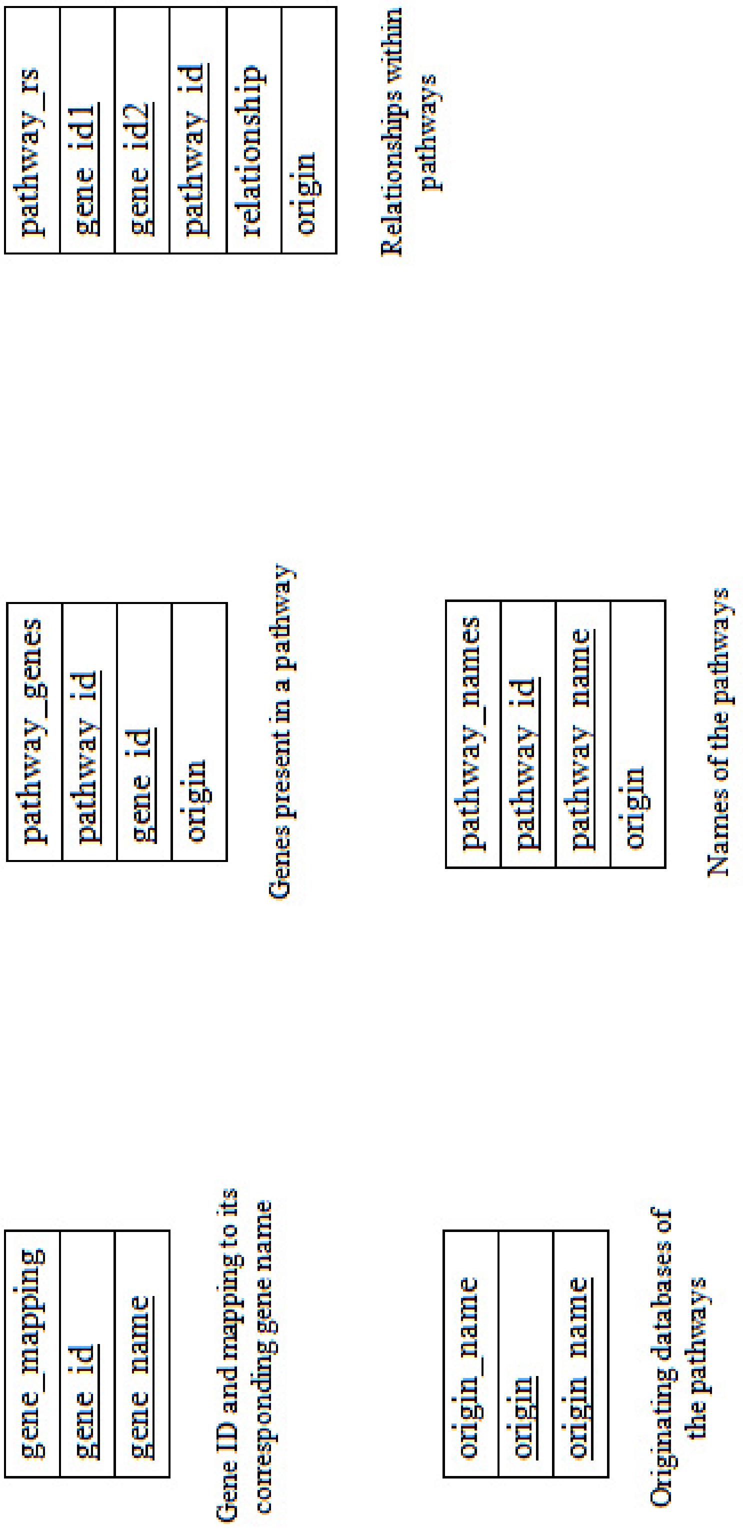
The advantages of having such an aggregated database with common API are:

+ Consistent data Where possible, we have combined information of similar pathways from different repositories. Thus if Gene A is quoted in Pathway P of Repository X, and Gene B is quoted in Pathway P of Repository Y, both Genes A and B are quoted in Pathway P of our new repository. This goes the same as well for the pathway relationships.

+ Consistent referrals to gene names We refer to all proteins and genes via consistent NCBI gene id-s. If a protein is being referenced, we represent that protein as the genes it translates to.

+ Consistent methods for data acquisition Computer scientists can use our single API to obtain pathway information from supported repositories. This reduces the need for the computer scientist to customise a new acquisition method based on the repository. In addition, data returned is in a single consistent format.

A single microarray experiment contains at least 10,000 genes. If we were to just require to find out the pathways each gene appears in, it would already take 10,000 API calls and this might take some time. To expedite processing, we have allowed users to download the entire database in a SQL dump. This allows the user to host his/her own data. Information that the user requires can be obtained directly through SQL queries, thereby reducing processing time.



pathway_rs
gene id1
gene id2
pathway id
relationship
origin

Relationships within pathways

pathway_genes
pathway id
gene id
origin

Genes present in a pathway

pathway_names
pathway id
pathway name
origin

Names of the pathways

gene_mapping
gene id
gene name

Gene ID and mapping to its corresponding gene name

origin_name
origin
origin name

Originating databases of the pathways

Figure 4.5: The database schema used for our pathway api database (PathwayAPI, 2009). Each column represents a separate table. The name of the table is represented as the first row in each column. The rest of the rows within each column refers to the table entities. The underlined rows refer to their primary keys.

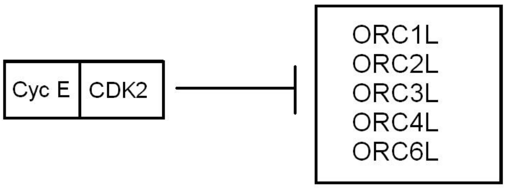


Figure 4.6: A diagram depicting how relationships would look like in Wikipathways (Wikipathways, 2004). Here, the genes CycE and CDK2 are depicted as inhibiting the ORC class of proteins, which contains genes ORC1L, ORC2L, ORC3L, ORC4L and ORC6L

Chapter 5

Pathway API — Evaluation

Chapter Synopsis

Summary

Chapter 4 outlined our techniques and methodology for the creation of our unified database. This chapter evaluates our database to ensure that it is consistent, comprehensive and compatible for integration into computational microarray algorithms. We make quantitative evaluations by comparisons between three databases (KEGG (Kanehisa and Goto, 2000), Ingenuity (Ingenuity, 1998) and Wikipathways (Wikipathways, 2004)). This comparison is made by calculating the percentage overlap in genes, gene pairs and pathways across these three datasets. Finally a qualitative examination is being done to arguing that our standardised nomenclature (such as gene references and key pathway features), makes our database compatible for integration into computational techniques.

Conclusions

After unifying information from three different biological databases (KEGG, Ingenuity, Wikipatways), we show that indeed there is a low level of consistency among them (specific figures provided below). Generally, the level of consistency for genes in similar pathways across databases ranges from 0% to 88% while the level of consistency for interacting genes pairs ranges from 0%-61%. Hence biological information stored within our unified *database is more comprehensive information as compared to the individual databases KEGG, Ingenuity or Wikipatways.*

Qualitatively, because we (1) unified pathway data from three independent biological sources, (2) created consistent data access methods and formats, (3) standardised nomenclature such as gene references and pathway key features, researchers will be able to access complete information (from different databases) easier by using a single access method (PathwayAPI) and using standardised nomenclature.

5.1 Introduction

Pathway API was implemented over a period of one to two years. A version of the system has been made public online at www.pathwayapi.com. Access to data within the database can be achieved by invoking the database API. The entire database has also been released to my colleagues at NUS and they are using the data on a limited basis.

5.2 Quantitative Evaluation

5.2.1 Database Consistency

Pathway databases (eg KEGG, Ingenuity, Wikipathways) have always been assumed to be consistent because they share a common data source: published literature (Wikipathways is based on established databases like KEGG or Netpath, hence sharing the same roots of published literature). We show here that this assumption is not true.

We reuse the definition that was done for the gene pair count and the gene agreement count in the earlier chapter. In the case of metabolic pathways, we define an interacting gene pair as proteins that catalyse adjacent steps in the pathway.

The three databases represent some of their pathway entries not as genes but as proteins or symbols depicting protein families or classes. In such instances we replace all such proteins and symbols with the genes they represent. For example, suppose that A activates B within a pathway, where A and B are symbols representing protein classes that are products of 3 genes and 2 genes respectively. We replace A activates B by 6 new activating relationships. We claim the validity of this replacement method because it exactly captures all the genes and relationships the original curator had intended. All statistics calculated here are based on the expanded relationships.

As listed in the earlier chapters, our investigation into database consistency began with a manual comparison on the agreement of the apoptosis pathway across databases. This achieved an extremely low level of agreement within a range of 11%-16% (Gene Pair Agreement Percentage) and 32%-46% (Gene Agreement Percentage). For clarity, we reproduce the results once again in Table 5.1.

The next step involves an automated extraction for the apoptosis pathway between the databases. The results are shown in Table 5.2. The results indicate a range of 12%-14% (Gene Pair Agreement Percentage) and 30%-46% (Gene Agreement Percentage). This is indicative that the above mentioned gene matching procedure is reliable and not missing significant numbers of equivalent genes.

Table 5.1: Table showing data overlap for apoptosis Pathway. This table shows the manual calculation of the gene/gene pair differences between the different repositories for the apoptosis pathway.

Apoptosis Pathway			
	KEGG x Ingenuity	KEGG x Wiki	Ingenuity x Wiki
Gene Pair Count:	151 vs 3374	151 vs 133	3374 vs 133
Gene Count:	89 vs 169	89 vs 82	169 vs 82
Gene Overlap:	33	38	26
Gene % Overlap:	37%	46%	32%
Gene Pair Overlap:	21	21	15
Gene Pair % Overlap:	14%	16%	11%

Table 5.2: This table shows the calculation of the gene/gene pair differences between the different repositories for the apoptosis pathway based on the automated processing described in this paper.

Apoptosis Pathway			
	KEGG x Ingenuity	KEGG x Wiki	Ingenuity x Wiki
Gene Pair Count:	182 vs 3486	182 vs 155	3486 vs 155
Gene Count:	84 vs 185	84 vs 79	185 vs 79
Gene Overlap:	28	36	24
Gene % Overlap:	33%	46%	30%
Gene Pair Overlap:	22	22	18
Gene Pair % Overlap:	12%	14%	12%

We subsequently followed up with an automated extraction and comparison between the databases. The ranges are 0%-88% (Gene Agreement Percentage) and 0%-61% (Gene Pair Agreement Percentage). These numbers confirm our earlier suspicion that there is an extremely low level of consistency between the databases. For results depicting the level of overlap for the other pathways refer to Table 5.3, Table 5.4 and Table 5.5.

Table 5.3: Table showing data overlap between KEGG and Ingenuity. The first column shows the names of the different pathways analyzed. Second column shows the number of genes within each pathway across the two databases. The third column shows the gene pairs present within each pathway. The column on gene overlap refers to the number of genes overlapping between KEGG and Ingenuity. The real numbers refer to the number of overlapping genes while the percentage figure in brackets refer to the percentage overlap. The last column refers to the gene pair overlap. In the same manner, the real numbers refer to the number of overlapping gene pairs while the percentage figure in brackets refer to the gene pair overlap percentage.

KEGG x Ingenuity				
Pathway Name	Gene Count	Pair Count	Gene % Overlap	Pair % Overlap
Apoptosis Signalling	89 vs 169	151 vs 3374	33(37%)	21(14%)
Axonal Guidance	129 vs 213	308 vs 1843	85(66%)	159(52%)
Calcium Signalling	179 vs 51	582 vs 202	18(35%)	0(0%)
Cell Cycle-G2M	119 vs 13	78 vs 18	11(85%)	11(61%)
Cell cycle	119 vs 31	78 vs 59	26(84%)	6(10%)
Fc epsilon RI Signalling	78 vs 75	184 vs 225	61(81%)	108(59%)
JAK/Stat Signalling	155 vs 144	868 vs 3192	42(29%)	88(10%)
Actin Cytoskeleton Signalling	217 vs 213	672 vs 2297	137(64%)	230(34%)
T cell receptor Signalling	94 vs 63	175 vs 133	41(65%)	39(29%)
TGF-Beta Signalling	87 vs 84	155 vs 113	12(14%)	5(4%)
VEGF Signalling	74 vs 69	240 vs 167	29(42%)	27(16%)
Wnt Signalling	152 vs 76	778 vs 134	33(43%)	11(8%)

Table 5.4: Table showing data overlap between KEGG and Wiki. Kindly refer to Figure 5.3 for a detailed explanation on the format of the table.

KEGG x Wiki				
Pathway Name	Gene Count	Pair Count	Gene % Overlap	Pair % Overlap
Apoptosis	89 vs 82	151 vs 133	38(46%)	21(16%)
Apoptosis Modulation by HSP70	89 vs 18	151 vs 33	14(78%)	5(15%)
Cell cycle	119 vs 91	78 vs 147	76(84%)	35(45%)
G1 to S cell cycle control	119 vs 67	78 vs 25	45(67%)	1(4%)
Complement and coagulation cascades	69 vs 65	69 vs 107	52(80%)	24(35%)
Focal Adhesion	203 vs 188	706 vs 288	154(82%)	110(38%)
Insulin Signalling	138 vs 159	412 vs 255	66(48%)	13(5%)
MAPK Cascade	269 vs 31	891 vs 55	23(74%)	24(44%)
Notch Signalling	46 vs 46	90 vs 98	39(85%)	32(36%)
Regulation of actin cytoskeleton	217 vs 151	672 vs 244	133(88%)	113(46%)
T Cell Receptor Signalling	94 vs 135	175 vs 261	37(39%)	6(3%)
TGF Beta Signalling	87 vs 52	155 vs 80	23(44%)	6(8%)
Tryptophan metabolism	51 vs 94	233 vs 33	29(57%)	2(6%)
Urea cycle	28 vs 66	69 vs 14	13(46%)	1(7%)
Wnt Signalling	152 vs 61	778 vs 184	49(80%)	34(18%)

Table 5.5: Table showing data overlap between Ingenuity and Wiki. Kindly refer to Figure 5.3 for a detailed explanation on the format of the table.

Ingenuity x Wiki				
Pathway Name	Gene Count	Pair Count	Gene % Overlap	Pair % Overlap
Apoptosis	169 vs 82	3374 vs 133	26(32%)	15(11%)
Calcium Signalling	51 vs 152	202 vs 111	14(27%)	0(0%)
Cell Cycle	13 vs 91	18 vs 147	7(54%)	5(28%)
G1/S Check point Regulation	31 vs 91	59 vs 147	24(77%)	10(17%)
IL-4 Signalling	21 vs 62	21 vs 47	8(38%)	1(5%)
IL6 Signalling	67 vs 100	148 vs 121	21(31%)	4(3%)
Insulin Receptor Signalling	66 vs 159	148 vs 255	40(61%)	12(8%)
TGF-Beta Signalling	84 vs 52	113 vs 80	13(25%)	0(0%)
p38 MAPK Signalling	53 vs 34	88 vs 35	13(38%)	4(11%)
T cell receptor Signalling	63 vs 135	133 vs 261	25(40%)	3(2%)
Wnt Signalling	76 vs 61	134 vs 184	17(28%)	0(0%)

5.2.2 Database Comprehensiveness

This section conducts an independent audit on the comprehensiveness of individual pathway databases.

The pie charts in Figure 5.1 and Figure 5.2 shows the comprehensiveness of databases (with respect to its gene and gene pairs). One can tell from the pie charts that the amount of overlap is very low, depicting a low level of comprehensiveness for all three biological databases analysed.

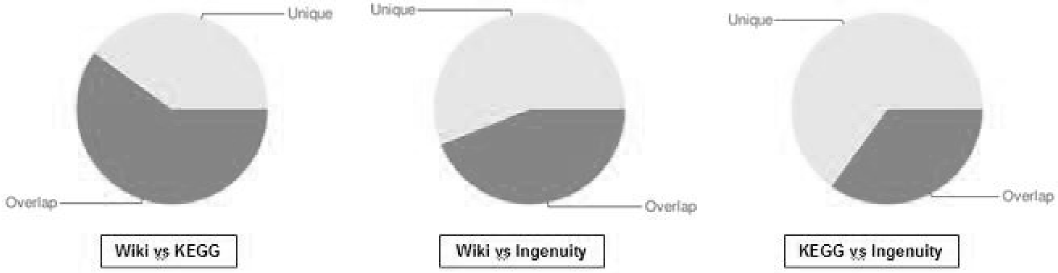


Figure 5.1: Pie charts depicting gene overlap proportions. The lighter shade refers to the proportions of unique genes while the darker shade refers to proportions where there was an overlap of genes. (Graphs generated with Google charts (Google, 2009))

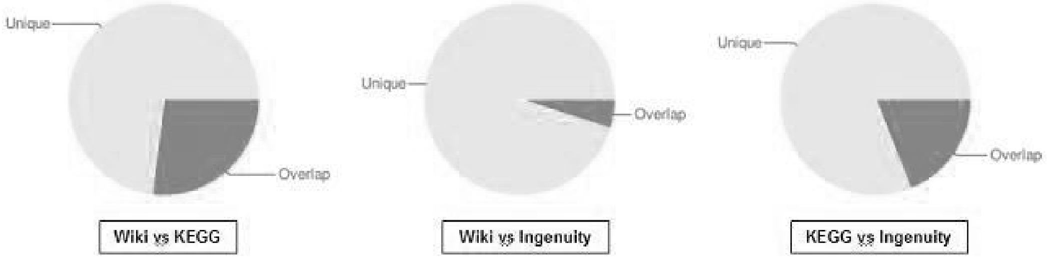


Figure 5.2: Pie charts depicting gene pair overlap proportions. Similarly, the lighter shade refers to the proportions of unique gene pairs while the darker shade refers to proportions where there was an overlap of gene pairs. (Graphs generated with Google charts (Google, 2009))

We use a metric called "Pathway Comprehensive Score" to study database comprehensiveness. This metric first counts the total number of unique pathways present within the three databases (Ingenuity, KEGG and Wikipathways). A score for each database is next calculated by dividing the number of pathways a database hosts by the total number of unique pathways. A score of 0 indicates that the database hosts nil pathways while a score of 1 indicates it hosts all the pathways.

KEGG achieved the highest score of 0.59. This was followed by Wikipathways (0.42) and Ingenuity (0.13). This short study indicates that KEGG Pathways remains the most comprehensive of all databases. This is illustrated by a Venn diagram in Figure 5.3.

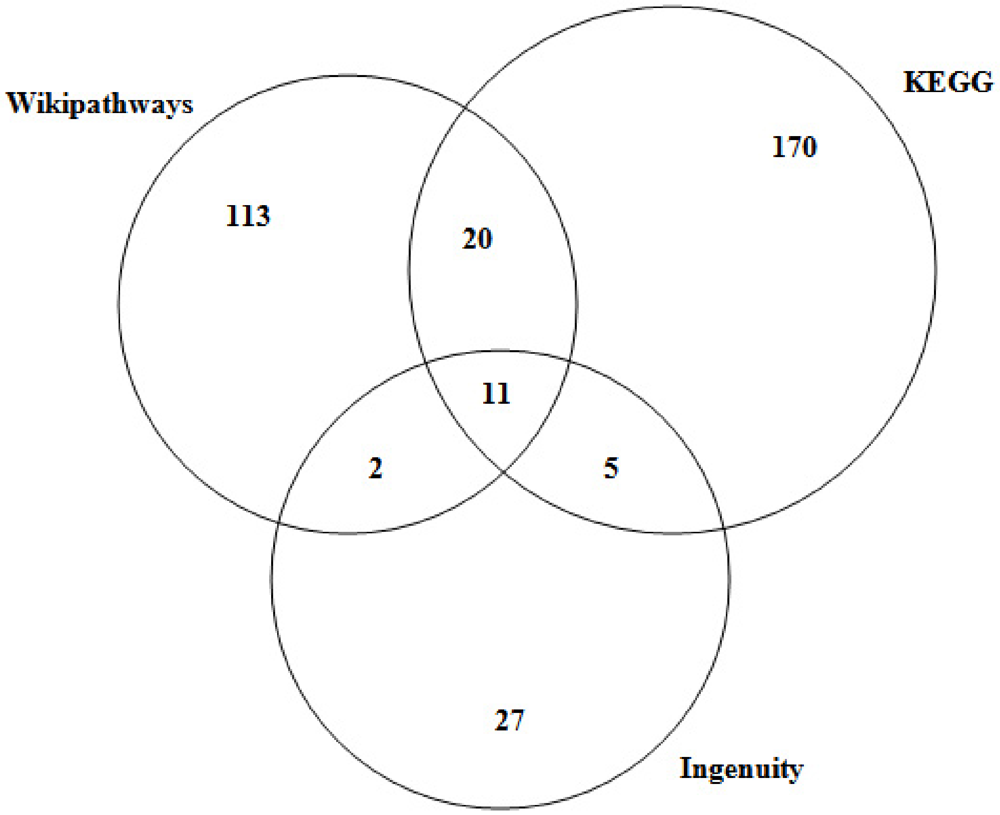


Figure 5.3: Venn diagram depicting overlapping pathways across the three databases, KEGG, Ingenuity and Wikipathways.

We match pathways across separate databases via the following technique: Given a pathway X in database 1, we generate a list of pathways Y in database 2. This list Y is ranked according to the length of the longest common substrings with pathway X. This list is next manually scanned to obtain the pathway which has the closest nomenclature match to pathway X.

To validate this technique of matching pathways, we did the following: Given a pathway X in source 1, we match it against the top three pathways in source 2 that has largest gene overlap with pathway X. 94% of the time we obtained a pathway match identical to the pathway we would have achieved if we had done by our technique.

In addition, for the remaining 6% of these pathways, we carry out the following comparison: Suppose that P-Q are the pathway pairs found by our algorithm and P-R are the pathways found by matching the genes. We found out that the number of gene pair intersections is higher in the P-Q pathway pair compared to P-R pathway pair. Hence we believe our name-based pathway matching is more likely to have correctly matched the pathways.

5.3 Qualitative Evaluation

5.3.1 Database Consistency

Microarray algorithms produce different (subjective) results depending on the database being used. This may cause further inconsistencies and confusions in analysis. This is especially accentuated because the boundaries of signalling pathways are not that clearly defined. Therefore scientifically, making the question of exactly which genes to include in pathways is quite subjective.

To make databases consistent, our strategy is to first aggregate all similar pathways from the different databases. This is followed by combining the gene and gene pair information for each pathway.

This solves the problem where we have genes or gene relationships lacking from certain pathways. Hence we can use pathway which is present across the different repositories.

5.3.1.1 Consistent data access and formats

Having obtained the data and having control over its data access and formats, we can accommodate several standard methods of accessing the data. As mentioned, we have written an API to provide quick access to the data online. However for serious practitioners who would require huge amounts (> 10,000) of API data connections daily, using the API for such purposes would be impractical. For such users, we would recommend downloading our sql dump which contains the entire collection of data available on our database. Doing so would allow the user to setup a database on his own localhost and access the data using very fast sql queries instead.

5.3.1.2 Consistent Molecular Representation

Our representation of proteins, genes etc are a lot more consistent because we convert all such molecular representations (eg genes, proteins, pathways) into individual genes, and represent them by their respective gene IDs (obtained from NCBI).

5.4 Discussions

It is widely accepted that analysing microarray experiments with biological information provides biological inferences of a greater detail. Examples of such analysis are (Draghici et al., 2007; Efroni et al., 2007; Tian et al., 2005).

However, such techniques run into issues if the data source used is not consistent or comprehensive. For example, using the same technique on a different database yields a differing analysis result.

Faced with such an issue, the solution is to integrate biological information across different data sources to obtain a more wholesome analysis. Yet the incompatibility of the different data sources renders this option extremely challenging.

Furthermore, our investigations reveal low levels of consistency, comprehensiveness and compatibility among three popular pathway databases (KEGG, Ingenuity and Wikipathways).

Our strategy of addressing this issue is to create an API (described in the previous chapter) which gives researchers access to various pathway databases of their choice as well as to an integrated database. This integrated database resolves all incompatibilities because now we have one API and one data format. The integrated database is also more comprehensive because it is the union of the data sources. Every gene/edge in any of the three data sources is also in the integrated database. Furthermore, the integrated database is equipped with a API to allow the user to conveniently identify inconsistencies and to resolve them in accordance to his specific application needs.

To ensure fast responsiveness, API connections are made towards a central unified database which keeps a cached copy of the records of the other databases. To ascertain that the cached entries are always kept up to date, entries from the cache are flushed periodically and automatically updated again from the reference databases.

There are many efforts on the aggregation of pathways data (like Reactome (Joshi-Tope et al., 2005), PathCase (Elliott et al., 2008; Krishnamurthy et al., 2003) and MappFinder (Doniger et al., 2003)). There are also many tools to explore, edit and export biological pathways (such as GenMapp (Salomonis et al., 2007), BioCyc (Karp et al., 2005), PathVisio (van Iersel et al., 2008), Cytoscape (Shannon et al., 2003)).

However manipulation of pathways in these earlier works still relies heavily on human intervention with little provision for programming interfaces. Indeed projects like Cytoscape and Pathcase have very sophisticated GUI visualisation tools to help researchers manipulate pathways. Yet such visualisation tools are impractical when one is required to analyse of thousands of genes across hundreds of pathways for each microarray experiment. The nearest to a programming interface was the provision of a AQI (Application Query Interface) (Elliott et al., 2008) where users can recall predefined queries using a web interface. Yet the scope of such queries remains limited and insufficient.

One issue we have with most data aggregators is their lack of explanation on how their data is kept updated. For instance, little mention is made on how the aggregated data is updated from the various repositories. In fact this issue is acknowledged in (Elliott et al., 2008). Here we set an expiry date for every data entry and once it expires, automated scripts are fired off to extract data from the data sources and populate them within our database cache.

Our final point deals with the aggregator's inaction to develop integrated pathway data from their diverse data sources. By standardising gene references and key features within pathways, we have the ability to integrate similar pathways together. As a result our integrated pathways are more comprehensive.

Contrasting to prior available methods, researchers can easily use our API to obtain data for each pathway either from the integrated database or from a specific database of their choice. This gives researchers a straightforward mechanism for incorporating pathway information into their microarray analysis.

However, because we use only known biological information in the analysis, it might suffer from "myopia" as additional new insights without supporting biological basis yet would be left undiscovered.

CHAPTER 6

Disease and Drug-Response Pathway Identification — Algorithm

Chapter Synopsis

Summary

Contemporary techniques for analysing biological information have two main deficiencies: one, they provide results that are not biologically descriptive and two, results obtained from one dataset often do not translate easily across a similar but different dataset. Here we demonstrate the different decision making steps taken to ensure that the results obtained will provide proper biological analysis to the microarray experiments. We first explain how we create the foundation to allow biologically descriptive results by deciding on a proper level of granularity required within the biological pathways of the unified database. Following which we explain how we arrange genes into their respective pathway components to take advantage of the gene-gene relationships within pathways. This is further substantiated with an explanation of statistical testing (which is required to test for significant pathway components). Finally, these are strung together with a detailed explanation and example of our algorithm.

Conclusions

We introduced the concept of a biological "subnetwork" as a suitable degree of granularity for providing biological descriptive results. This concept was chosen such that it could provide detailed biological analysis (such as individual differential gene expression), broad based information (such as the specific topological interactions of these genes within the subnetworks) and even high-level functional information (such as the original pathway of the subnetwork).

In addition, we have paid special attention in our algorithmic design (e.g. in the manner of gene selection, subnetwork scoring, etc) to generate consistent results over different microarray datasets. These algorithmic details are explained in this chapter while the performance of it on actual microarray experiments is demonstrated in the next.

6.1 Introduction

As mentioned earlier, most contemporary algorithms (Liu et al., 2007; Pavlidis et al., 2002; Subramanian et al., 2005) concentrate on finding individual genes or gene pairs which are differentially expressed. We discussed that using such fine granularity of information would provide the researchers with long lists of genes/gene lists thus rendering them ineffective for making biological inferences. Recent advancements have attempted to solve such issues by using gene sets or gene pathways.

GSEA (Subramanian et al., 2005) uses the strategy of using gene sets to detect entire groups of genes acting differentially. The research group for GSEA (at Broad Institute) manually curated their own biological data, placing the genes into individual sets according to various parameters (eg chromosomal location, biological process, molecular function, etc). These gene sets are given to the GSEA algorithm which detects the activated gene sets in the microarray experiment. Because the gene sets are biological correlated to one another, GSEA can immediately provide biological inferences based on the functional gene sets.

The other known solution (used by pathway express (PathwayExpress, 2009), ORA (Khatri and Draghici, 2005) and FCS (Pavlidis et al., 2002)) is to use known biological pathways. These pathways are entire groups of genes which are agreed to have a relationship with one another via consortiums or domain experts. Hence this faces the immediate issue of deciding which pathways are more accurate because the decision of which genes should belong to a pathway is still very subjective (Green ML, 2006). We expect this problem to accentuate as our knowledge of pathways and genes increase.

The main problem of using either gene sets or gene pathways lies in the biological fact that not all the genes in a set / pathway are affected within a phenotype. Indeed it is expected that perhaps one portion (say 5-10 genes) of a pathway are affected within a disease phenotype and a strategy like using entire gene sets or gene pathways result in such information being missed. In addition, although gene sets or pathways are being used, these techniques continue to score entire gene sets or pathways **via the individual genes**. Hence their technique disregards entirely the relationships between the genes of the gene set and pathway (Sivachenko et al., 2007; Subramanian et al., 2005).

Our solution to this is to identify gene regions that are differentially expressed using connected components. We define the term "subnetwork" or "connected component" as "a set of genes and relationships where all genes in the connected component are reachable by all other genes in the (undirected) connected component. Reachability between genes is established by the existence of an undirected path between the genes of the connected component." This is analogous to the definition of "connected components" in classical graph theory (Cormen et al., 2001). In this thesis, we consider connected components as a portion of a pathway that fits the above definition. Using connected components as the level of granularity in analysing our microarray data, we are hence able to:

1. Give detailed information on genes which are differentially expressed.

2. Provide general analysis and biological inferences on connected components which are differentially expressed.

3. Directly focus into relevant portions of a pathway.

6.2 Choice of Datasets

Currently, there are two types of microarray experiments, namely spotted microarrays (which were pioneered by (Eisen and Brown, 1999)) and oligonucleotide microarrays, invented by (Chee et al., 1996).

The difference is mainly due to the difference in manufacturing of the probes. The probes of spotted microarrays (oligonucleotides, cDNA, PCR fragments) are synthesised prior to deposition on the array surface and are then "spotted" onto glass surface. Probes of oligonucleotide microarrays are short sequences designed to match sequences of known open reading frames. These sequences are designed to represent single genes or family of genes.

Numerous studies have been conducted to compare the variation of results across different platforms and results have been shown to be largely inconclusive. Some of these studies have shown that the difference in variation is significant (Kuo et al., 2002; Rogojina et al., 2003; Tan et al., 2003) while others have argued that such differences are acceptable (Ishii et al., 2000; Yuen et al., 2002).

This phenomena is known as “batch effect” (Lander, 1999) which is often experienced when experiments are carried out independently (and especially across different platforms). Batch effects can be considered as non-biological experimental variation often experienced when dealing with multiple batches or platforms of microarray experiments (Rhodes et al., 2004). This makes analysing datasets across batches difficult. Indeed attempts have been made to create algorithms to reduce the batch effect by using eigen vectors or certain forms of mathematical models (Alter et al., 2000; Benito et al., 2004; Johnson et al., 2007). However require a large number (> 25) of patients within a phenotype or these mathematical models fail when some underlying distribution for the data is not met.

For the datasets we have chosen, most of them are from oligonucleotide Affymetrix platforms. However we have also chosen experiments where one dataset is from the oligonucleotide platform while the other is a spotted cDNA microarray. We show that we can obtain consistent results even across different platforms and types of microarrays.

6.3 Creation of Connected Components from Microarray Data

We create connected components for each dataset by first creating a list of genes that are significantly expressed and forming connected components from this gene list based on prior biological knowledge from the integrated database as described in Chapter 4 and Chapter 5.

6.3.1 Derivation of highly expressed set of genes

We derive our set of highly expressed genes of a phenotype A by first ranking the gene expression values of each patient of phenotype A. The top $n\%$ of genes from each patient is next selected. Finally we select genes that appear in the top $n\%$ of genes for at least $m\%$ percent of the patients of phenotype A. This forms the final list GL of genes which are considered to be highly expressed within phenotype A.

This technique removes the need to carry out data normalisation because we are directly ranking genes of each patient according to their expression levels.

Some of the other techniques such as ORA and FCS uses a fixed threshold to obtain genes which are highly expressed. For instance, we could set the threshold to be ”Mean + Std Dev” which implies that all genes with an expression value higher than ”Mean + Std Dev” would be considered as significant. However we abandoned this approach because it introduces additional points of failure. There are multiple techniques put forth to the simple step of array normalisation (Bilban et al., 2002; Quackenbush, 2002; Yang et al., 2002) and the techniques to be used might be very subjective. In addition, these techniques might yield very different results especially if the experiments are from different batches and are especially apparent on different platforms.

Hence we put forth our approach of ranking genes and choosing the top $n\%$ of genes common to $m\%$ of the patients because this technique is general enough to be applied in various scenarios. Our motivation for such a technique is to alleviate batch issues by concentrating on the genes which are expressed relatively higher in the phenotype. The other alternatives include, for example, carrying out array normalisation, adjusting for batch effects followed by using the threshold "Mean + Std Dev". However, we have found in our experiments that our simpler technique already works well enough.

6.3.2 Derivation of connected components

From this list of top genes, we refer to the biological data repository created in Chapter 4. Using the information within that database, we segregate that list of top genes into their respective pathways. This is a list of genes considered highly expressed arranged according to their pathways. We represent genes from the i^{th} pathway as GL_i.

The next step involves finding the relationships within the genes of each pathway. Hence for the i^{th} pathway, we pair up each and every gene within GL_i and check with the data repository if such a gene-gene connection exists within the pathway. (We point out that we are making a simplification here. Instead of looking at relationships between genes, example gene g1 activates gene g2, we focus instead only on connections. This means that regardless of the activating or inhibiting relationships between two genes within a pathway, we only represent them as gene g1 is connected to gene g2. So we lose the directional aspect of relationships. (E.g., we will be unable to determine if the activating relationship between gene g1 and gene g2 is initiated by gene g1 or gene g2.) Again, for the i^{th} pathway, this gives us a list of gene pairs represented as GP_i.

As a final step to obtain the connected components, we first represent each gene within GL_i as a set (with only one member gene). Next we iterate through each gene pair within GP_i. For each gene pair (gene g1 and gene g2), we test if the set containing gene g1 is equal to the set containing gene g2. If they are not equal, the sets are merged. Else we move on to the next gene pair.

This algorithm is made more lucid with the pseudo code and example as shown below:

```
connected_components(GL)
   for each gene g within GL
      make-set(g)
   end for

   for each edge (g1, g2) within GL
      if set(g1) != set(g2)
         merge(g1, g2)
      end if
   end for
```

This small code takes in the gene set GL_i as input. The first loop creates a set from each gene within the list. The second loop goes through all the individual gene pairs and merges the gene sets from each gene within the gene pair if they are not equal. Note that the function $set(g1)$ means a retrieval of the gene set containing the gene g1.

An example, taken from (Cormen et al., 2001) can be seen in Table 6.1. Here we begin with 10 genes and 7 relationships within the pathway. The first column gives the different relationships and the first row gives the initial sets, where each set only contains just a single individual gene. Each new row gives an iteration of the second for loop in the pseudo code. For instance, in the first iteration, the pair (b, d) is processed. This merges the sets which contains the gene b and the gene d. Hence the combined set $\{b, d\}$. Take note how the connected components grow during the iterations, giving us 4 different connected components ($\{a, b, c, d\}$, $\{e, f, g\}$, $\{h, i\}$ and $\{j\}$) at the end of the routine.

Table 6.1: Example depicting how connected components are generated. The first row refers to the starting sets where each gene is in its individual set. The first column refers to the 7 relationships. Each new row shows the processing of the gene sets according to the gene relationship of the first column. For instance, in the first relationship (b, d), the gene sets b and d are merged to form the gene set b, d.

	{a}	{b}	{c}	{d}	{e}	{f}	{g}	{h}	{i}	{j}
(b,d)	{a}	{b,d}	{c}		{e}	{f}	{g}	{h}	{i}	{j}
(e,g)	{a}	{b,d}	{c}		{e,g}	{f}		{h}	{i}	{j}
(a,c)	{a,c}	{b,d}			{e,g}	{f}		{h}	{i}	{j}
(h,i)	{a,c}	{b,d}			{e,g}	{f}		{h,i}		{j}
(a,b)	{a,b,c,d}				{e,g}	{f}		{h,i}		{j}
(e,f)	{a,b,c,d}				{e,f,g}			{h,i}		{j}
(b,c)	{a,b,c,d}				{e,f,g}			{h,i}		{j}

Figure 6.1 shows clearly in a pictorial form how we obtain the connected components from the microarray experiment and prior biological information.

After obtaining the connected components, we score them with a metric (introduced later in this chapter). The connected component is scored across both phenotypes of the experiment. Hence each connected component ends up with two scores, one for phenotype A and the other for phenotype B. Details of our scoring technique is elaborated later in Section 6.6.

Our aim then is to devise a statistical test to find components whose score distribution differs consistently across phenotypes of a disease. The next section provides more information on how we devise the statistical tests to reject / not reject this hypothesis.

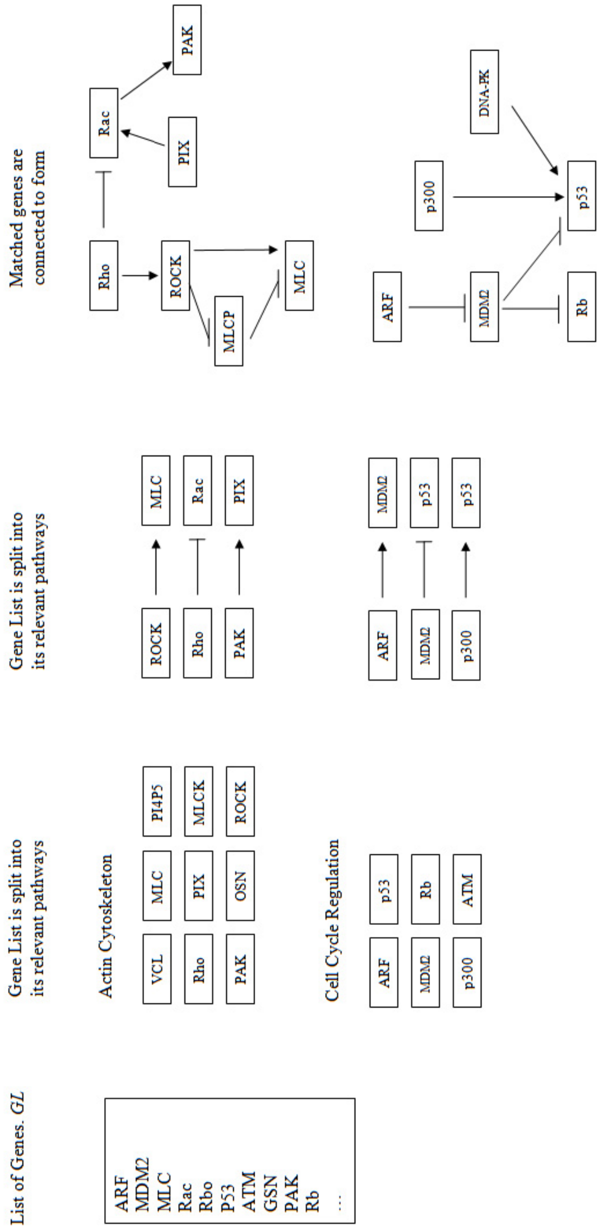


Figure 6.1: Image depicting how we split the genes GL up to their respective pathways, match them into e gene pairs and finally connected into their respective cc subnetworks.

6.4 Statistical Background

Before analysing the different types of hypothesis testing that are available, we first go through some of the basic technical details of hypothesis testing.

6.4.1 Hypothesis Testing

A hypothesis is a proposed explanation for an observable phenomenon. A null hypothesis is a specific baseline statement to be tested and it usually implies a state of ”no difference”. Here, the null hypothesis would probably take the form: that there is no difference between the means of the expression values across the two phenotype groups.

Hypothesis testing is a scientific process to examine (on the observed phenomenon) if a hypothesis is plausible or not. In general, hypothesis testing follows the next five steps.

1. State the null hypothesis
2. Determine the level of significance and whether it is a one-tailed or two-tailed test. This gives us the thresholds required to reject the null hypothesis.
3. Compute the test statistic and p-value
4. Reject or do not reject the null hypothesis based on the criteria (test size, significance level, tail test) and the calculated test statistic or p-value.
5. Draw conclusions from rejection / non rejection of the null hypothesis

The classical approach to hypothesis testing computes a test statistic or p value and compares it with a critical value or significance value. If the test value goes above the critical value or the p value goes below the significance level, the null hypothesis is rejected. When rejecting the null hypothesis, two errors may emerge. One where a null hypothesis is rejected when it is true (Type I errors) and two where we failed to reject the null hypothesis when it is not true (Type II errors). One of the manners used in reducing Type I errors is in the proper selection of the level of significance. A level of significance at 0.05 means that we are comfortable with the risk of having five incorrect predictions for every 100 trials. Hence a level of significance at 0.1 would mean that the test is rather lenient, more apt to produce Type I errors and hence less convincing. A strict level of significance of 0.01 imply a stricter and more convincing test.

6.4.2 Types of Hypothesis Testing

With this understanding the concepts of hypothesis testing, we analysed the different techniques available that can be used for hypothesis testing between our experimental phenotypes. Such means of testing are expounded as below. In addition to explaining the technical details of each technique, we explain how this technique can be applied in our context.

6.4.2.1 t-test

t-tests are employed to test the significance of the difference between the means of two normally distributed populations. If the means of the population differ by a large amount, it would have a large t-value and normally a low p-value. If the p-value is sufficiently low, we reject the null hypothesis that the means are equal and conclude that the means are significantly different. Such tests are typically referred to as "unpaired" or "independent samples" t-tests, because the two samples being compared are independent from one another. An example of such an independent test occurs when we are trying to determine if diabetic subjects are more likely to be obese then in normal non-diabetic subjects. Conversely, an example of a dependent test occurs when we measure the size of a patient's tumour before and after a treatment. If the treatment is effective, we would expect the tumour to have shrunk after it. Such a statistical test is often called a paired dependent t-test. Here we are concerned with the former.

In our analysis, we let the statistical null hypothesis be the case where the distribution of scores for the connected components are similar across the two phenotypes. We assume that the patients from both phenotypes have the same variance. The number of degrees of freedom for either group is the total sample size minus two. This would be used for significance testing. Hence the formula of such a t-test would be:

$$t = \frac{\bar{X_1} - \bar{X_2}}{S_{X_1X_2}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \tag{6.1}$$

where

$$S_{X_1X_2} = \sqrt{\frac{(n_1-1)S_{X_1}^2 + (n_2-1)S_{X_2}^2}{n_1+n_2-2}} \tag{6.2}$$

$S_{X_1X_2}$ is defined as the estimator of the common standard deviation of the two samples, n_1 is the number of samples in the first phenotype and $\bar{X_1}$ is the mean of the values in the patients of the first phenotype. n_2 is correspondingly the number of samples in the second phenotype and $\bar{X_2}$ is the mean of the values in the patients of the second phenotype.

Application

For instance, we take the DMD dataset and compare between the patients suffering from DMD with normal subjects. With a score assigned for each component to each patient. We create an array of component scores for the DMD patients as well as a corresponding array of component scores for the normal subjects. We simple apply 6.1 to the two arrays and calculate the t-score for component. This is repeated for the rest of the connected components.

6.4.2.2 chi-square

In this section we describe the technical details of Pearson's chi-square test. This chi-square test will test the frequency distribution with that of a theoretical distribution to determine if the distributions between observed frequency distribution differ from that of the theoretical distribution. A key requirement is the mutual exclusivity of the individual events. For instance, if a patient appears in the BCR leukemia subtype, that same patient cannot appear in the MLL leukemia subtype.

The value of the test statistic is as follows:

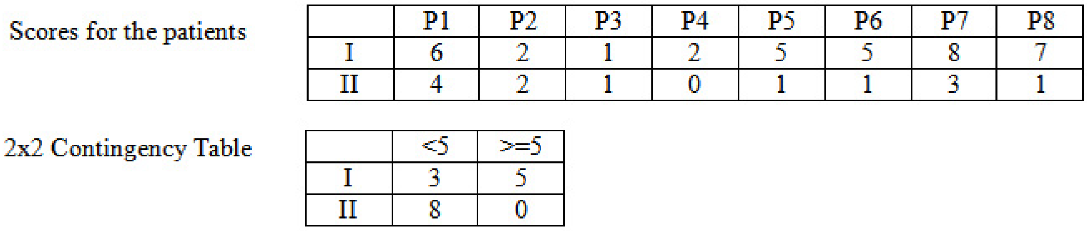
$$\chi^2 = \sum_{i}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{6.3}$$

Where n is the number of possible outcomes and the degrees of freedom is equal to the number of possible outcomes minus 1. However the chi-square test makes a few assumptions. Firstly, a sample with a sufficiently large size is assumed (to reduce the probability of committing a Type II error). It is essential that each observation cell within the contingency table has a size of five or more else the approximation to the chi-square distribution will not be valid.

Application

Using the DMD dataset, we compare between the patients suffering from DMD with normal subjects. The normal patients are considered as the expected dataset while the DMD patients are considered as the observed dataset. A score for each component is assigned to each patient. We pick a threshold for the score to split the patients into the 2x2 contingency table. An example on how this is done is in Figure 6.2. The threshold chosen here was the value 5. We have for Phenotype I 4 patients with a component score greater or equal than 5, 3 patients less than 5. For Phenotype II all patients were below the threshold of 5. From the creation of this contingency table, we were able to determine the p-value for each component and determine the components reject the null hypothesis. Such a rejection would mean that the distribution for that component is different between the DMD patients and the normal subjects. This is repeated for the rest of the connected components.

When the chi-square test was deployed unto our experiment, it was able to differentiate the components which behaved differently across the two phenotypes. There was however an extremely serious issue, which was the size of values within the 2x2 contingency table. Most of the time, the values fell below < 5. Although we understand that we could have alleviated this problem by using the Yate's correction, we are aware



Scores for the patients

	P1	P2	P3	P4	P5	P6	P7	P8
I	6	2	1	2	5	5	8	7
II	4	2	1	0	1	1	3	1

2x2 Contingency Table

	<5	>=5
I	3	5
II	8	0

Figure 6.2: Illustration of how we create the 2x2 contingency table. The top table refers to the scores obtained for the patients. P1, P2, .. P8 refers to the identities of individual patients. I and II refers to the phenotype. By counting the number of patients in the top table, with score $\leq$ 5 for phenotype I, we obtain he figure of three in the top left cell of the contingency table. The reminder of the contingency table can be calculated in a similar fashion.

of the fact that Yates' correction tends to overcorrect resulting in a more conservative result, failing to reject the null hypothesis when it should.

6.4.2.3 One-way ANOVA

The one-way ANOVA is a statistical test of testing if the means of several groups are all equal. Therefore, it can be considered to be a generalisation of the two-sample t-test for more than two independent groups. The motivation for using the ANOVA can be best illustrated with a simple example. For instance, we are testing the response of three different subtypes of ALL Leukemia, BCR, MLL and E2A. On the surface, one of the ways to determine if one subtype differs from the other subtype is by performing a separate t-test between each of the possible pairs of subtypes. For instance, between BCR and MLL, MLL and E2A and lastly BCR and E2A. Such a strategy is not feasible because if t-tests are performed on multiple pairs of means, the probability that one phenotype is classified as "significant" at the 0.05, is substantially greater than 0.05. Hence the ANOVA was created to address such complications.

ANOVA uses a metric known as the sum of squares (SST) which is given by the formula below:

$$SST = \sum_{i=1}^{r} \sum_{j=1}^{c} (X_{ij} - \bar{\bar{X}})^2 \tag{6.4}$$

Where r is the number of rows in the table, c the number of columns and $\bar{\bar{X}}$ the grand mean of the entire table.

The next metric is known as the Treatment Sum of Squares, given by:

$$SSTR = \sum_{j=1}^{c} r_j(\bar{X}_j - \bar{\bar{X}})^2 \tag{6.5}$$

Where r_j is the number of patients in the j^{th} phenotype and $\bar{X}_j$ is the mean of the j^{th} phenotype. (An example illustrating the calculation of SST and $SSTR$ can be found in Table 6.2 and Table 6.3.)

Table 6.2: Sample figures to illustrate the Anova algorithm. The figures refers to the microarray expression values for the different leukemia subtypes, BCR, MLL and E2A for a single gene.

BCR	MLL	E2A
16	20	18
15	19	19
17	21	18
15	16	23
20	18	18

Table 6.3: Sample figures illustrating the calculation of SST and SSTR. The raw expression values are taken from Table 6.2 and the formulae to calculate the values referred from Equation 6.4 and Equation 6.5.

Metric	BCR	MLL	E2A	Total
Mean	16.6	18.8	19.2	18.2
SST_j	17.2	14.8	18.8	70.4
SSTR_j	12.8	1.8	5.0	

Lastly, the sum of squares is given by:

$$SSE = \sum_{i=1}^{r}\sum_{j=1}^{c}(X_{ij} - \bar{X}_j)_2 \tag{6.6}$$

Finally we obtain two variance estimates called the $MSTR$ and the MSE. These are provided in Equation 6.7 and Equation 6.8 respectively. $MSTR$ measures the degree of the variance between the different phenotypes BCR, MLL and E2A while MSE reflects the degree of variance of the patients within the same phenotype. Understanding this, we would expect that if $MSTR$ is larger than MSE, we would be more liable to reject the null hypothesis. Conversely if MSE is larger than that of $MSTR$, we would be liable not to reject the null hypothesis.

$$MSTR = \frac{SSTR}{c-1} \tag{6.7}$$

$$MSE = \frac{SSE}{N-c} \tag{6.8}$$

where c refers to the number of phenotypes in the experiment and N refers to the total number of observations. The F score is simply taken to be the division between $MSTR$ and MSE.

$$F = \frac{MSTR}{MSE} \tag{6.9}$$

Application

As mentioned above, the ANOVA method is widely used to determine if the means of various groups of phenotypes are equal or not. In the experiments conducted in this thesis, we only compared between two phenotypes. Hence when ANOVA is executed on just two phenotypes, it effectively reduces to the t-test (ANOVA is merely a generalisation of the t-test to compare the means between more than two phenotypes) Therefore there is no difference between using the t-test or ANOVA here. However should we decide to include more phenotypes into our experiments at a future date, using the ANOVA test to find test the significance of the differences of means between > 2 phenotypes would be an extremely viable option.

The remaining two techniques that follow are, strictly speaking, non statistical tests. However, because these techniques are capable of measuring the response differences between the two phenotypes, they are considered as potential candidates for testing the difference in scores for the connected components between the phenotypes.

6.4.2.4 Entropy

The definition for entropy used is similar to that used by Shannon in information theory. It is the measure of the uncertainty associated within a variable. For instance, a fair coin has an entropy of 1 because it is uncertain if the outcome is going to be heads or tails. Conversely in tossing a biased coin which only turns up heads, it has an entropy rate of 0 since every toss of the coin is predictably heads. The entropy of can be calculated as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) log_2 p(x_i) \tag{6.10}$$

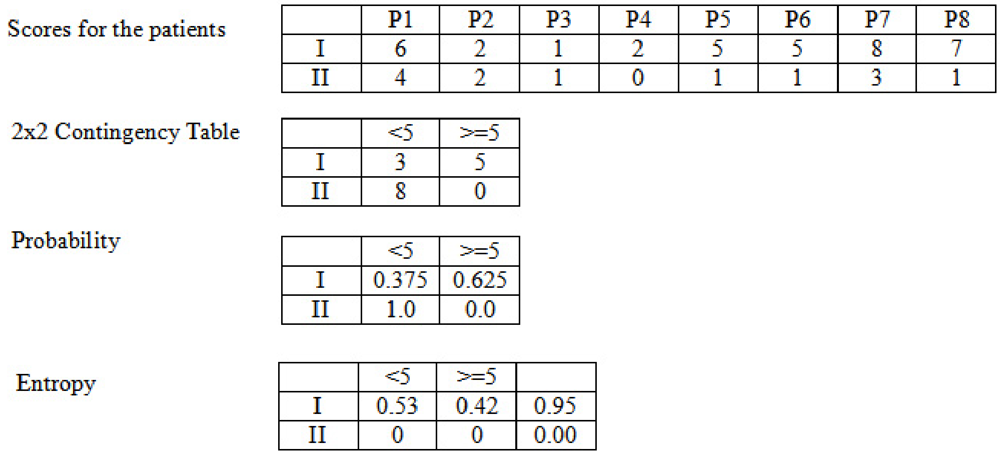
Application

In our context, it is the measure of uncertainty associated with the score of a connected component has in both phenotypes of the experiment. In the same manner as the chi-square test, we first split the patients into a 2x2 contingency table. The probability values were calculated as in Figure 6.3 for each entry of the contingency table (column wise, according to each phenotype). We chose the option of calculating the entropy value for each component for each phenotype. The Table 6.4 below how the entropy values change when the data is in extremes.

Although we are able to determine if the scores are skewed within the phenotypes, we have no direct way of finding out how the scores in the phenotypes are skewed. Hence we are unable to differentiate the following two scenarios:

1. Scores in phenotype I are skewed high and the scores in phenotype II are skewed low (and vice versa)
2. Both the scores in phenotype I and phenotype II are skewed high (or both of them are skewed low)

The entropy scores for both phenotypes in both scenarios will be low. However the first scenario is desirable (because it shows a difference in the distribution of scores between the phenotypes) while the second is not (because they would have the same score distribution). Therefore because of this inability to distinguish between these two important scenarios, this technique for testing may not be very applicable.



Scores for the patients

	P1	P2	P3	P4	P5	P6	P7	P8
I	6	2	1	2	5	5	8	7
II	4	2	1	0	1	1	3	1

2x2 Contingency Table

	<5	>=5
I	3	5
II	8	0

Probability

	<5	>=5
I	0.375	0.625
II	1.0	0.0

Entropy

	<5	>=5	
I	0.53	0.42	0.95
II	0	0	0.00

Figure 6.3: Illustration of how we create the probability values column wise from the 2x2 contingency table. The top two tables are reproduced from Figure 6.2.

Table 6.4: Table depicting the behaviour of entropy values according to the behaviour of the phenotype values. For instance, when the values of Phenotype I and Phenotype 2 are skewed, both their entropy values will be high.

Phenotype 1	Phenotype 2	Entropy 1	Entropy 2
Random	Random	High	High
Random	Skewed	High	Low
Skewed	Random	Low	High
Skewed	Skewed	Low	Low

6.4.2.5 Gini Coefficient

The gini coefficient is a measure of inequality in a sample group. It is often used in economics to measure the difference in equality among different income groups. It can be obtained by dividing the difference between every possible pair of individual – the ”relative mean difference” – by the mean size. With μ defined as the mean size, the gini coefficient can be calculated by using the formula in Equation 6.11.

$$G = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - x_j|}{2n^2\mu} \tag{6.11}$$

The lower the coefficient is, the more equal the distribution is. When the coefficient is 0, it would mean that there is perfect equality and everyone receives the same percentage of the total income. When the coefficient is 1, there is perfect inequality and a single person gets 100% of the income and the remaining (infinite) population receive none.

Application

In our context, we use the scores obtained in the components as the income figure. Hence we take the relative mean difference between the component scores to depict the gini coefficient. Similar to our metric using entropy, we wish to find scenarios where the component scores are as biased as possible. Meaning that the scores for the phenotype I are high and the scores the phenotype II are low, and vice versa. Unfortunately this metric suffers from the same limitations as using the entropy as a metric (see Table 6.5).

Table 6.5: Table depicting the behaviour of gini coefficient values according to the behaviour of the phenotype values. For instance, when the values of Phenotype I and Phenotype 2 are uniform, both their entropy values will be low.

Phenotype 1	Phenotype 2	Gini Coeff 1	Gini Coeff 2
Uniform	Uniform	Low	Low
Uniform	Skewed	Low	High
Skewed	Uniform	High	Low
Skewed	Skewed	High	High

In a parallel fashion, the gini index is unable to differentiate the scenarios as in Section 6.4.2.4, where the scores for both phenotypes are skewed high (or skewed low) between the scenario where the scores of phenotype is skewed high, the scores of other phenotype is skewed low. In this case, the gini coefficient gives a high gini coefficient in both phenotypes for both scenarios, and there is no direct way of segregating instances of one scenario from the other.

6.4.2.6 Final selection of hypothesis test: t-test

To reiterate the objective that the test is supposed to serve, it's main purpose is to easily select components which gives unfair distribution scores to different phenotypes. If such components are found, it would mean that the component score is skewed towards some particular phenotype within the disease and not the other phenotype. The three key scenarios where this occurs is when:

1. Scores in phenotype I are skewed high and the scores in phenotype II are skewed low (and vice versa)
2. Scores in phenotype I are skewed high and the scores in phenotype II are uniform (and vice versa)
3. Scores in phenotype I are skewed low and the scores in phenotype II are uniform (and vice versa)

Naturally components which exhibit the first characteristic should be given a high significance value rather than components exhibiting the second and the third scenario. Given the above criteria, only the t-test, chi-square and ANOVA fulfil them. Both the entropy and gini coefficient fail as they are unable to sieve out components that fall under the first criteria. Hence we are left with only the t-test, chi-square and ANOVA. However, some of the parameters required for the chi-square test are below the required threshold. Lastly, the metrics ANOVA and t-test are both suitable candidates for the hypothesis test, fulfilling all three criteria stated. (Indeed ANOVA is a generalization of the t-test when applied to more than two variables.) To make our algorithm as general as possible, it would be best to use the ANOVA as the hypothesis testing method of choice. However as all our experiments in this thesis only have two phenotypes, both the ANOVA and t-test produce the same results. Hence, for simplicity, we have used the t-test for hypothesis testing in this thesis.

6.4.3 Calculation of p-value

The previous section gives an explanation of hypothesis testing and examines the different types of hypothesis tests that can be used in our experiment. The merits of each type of test are implemented and we analysed how suitable these hypothesis tests are. Based on this analysis, we conclude that the t-test is the most suitable choice as it could meet our requirements of finding components that gave different distributive scores across the phenotypes.

We form the component scores of both phenotypes, we obtain a t value (based on the formula for t-test) commonly known as a t-score. This t-score is required to be converted to a p-value to decide if the component should be significant or not. If its p-value is below a certain threshold (say 0.05), we would reject the null hypothesis that the mean of the phenotypes are statistical similar to one another. This means that we have a 5% chance that component gives a statistically different score between the two phenotypes. There are two main techniques of obtaining the p-value from the t-score, one by statistical distribution tables and the other from permuting the empirical data. We describe both techniques briefly below.

6.4.3.1 Usage of Statistical Tables

In probability and statistics, the t-distribution statistical tables is a probability distribution that is used when estimating the p-value based on the t-score and its sample size (degrees of freedom). Generally it is bell-shaped, flatter for smaller sample sizes. Hence getting the p-value from statistical tables involves the straightforward way of reading it from a table with the t-score and its corresponding degrees of freedom. The crucial assumption made here is that getting the p-value from the statistical table assumes that population follows a normal distribution.

6.4.3.2 Permutation Analysis

The usage of t-test tables requires that the distribution follows a normal distribution. However if such an assumption is not certain, the other technique of creating the distribution is by permuting the labels randomly from the experiments and creating a distribution through such randomisations. This technique is known as permutation analysis.

Specifically, for each dataset, we permute the phenotype labels, create a new set of connected components and recompute the scores of this new set of connected components. This process of permuting the phenotype labels, creating connected components and lastly recomputing of the scores of the connected components is repeated for n times. The set of all the randomly generated scores for the connected components forms the null distribution for dataset. The empirical p-value for each component can then be calculated relative to this null distribution. More importantly, the permutation of class labels preserves gene-gene correlations and thus provides a more biologically reasonable assessment of significance than would be obtained by permuting genes (Subramanian et al., 2005).

6.4.3.3 Final Selection of Distribution for p-value Calculation: Permutation Test

Due to the fact that we have no manner of determining if the scores from the connected components follow a normal distribution, taking the p-values from statistical tables may result unnecessarily in both Type I and Type II errors. Hence we use the randomisation process to produce the distribution for each dataset. Using this distribution, we calculate the p-values for each dataset and obtain the components which are deemed as significant. We render components significant if their empirical p-values are less than 0.05.

6.4.4 Calculation of the p-value from Randomisation Test

From the distributions obtained from the randomisation processes, we have two methods of calculating the p-value. From the previous section, we have pointed out that during the randomisation process we randomise the phenotype labels and create randomised connected components. For each such randomly generated connected component, they have a corresponding t-score as well as their component size. Hence we have two distinct distributions. The first is the random distribution of the sizes of such components. The second is the random distribution of the t-scores of such components. We can either user both distributions to create our p-value or just the latter distribution to create the p-value.

6.4.5 Significance of subnetworks

We explain here how the p-value is generated from the permutation tests. There are two ways to generate this p-value, from (1) the combined distribution of t-scores and component sizes of the subnetworks or (2) the distribution of the t-scores of the subnetworks. Both techniques are explained briefly below.

6.4.5.1 Usage of component sizes and t-scores

As mentioned above, one of the methods of obtaining the score from the permutation process is via both the randomly generated component sizes and their t-scores. Permuting the phenotype labels of the microarray data, and repeating this permutation for n iterations gives us three distributions: the distribution of the size of the subnetworks produced using the randomised datasets (Figure 6.4); the distribution of the score of the subnetworks produced using the randomised datasets (Figure 6.5); and finally a combined histogram distribution of the size and score of the subnetworks produced using the randomised datasets (Figure 6.6).

Figure 6.5 gives the distribution of the scores of the components. The x-axis refers to the score obtained by the randomly generated components and the y-axis refers to the percentage of components who has that score. Hence in that figure, we have approximately 13% of the components with a score of 1, 5% of components with a score of 2, etc. Figure 6.4 shows the size of the randomly generated components. Here we find that we have more than 45% of the components with a size of 2 genes, 20% with a size of 3 genes, etc. The last figure, Figure 6.6 somewhat shows a histogram of these two distributions. The x-axis is the random component score while the y-axis gives the random component sizes. The z-axis gives the empirical p-value associated with such a component score and size. We calculate this p-value by dividing the number of components with at least size $s1$ and score $s2$ by the total number of components generated by the permutation process.

We can thus find the set of significant subnetworks by calculating the empirical probability of each subnetwork. Specifically for each subnetwork of size $s1$ and score $s2$, we find (from the histogram) the empirical probability of a subnetwork of at least $s1$ genes with score of at least $s2$ randomly occurring. As an example, Figure 6.4, Figure 6.5 and Figure 6.6 shows the distribution for dataset on childhood ALL subtype identification. The blue portion in Figure 6.6 depicts subnetworks whose p-value is below that of 0.05. This figure clearly shows that the subnetworks which we select (size $\geq$ 6 genes and score $\geq$ 2.0) have a p-value small enough (0.05) to render such subnetworks significant.

6.4.5.2 Usage of t-scores

The other technique of obtaining the p-value is to use the scores of the components, based on the distribution shown in Figure 6.5. Specifically, this means that scores which rank in the top 5% (with respect to the random distribution) are considered as significant.

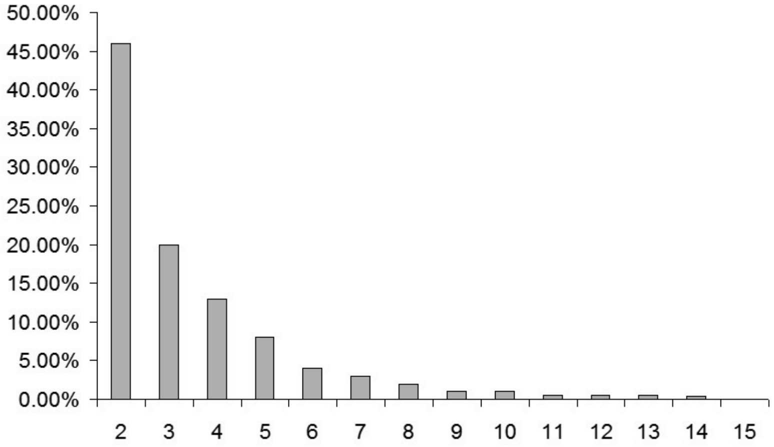


Figure 6.4: Size (number of genes) distribution of the randomly generated subnetworks. The x-axis refers to the number of genes present within each subnetwork. The y-axis refers to the percentage of randomly generated subnetworks with that corresponding subnetwork size.

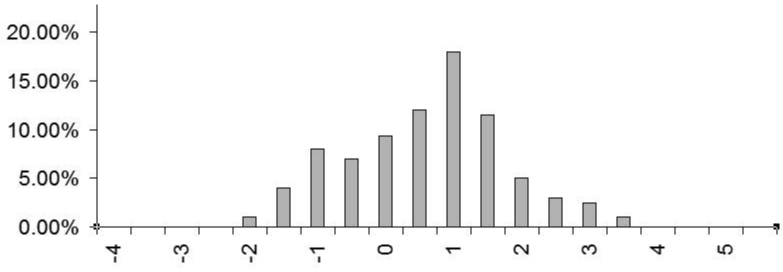


Figure 6.5: Score distribution of the randomly generated subnetworks. The x-axis refers to the components scores of the randomly generated components. The y-axis refers to the percentage of randomly generated subnetworks having that component score.

6.4.5.3 Final selection of technique for p-value calculation: Usage of t-scores

After implementing both techniques, we realised that when both component scores and component sizes are used, components that are large in size but with very low scores would be deemed as significant. Because of the low score, we realised that they do not

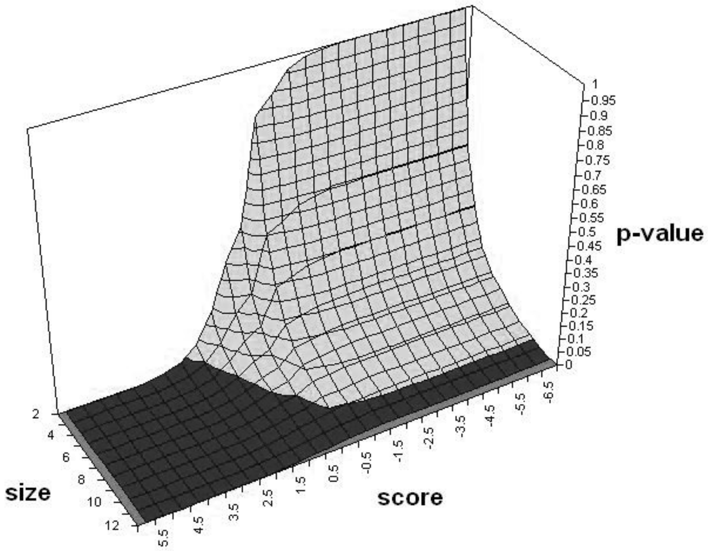


Figure 6.6: Three dimensional histogram of the size and score distribution of the randomly generated subnetworks. Note that the portion shaded in blue are the subnetworks rendered significant.

give good segregation results across the phenotypes and are deemed significant largely because of their huge size. In contrast, when we used the t-scores to calculate the p-values, the components deemed significant could obtain better segregation across the two phenotypes.

6.5 Detailed Approach

This section documents the individual steps of our technique. We first hypothesise that specific biological processes within pathways are relevant to specific diseases. Thus our approach concentrates on identifying these biological processes that we termed “subnetworks”. These subnetworks should be largely the same across independent datasets of the same disease. Because the probability of such a subnetwork of highly expressed genes randomly occurring is sufficiently low, we are able to conclude that these subnetworks have a strong biological relevance with respect to the disease. Furthermore, such a subnetwork provides intricate information on the interplay and relationship between the genes, which is advantageous in guiding subsequent research.

This technique also removes sporadic genes that appear solitary within a biological pathway (because of their higher possibility of being a false positive).

Only two types of gene-gene relationships are considered: inhibition and activation. In the example in Figure 6.7, we see the genes ATM, CHK1, CHK2 and MDM2 with the relationships: ATM activating CHK1, CHK2 and MDM2 inhibiting p53. Thus we define the term “relationship” between a pair of genes X and Y as a situation where either X “activates” Y or X “inhibits” Y.

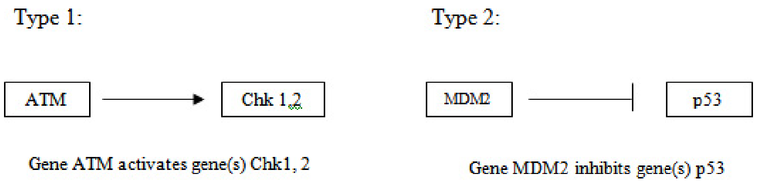

Type 1:
ATM
Chk 1,2
Gene ATM activates gene(s) Chk1, 2
Type 2:
MDM2
p53
Gene MDM2 inhibits gene(s) p53

Figure 6.7: Example of the two gene-gene relationships. Left: Type 1 relationship where gene ATM activates both genes Chk1 and Chk2. Right: Type 2 relationship where gene MDM2 inhibits gene p53. (Image reproduced from (Soh et al., 2007))

Because of the fine granularity of analysis, the pathway repository must allow us to easily segregate the original microarray data into its relevant pathways, gene relationships and subnetworks. Due to the large amount of data, the pathway repository must also facilitate the development of automated analysis workflows. The repository therefore is required to have the following characteristics:

- Gene annotations have to be consistent with that in microarray experiments.
- Individual gene relationships within pathways have to be provided.
- The database must have a programmatic interface to access the data.

However, this set of stringent criteria eliminates contemporary pathway databases such as Ingenuity (Ingenuity, 1998), BioPax (Kotecha et al., 2008), and GenMapp (Dahlquist et al., 2002), and we are left with KEGG (Kanehisa and Goto, 2000). However, KEGG has a number of limitations. Firstly, its collection of pathways is not sufficiently comprehensive (Green ML, 2006). For example, our analysis in previous sections showed that 78.8% of pathways in Ingenuity and 64.4% of pathways in Wikipathways are not contained in KEGG. Secondly, KEGG still uses an old-fashioned SOAP/XML interface. So we developed PathwayAPI (PathwayAPI, 2009) which offers the combined pathway information of KEGG, Ingenuity, and Wikipathways along with a modern JSON-based application programming interface.

Our technique (to be described later) is applied on the disease types listed below with two different datasets analysed independently for each disease type. The selection of the two datasets for each disease is made because they were used to compare gene selection methods in earlier papers (Zhang et al., 2009). In addition, the two datasets for each disease type are from different platforms, thus providing a more stringent test as they make it harder for the gene selection algorithms to consistently select the same genes independently from the two datasets. The disease types of interest include:

- Leukemia: Comparison between leukemia subtypes ALL and AML. Golub (Golub et al., 1999) uses the Affymetrix HU6800 GeneChip with 47 ALL and 25 AML patients. Armstrong (Armstrong et al., 2002) uses the Affymetrix HG-U95Av2 GeneChip with 24 ALL patients and 24 AML patients.

- Childhood Acute Lymphoblastic Leukemia (ALL) Subtype: Comparison between two subtypes of childhood ALL leukemia, E2A-PBX1 and BCR-ABL. Mary (Ross et al., 2004) uses the Affymetrix HG-U95Av2 GeneChip with 15 BCR-ABL patients and 27 E2A-PBX1 patients. Yeoh (Yeoh et al., 2002) uses the U133A GeneChip with 15 BCR-ABL patients and 18 E2A-PBX1 patients.

- Duchenne Muscular Dystrophy (DMD): Comparison between patients suffering from DMD and normal patients. Haslett (Haslett et al., 2002) uses the Affymetrix HG-U95Av2 GeneChip while Pescatori (Pescatori et al., 2007) uses HG-U133A GeneChip. Haslett contains 24 samples from 12 DMD patients and 12 unaffected controls and Pescatori consists of 36 samples from 22 DMD patients and 14 controls.

- Lung Cancer (Squamous): Comparison between patients suffering from squamous cell lung carcinomas and normal patients. For lung cancer, the cDNA microarray data consisted of 13 samples with squamous cell lung carcinomas and five normal lung specimens (Garber et al., 2001), while the data by Affymetrix human U95A oligonucleotide arrays consist of 21 squamous cell lung carcinomas and 17 normal lung specimens (Bhattacharjee et al., 2001).

6.6 Methods

Overview Suppose that the phenotype investigated on is d and the remaining phenotypes are simply classified as $\neg d$. We first extract genes which are highly expressed within this phenotype d from the microarray experiment. This set of genes is next segregated into their respective subnetworks using apriori biological information from the pathway repository. This gives us a list of subnetworks cc (whose genes are highly expressed) within d. A score (depending on the size of the subnetwork and its consistency among the patients) is next calculated and assigned to each subnetwork. Finally we estimate the p-value of every single subnetwork within the list and keep those which are significant. This is elaborated in the following steps:

Step 1: Subnetwork Extraction We create a ranked gene list for each patient within a phenotype according to the gene expression level of that patient. From this ranked gene list we extract only the top $\alpha\%$ of genes for each patient. This condensed gene list is referred to as G_{P_i} for the i^{th} patient P_i. We next iterate across gene lists G_{P_i}

only for patients of phenotype d, extracting only genes which appear in more than $\beta\%$ of the patients of phenotype d. This creates a list of genes GL which turns up highly expressed across most of the patients of phenotype d. Finally, using the programmatic interface of the database, gene list GL is segregated up the gene list into the respective subnetworks. In our experiments, α is taken to be 10 and β to be 50.

To segregate GL into the different subnetworks, we first split gene list GL into its pathways and the gene-gene relationships within these pathways. (We highlight that a gene is allowed to appear in more than one pathway.) Next by treating each gene as a vertex and each gene-gene relationship as an edge, we can easily locate the connected components (subnetworks) formed by these edges (gene-gene relationships) and vertices (genes). This process is illustrated in Figure 6.1.

Step 2: Subnetwork Scoring We assign a score vector $Ssp_{sp,d}$ with respect to phenotype d to each subnetwork sp within cc according to Equation 6.12.

$$Ssp_{sp,d} = \langle Scc_{sp,1,d}, Scc_{sp,2,d}, ..., Scc_{sp,n,d} \rangle \tag{6.12}$$

Where n is the number of patients in phenotype d. The formula $Scc_{sp,i,d}$ for the i^{th} patient (also the i^{th} element of this vector) is given by:

$$Scc_{sp,i,d} = \sum_{j=1}^{g} Sg_{sp,j,d} \tag{6.13}$$

$Sg_{sp,j,d}$ refers to the score of the j^{th} gene (say, gene x) in the subnetwork sp for phenotype d. (This score $Sg_{sp,j,d}$ is given by Equation 6.14) and is simply given by:

$$Sg_{sp,j,d} = k/n \tag{6.14}$$

Here, k is the number of patients of phenotype d who have both gene x highly expressed (top $\alpha\%$) and n is the total number of patients of phenotype d. The entire Step 2 is repeated for the other disease phenotype $\neg d$, giving us the score vectors, $Ssp_{sp,d}$

and $Ssp_{sp,\neg d}$ for the same set of connected components. The t-test is finally calculated between these two vectors, creating a final score for each subnetwork within cc.

Step 3: Subnetwork Significance We repeat Steps 1 and 2 for all the phenotypes in the dataset to extract a list of subnetworks SN. The significance of the observed subnetworks is estimated by randomly permuting the phenotypes labels, re-extracting the subnetworks and recomputing their scores. This generates a null distribution for the score and size of the subnetworks. The p-value of each subnetwork is then calculated relative to this null distribution.

A Randomly swap the phenotype labels of the patients, recreating the subnetworks and recalculating their scores.

B Repeat [A] for 1,000 permutations. This creates a two dimensional histogram of the scores and sizes of the subnetworks.

C Estimate the nominal p-value of each subnetwork by using the histogram created in point [B].

Finally, we consider subnetworks whose p-value was sufficiently small ($\leq$ 0.05) to be significant. Doing so would provide us with an independent set of significant subnetworks SN for each dataset. Using our algorithm, we have managed to show that we are able to obtain consistent significant subnetworks across different datasets of the same disease. This is illustrated in further details in the next chapter.

6.7 Example

We present some short results here based on the DMD datasets from (Pescatori et al., 2007) and (Haslett et al., 2002) to illustrate the working of our algorithm.

Running the algorithm concurrently on both datasets and comparing the significant components across the datasets, we saw an overlap percentage of 58.33% (7 overlapping components) between the two datasets. Overlap of the genes within the components was computed and an overlap percentage value of 69.23% was obtained. More detailed comparisons with the other algorithms will be presented in the following chapter.

How we obtain the significant components is as follows: In the Pescatori dataset, we have 22 DMD patients and 14 normal patients. Our first objective is to locate the connected components that have a majority of their genes highly ranked in the majority of the DMD patients.

We first rank the genes of each patient according to their expression values. The top 10% of the genes is next extracted from the ranked list of each DMD patient. From the top 10% gene list from each DMD patient, we select only the genes which appear in more than 50% of the patients. This provides us with a list of genes that are generally highly expressed among the DMD phenotype for the Pescatori dataset. We construct a set of connected components from this list of genes and our aggregated database.

We next calculate a score of each connected component for each patient (from both phenotypes) according to the formula provided in 6.12. Finally we calculate the p-value of each connected component by taking the t-statistic test of the connected component score across both the DMD and Normal phenotype patients. Existance of a connected component with high t-value suggests that there is a significant difference between the DMD and Normal patients within the datasets, thus making that particular component more significant.

The p-value is calculated via a randomisation process similar to the randomisation processes seen in (Subramanian et al., 2005). Basically, using the data from the Pescatori dataset, we do a random assignment of the 22 DMD and 14 normal phenotypes. With this new set of data with randomised phenotypes, we repeat the procedure for calculation of the t-score again. This involves ranking of genes, obtaining the genes which are significant, recreating the subnetworks once again and finally calculating a t-score for each subnetwork. This randomisation process is repeated for 1000 permutations, producing a distribution of randomly generated connected components with their relevant t-scores. This distribution is used to calculate the p-value of each of the connected components of the original Pescatori dataset. From this original set of components, we find the components with scores higher than 95% of the component scores from the randomised distribution. These components are rendered as significant components.

CHAPTER 7

Disease and Drug-Response Pathway Identification — Results

Chapter Synopsis

Summary

We compare our technique with several popular methods of microarray analysis such as SAM (Tusher et al., 2001), t-test (Cui et al., 2005) and GSEA (Subramanian et al., 2005) in this chapter. This comparison is made on four different disease types and eight different datasets (Leukemia (Armstrong et al., 2002; Golub et al., 1999), Leukemia Subtypes (Ross et al., 2004; Yeoh et al., 2002), DMD (Haslett et al., 2002; Pescatori et al., 2007), Lung Cancer (Bhattacharjee et al., 2001; Garber et al., 2001)). We make our comparisons by finding:

1. *The significant pathway overlap between datasets of the same disease*
2. *The significant gene overlap between datasets of the same disease*
3. *The size of subnetworks obtained from our technique compared with the size of subnetworks obtained from the t-test*
4. *Consistency of significant genes with genes obtained from t-test*
5. *Biological relevance of sample subnetworks*

Conclusions

We illustrate that we consistently outperform the other techniques in our experiments. Specifically we achieved,

1. *a significant pathway overlap of* 47.63% *to* 83.33% *as compares to GSEA (*0% *to* 55.6%*).*
2. *a significant gene overlap of* 51.18% *to* 93.01% *as compares to that of the GSEA (*2.38% *to* 28.90%*), t-test (*49.60% *to* 73.01%*) or SAM (*49.96% *to* 81.25%*).*
3. *subnetworks of a larger size (5 to 16 genes) as compares to t-test (2 to 5 genes).*
4. *subnetworks whose genes are also marked as significant by the t-test*
5. *biological validation of two sample subnetworks by existing biological literature.*

This clearly demonstrates that our technique generates significant subnetworks and genes that are more consistent across datasets compared to the other popular methods available (GSEA, t-test and SAM). The large size of subnetworks which we generate indicates that they are generally more biologically significant (less likely to be spurious). To validate our results, we show that most of our genes from the generated subnetworks have also been considered significant by the t-test. In addition, we have chosen two sample subnetworks and validated them with references from biological literature. This shows that our algorithm is capable of generating descriptive biologically conclusions.

7.1 Introduction

To demonstrate the utility of our algorithm, we employ it to analyse the differential response between phenotypes of a few diseases. This is employed independently across two different datasets of the same disease.

To show that the output connected components from both datasets of the same disease are consistent, we first analyse the overlap of significant connected components over both datasets of the same disease. A high level of overlap means that the results are more biologically significant, being more consistent over datasets of the same diseases. This is compared with GSEA where we ran the GSEA algorithm over both datasets of the same disease, found the significant pathways from each dataset and calculated the pathway consistency overlap across the two datasets for each disease.

After analysing the overlap on connected components / pathways, we concentrate on analysing the overlap of significant genes from the two datasets of the same disease. A gene list is formed by taking the genes within all the significant components of each dataset. Next we calculate the percentage overlap within the gene list across both datasets of the same disease. This gene percentage overlap is compared with three other algorithms: t-test, SAM and GSEA.

With the significant genes (obtained from the t-test) from the two datasets of the each disease, we make two further comparisons:

1. We create connected components from this list of genes to compare the sizes of connected components obtained from the t-test with that from our technique. The purpose is to find out which technique generates larger (and possibly more significant) components.

2. We analyse if the genes found within our connected components are also contained within this list of significant genes. This test acts as an independent check that genes obtained from our algorithm have also been confirmed to be significant by another algorithm.

With the connected components obtained from our technique, we score them using our formula in Equation 6.12 and show a few sample histograms depicting the scores for the connected components. These sample histograms differentiate the patients of the different phenotypes for the different diseases.

For ease of explanation, the results for our technique are named under the acronym SCC, which simply stands for “significant connected components”

7.2 Significant Subnetworks Overlap

For each disease, two lists of significant subnetworks are identified by applying our technique independently on the two different datasets for the disease. We next calculate the percentage overlap between the two lists of significant subnetworks.

This result is compared with another algorithm that extracts significant gene lists from microarray data, GSEA. The individual pathways from the database (PathwayAPI) and their associated genes are used as input gene sets for GSEA. Hence running GSEA with this database of pathways gives us a selected set of pathways deemed as significant by GSEA. GSEA is applied to both datasets of the same disease. For each dataset, we obtain a list of pathways significantly expressed and remove the pathways whose FDR q-value falls below 0.25. Finally we calculate the percentage intersection between the remaining pathways within these two lists.

Results indicate that our technique (as compared to GSEA) consistently gives a higher percentage overlap for different datasets of the same disease. Here, our technique obtained a high overlap percentage for these datasets (47.63% to 83.33%). As an example from Table 7.1, the percentage overlap of pathways in determining the ALL Subtype (second row in that table) in SCC is 47.63% while that for GSEA is 23.1%. The full results can be observed in Table 7.1. Table 7.2 shows the number of overlapping significant pathways for each disease type.

Table 7.1: Table showing the percentage overlap significant pathways between the datasets. Each row refers to a separate disease (as indicated in the first column). Each disease is tested against two datasets depicted in the second and third column. The overlap percentages refer to the pathway overlaps obtained from running SCC (column 4) and GSEA (column 5).

Disease	Dataset 1	Dataset 2	SCC	GSEA
Leukemia	Golub	Armstrong	83.33%	0%
ALL Subtype	Ross	Yeoh	47.63%	23.1%
DMD	Haslett	Pescatori	58.33%	55.6%
Lung	Bhattacharjee	Garber	90.90%	0%

Table 7.2: Table showing the number of significant overlapping pathways between the significant pathways. Each row refers to a separate disease (as indicated in the first column). Each disease is tested against two datasets depicted in the second and third column. The overlapping figures refer to the pathway overlaps obtained from running SCC (column 4) and GSEA (column 5).

Disease	Dataset 1	Dataset 2	SCC	GSEA
Leukemia	Golub	Armstrong	20	0
ALL Subtype	Ross	Yeoh	10	6
DMD	Haslett	Pescatori	7	10
Lung	Bhattacharjee	Garber	9	0

This high level of overlap between different datasets means that our technique is able to obtain more consistent results across datasets of the same disease. Such consistency may indicate that our algorithm is able to obtain results that are more significant because of our technique and granularity of the usage of biological data within our analysis.

We obtain a low result overlap from GSEA possibly because the pathways from PathwayApi are very large and GSEA relies on a large portion of a pathway exhibiting a correlated change. Hence when only a subset of a pathway demonstrates differential expression, GSEA may be unable to pick this up. We verified this hypothesis by feeding into GSEA subnetworks that we found from our algorithm into the leukemia datasets. Indeed, the overlap percentage improved. The following subnetworks were found to be significant and overlapping.

Table 7.3: Overlapping subnetworks when running our subnetworks on GSEA

Overlapping Subnetworks
leukemia_Glutathione metabolism_GPX1
leukemia_ERK/MAPK Signaling_GRB2
leukemia_Glutathione metabolism_GPX1
leukemia_Oxidative Stress_FOS
leukemia_Focal Adhesion_MAP2K2

7.3 Significant Genes Overlap

From the two lists of significant subnetworks for each disease, we extract two lists of genes that occur in the two corresponding lists of subnetworks and find out the percentage overlap of these genes that are present. This percentage is defined as the number of overlapped genes divided by the number of genes in the smaller list (We term the number of genes in the smaller list as γ).

We carry out the t-test on our datasets by calculating a p-value for each gene and only selecting the genes which has a p-value of less than 0.05. We next take only the top γ genes that are significant. Following which we evaluate the percentage overlap between the two lists of top γ genes from each dataset.

For GSEA, we obtain the significant set of genes by first selecting the leading edge set of genes from the well expressed pathways for each dataset. We obtain two such lists for each dataset and calculate the percentage overlap between these two lists (The percentage overlap is simply defined as $\frac{\gamma}{\delta}$ where γ is the number of overlapping genes between the two lists and δ is the size of the smaller gene list).

The results, shown in Table 7.4, Table 7.5 and Table 7.6 show that the gene overlap obtained from GSEA, t-test and SAM are consistently and significantly lower (2.38% to 28.90% for GSEA, 49.60% to 73.01% for t-test, 49.96% to 81.25% for SAM) as compared to that of our technique (51.18% to 93.01%).

In Table 7.5 and Table 7.6, there are two columns with the same labels. Specifically in Table 7.5 there are two headers with the same column "t-test" and in Table 7.6 are two columns with the same column label "SAM". Notice however that the number of genes being compared are different. For example in Table 7.5, for the first disease leukemia, we compare the gene overlap in the first 1239 genes in the first column for the t-test and 84 genes in the second column for the t-test. We obtain the comparison figure of 1239 genes by counting the number of genes which are significant by the t-test (p-value <

0.05). The other comparison figure of 84 genes is obtained by counting the total number of genes present within our subnetworks obtained for the leukemia datasets.

We point out that the gene overlap for SAM is better in some instances. This is probably due to the stringent criteria that we have for some of the parameters. For instance, we select the top $n\%$ of genes that appear in $m\%$ of patients of a phenotype. We have used the values of 10 for n and 50 for m. If these values are relaxed, it would probably result in more genes being selected and provide a better overlap value.

Table 7.4: Table showing the number and percentage of significant overlapping genes. γ refers to the number of genes compared against and is the number of unique genes within all the significant connected components of the disease datasets. The gene overlap refers to the percentage gene overlap between the two datasets of a disease for SCC (column 3) and GSEA (column 4).

		SCC	GSEA
Leukemia	Num Genes	γ =84	84
	Genes overlap	91.30%	2.38%
ALL Subtype	Num Genes	γ =75	75
	Genes overlap	93.01%	4.0%
DMD	Num Genes	γ =45	45
	Genes overlap	69.23%	28.9%
Lung	Num Genes	γ =65	65
	Genes overlap	51.18%	4.0%

Table 7.5: Table showing the number and percentage of significant overlapping genes. γ refers to the number of genes compared against and is the number of unique genes within all the significant connected components of the disease datasets. The gene overlap refers to the percentage gene overlap between the two datasets of a disease for SCC (column 3) and t-test (column 4).

		SCC	t-test	t-test
Leukemia	Num Genes	δ =84	1239	84
	Genes overlap	91.30%	73.01%	14.29%
ALL Subtype	Num Genes	δ =75	1072	75
	Genes overlap	93.01%	60.20%	57.33%
DMD	Num Genes	δ =45	1319	45
	Genes overlap	69.23%	49.60%	20.00%
Lung	Num Genes	δ =65	2091	65
	Genes overlap	51.18%	65.61	26.16%

Table 7.6: Table showing the number and percentage of significant overlapping genes. γ refers to the number of genes compared against and is the number of unique genes within all the significant connected components of the disease datasets. The gene overlap refers to the percentage gene overlap between the two datasets of a disease for SSP (column 3) and SAM (column 4).

		SCC	SAM	SAM
Leukemia	Num Genes	δ =84	1305	84
	Genes overlap	91.30%	49.96%	22.62%
ALL Subtype	Num Genes	δ =75	464	75
	Genes overlap	93.01%	81.25%	49.33%
DMD	Num Genes	δ =45	126	45
	Genes overlap	69.23%	76.98%	42.22%
Lung	Num Genes	δ =65	966	65
	Genes overlap	51.18%	65.61	24.62%

7.4 Size of subnetworks obtained from t-test

This section shows that the size of the subnetworks we obtain using our algorithm is significantly larger than those we would obtain from the t-test algorithm. Naturally the larger the components are, the more biological inferences can be drawn from the microarray data and the more significant they are likely to be. We first obtain a ranked gene list for each dataset using the t-test algorithm. We therefore obtain two ranked gene lists for each disease i. Assuming once again that the total number of genes present within the significant subnetworks for a disease i is δ_i, we extract the top δ_i genes common to both ranked gene lists of each disease i. Lastly, we calculate the size of the subnetworks formed by this top δ_i genes. The results in Table 7.7 show that the subnetworks obtained are smaller in size ($\leq$ 5 genes) and hence less interesting and significant.

7.5 Validity of Subpathway Genes

To check the validity of the connected components selected, we compare the genes are present within each connected component with those deemed significant by the t-test. A high percentage would mean that we are able to capture connected components which are highly consistent to established methods such as the t-test. The tables below, Table

Table 7.7: Table comparing the size of the subnetworks obtained from the t-test and from SCC. The first column shows the disease that is being considered and the second column shows the number of genes used to create the subnetworks. The third column (which comprises additionally of 4 subcolumns) depicts the number of genes present within each subnetwork for the t-test. Similarly the fourth column depicts the number of genes present within each subnetwork for SCC. So for instance in the leukemia dataset, we have 8 subnetworks with size 2 genes, 1 subnetwork with size 3 genes for the t-test. For SCC, we have 2 subnetworks with size 5 genes, 3 subnetworks with size 6 genes, 2 subnetworks with size 7 genes and 1 subnetwork with a size of ≥ 8 genes

Disease	γ	Num genes (t-test)				Num genes (SCC)			
		2	3	4	5	5	6	7	≥ 8
Leukemia	84	8	1	0	0	2	3	2	1
Subtype	75	5	1	1	1	1	0	1	6
DMD	45	3	1	0	0	1	0	0	5
Lung	65	3	2	1	0	5	3	0	1

Table 7.8: Table depicting the percentage of genes from connected components which are also significant for the t-test. The first column depicts the name of the subnetwork considered. The second column depicts the percentage of genes from that subnetwork which are also deemed significant for the t-test. (Leukemia datasets (Armstrong et al., 2002; Golub et al., 1999))

Component name	Percentage
leukemia_B Cell_VAV1	81.82%
leukemia_Purine metabolism_NP	83.33%
leukemia_Phosphatidylinositol signaling_PLCG2	100.00%
leukemia_Regulation of actin cytoskeleton_RAC1	57.14%
leukemia_Proteasome Degradation_UBC	100.00%
leukemia_Regulation of Actin Cytoskeleton_RAC1	57.14%
leukemia_B Cell_NFKB1	80.00%
leukemia_Regulation of actin cytoskeleton_CSK	75.00%
leukemia_B Cell Receptor Signaling_POU2F2	75.00%
leukemia_IL6 Signaling_IL8	75.00%
leukemia_Focal Adhesion_ACTB	100.00%

7.8 to Table 7.11 show the different components found significant within their respective disease sets. The corresponding percentage depicts the percentage of genes present within the connected component which is also significant by the t-test (taken with a p-value threshold of 0.05). We can observe from the tables that the bulk of the components have a high consistency percentage, falling between 70% to 100%.

Table 7.9: Table depicting the percentage of genes from connected components which are also significant for the t-test. The first column depicts the name of the subnetwork considered. The second column depicts the percentage of genes from that subnetwork which are also deemed significant for the t-test. (Leukemia Subtype datasets (Ross et al., 2004; Yeoh et al., 2002))

Component name	Percentage
MLLBCR_Fatty acid metabolism_ACAA1	28.57%
MLLBCR_Valine, leucine and isoleucine degradation_HSD17B10	40.00%
MLLBCR_B Cell_BLNK	72.73%
MLLBCR_Valine, leucine and isoleucine degradation_HSD17B10	33.33%
MLLBCR_B cell receptor signaling pathway_BLNK	72.73%
MLLBCR_Acute myeloid leukemia_FLT3	44.44%
BCR_Chronic myeloid leukemia_ABL1	75.00%
BCR_Fc Epsilon RI Signaling_PIK3C2B	70.00%
BCR_T Cell Receptor Signaling Pathway_RASA1	44.44%

Table 7.10: Table depicting the percentage of genes from connected components which are also significant for the t-test. The first column depicts the name of the subnetwork considered. The second column depicts the percentage of genes from that subnetwork which are also deemed significant for the t-test. (DMD datasets (Haslett et al., 2002; Pescatori et al., 2007))

Component name	Percentage
DMD_Tight junction_RHOA	87.50%
DMD_Integrin Signaling_TTN	75.00%
DMD_ECM-receptor interaction_SDC3	88.89%
DMD_Tight junction_RHOA	85.71%
DMD_Leukocyte transendothelial migration_ACTB	83.33%
DMD_Actin Cytoskeleton Signaling_MYL9	78.57%
DMD_Calcium signaling pathway_CALM1	80.00%

Table 7.11: Table depicting the percentage of genes from connected components which are also significant for the t-test. The first column depicts the name of the subnetwork considered. The second column depicts the percentage of genes from that subnetwork which are also deemed significant for the t-test. (Lung datasets (Bhattacharjee et al., 2001; Garber et al., 2001))

Component name	Percentage
SCC_Notch signaling pathway_NOTCH3	100.00%
SCC_ECM-receptor interaction_SDC1	69.23%
SCC_Adherens junction_CTNNB1	100.00%
SCC_Tyrosine metabolism_ADH1B	100.00%
SCC_Phenylalanine metabolism_ALDH3B1	100.00%
SCC_Tryptophan metabolism_WBSCR22	80.00%
SCC_Natural killer cell mediated cytotoxicity_TNFSF10	60.00%
SCC_Insulin Recpetor Signaling_AKT3	100.00%
SCC_Glycogen Metabolism_PYGM	60.00%

7.6 Histograms of Connected Components

Finally, we show the histogram of the distributions of the different connected components. The y-axis of the histograms depict the number of patients while the x-axis depicts the percentage of genes that are high in that segment. We will first show a few sample histograms (Figure 7.1) where we are able to obtain nice histograms which clearly show the difference between the phenotypes of the various diseases. The rest of the histograms are included in the Appendix for the interested reader's reference.

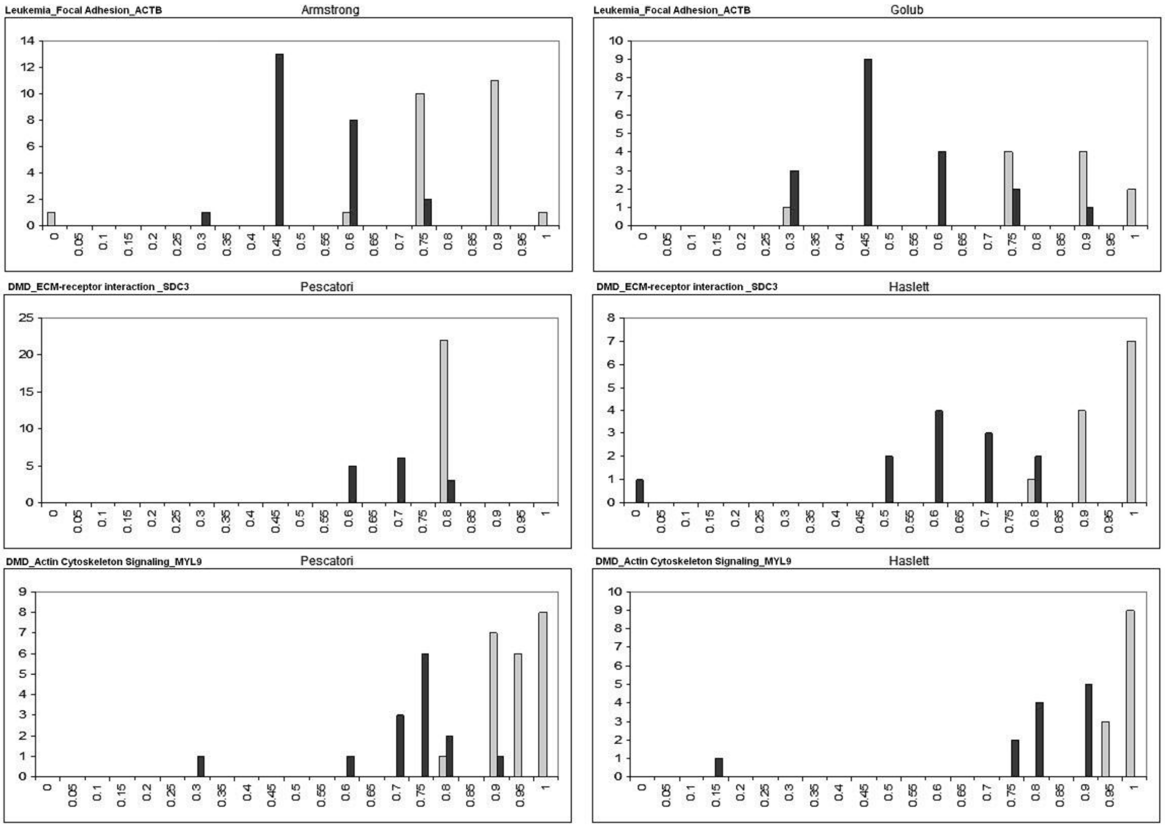


Figure 7.1: Sample histograms depicting the scores obtained from the connected components. The top two graphs, refer to the 097_304_1_Golub_AML connected component. The left graph shows the scores obtained from the Armstrong microarray experiment and the right from the Golub microarray experiment. The x-axis refers to the score assigned and the y-axis refers to the number of patients with that score. The dark colored bars refers to the patients suffering from AML while the light colored bars refers to patients suffering from ALL. For the DMD graphs, dark colored bars refers to normal control patients while light colored bars refer to patients suffering from DMD.

7.7 Biological Relevance of Sample Subnetworks

Two small sample subnetworks are chosen here to show the biological significance of the results obtained. The first which we describe below in Figure 7.2 is generated from the leukemia dataset. The genes within this subnetwork are very substantially supported by literature with respect to their role in leukemia. For instance, the gene RAC (which regulate a diverse array of cellular events) is referenced in (Krishna and LeDoux, 2006; Wang et al., 2009) as having an effect on leukemia. Other genes within the network are Rhoa (regulates the actin cytoskeleton in formation of stress fibers) in (Booden et al., 2002; Kristelly et al., 2004), Vav1 (plays a major role in development and activation of T-cell and B-cell blood cells) in (Katzav, 2007) and IQGAP (regulates cell adhesion, morphology and motility) in (Juliana et al., 2006).

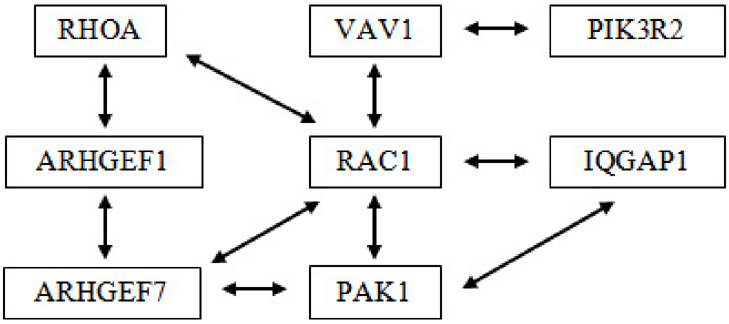


Figure 7.2: A sample pathway component from leukemia dataset (Armstrong et al., 2002; Golub et al., 1999).

The next subnetwork shown in Figure 7.3 is generated from the DMD disease datasets, and is taken from the Apoptosis pathway. Results from our algorithm indicated that the genes groups MYL and MYH are significantly differentiately expressed between the DMD patients and the normal patients. MYH (myosin, heavy chain) and MYL (myosin, light chain) are known to be major gene groups involved in release of mechanical energy allowing muscles to contract. These genes are heavily quoted in literature with regard to their involvement in the disease DMD: MYH3 and MYH8 (Haslett et al., 2002), MYH6 (Balagopal et al., 2006), MYH7 (Baker et al., 2006), MYL1, MYL2, MYL3, MYL4, MYL5, MYL6 and MYL9 (Balagopal et al., 2006). In addition, the gene titin which was identified. Titin is a gene which encodes a large protein of the spinal skeletal muscles and its mutation is widely found to occur in various types of muscular dystropy (Garvey et al., 2002; Gerull et al., 2002; Hackman et al., 2002; Itoh-Satoh et al., 2002).

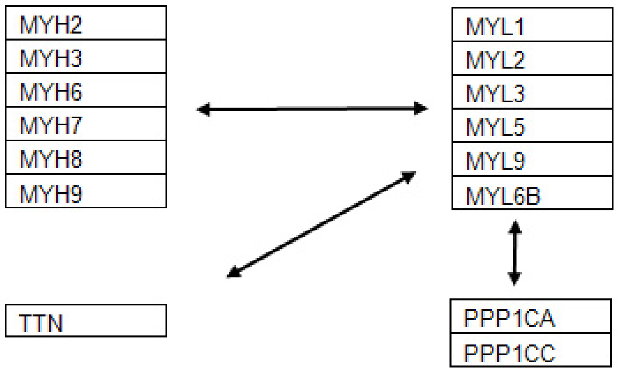


Figure 7.3: A sample pathway component from DMD dataset (Haslett et al., 2002; Pescatori et al., 2007).

CHAPTER 8

Conclusions

8.1 Conclusions

Microarray experiments are crucial because they measure the behaviour of individual genes with respect to diseases or treatments. Results from these experiments are heavily scrutinised to obtain biological insights into the occurrence of diseases or the effectiveness of certain types of treatments.

In order to provide more indepth analysis to experiments, contemporary algorithms have incorporated biological information into their analysis so that the analysis can be more descriptive and hopefully useful to the researchers. Our techniques have taken this approach one step further. Firstly, we no longer consider prior biological knowledge as a separate aspect of microarray analysis. Rather, we take into account the integrity of the biological information that is being provided into the algorithm for analysis. Secondly, our algorithm uses both the gene-gene interaction information and pathway information in our analysis. Because of these two enhancements, we are able to generate subnetworks in real-time according to the responses of the microarray experiments. These contributions helps us avoid some of the potential caveats present within microarray experiments.

Our first rigorous step (as explained in Chapter 3) of demonstrating qualitatively the huge inconsistencies within current databases is already a non-trivial contribution. This step is crucial because it demonstrates the lack of coherency (qualitatively results seen in Chapter 5) between microarray analysis if different databases are to be used. Hence we created our own database by combining the information from several other well established biological databases. This ensures that the prior biological information going into the algorithm is generally agreed upon among the databases and hence we can expect greater consistency in the results.

Referencing our chapter about algorithmic design (chapter 6), we again distinct ourselves from the algorithms such as GSEA and NEA (Sivachenko et al., 2007) by using both gene-gene relationships and pathway information. The paper by (He and Zhang, 2006) pointed out that such hubs in gene-gene interaction networks correspond to essential genes because they have a higher probability of involvement in essential gene-gene interactions. Therefore the usage of gene-gene relationships provides us with information on how genes affect one another and pathway information organises such relational information within their relevant biological processes. In contrast, GSEA only uses gene sets (akin to our pathway information), hence losing crucial information on the individual processes occurring between genes. In general, not all genes in a gene set are connected to each other (and rightfully so), GSEA might rank gene sets as significant although the number interactions between the proteins within each gene set is weak. In addition, other processing steps are required to decipher how genes within a gene set interact with one another. In addition, GSEA might also rank a gene set as insignificant if the gene set is very big and most of the genes are not differentially expressed, even when there is a path within the large set that is differentially expressed. This happens when a portion of the gene set is significantly differentially expressed, but the rest of the genes are not. Thus even though that particular portion of the gene set is biologically significant, this will not be indicated out by GSEA.

We are certainly not the first to integrate gene-expression data with gene-gene relationships. GNEA (Liu et al., 2007) is one such example. GNEA uses a global protein-

protein interaction network, finds a subnetwork within this global interaction network and compares this subnetwork within gene sets to find out the significant gene sets.

However in creating a single global biological interaction network, it makes the biological assumption that the local behaviour of proteins can be translated in a similar fashion globally and that gene expression levels are in a tight correspondence to protein levels (which is not generally true). A similar issue is raised in (Sivachenko et al., 2007) where the authors argued that proteins which are very well connected have an extremely high chance of obtaining a low p-value and being ranked as significant. Because of the high connectivity of such proteins, they are liable to be involved in various disjoint biological processes, leading to the error of combining independent subnetworks through these proteins. To prevent such scenarios, we instead implemented our algorithm via identifying localised gene-gene subnetworks within pathways.

In addition, we show in Chapter 7 that our technique generates significant subnetworks and genes that are more consistent across datasets compared to the other popular methods available (GSEA, t-test and SAM). The large size of subnetworks which we generate indicates that they are generally more biologically significant (less likely to be spurious). To validate our results, we show that most of our genes from the generated subnetworks have also been considered significant by the t-test. In addition, we have chosen two sample subnetworks and validated them with references from biological literature. This shows that our algorithm is capable of generating descriptive biologically conclusions.

Our final contribution lies in our ability to create connected components (of known pathways) in real time based on microarray data. This allows us to obtain connected components according to the microarray data. GSEA uses fixed gene sets and determines if these gene sets are significant or not. GNEA first finds subnetworks from a global PPI (protein protein interaction) network. The GNEA next carries out a statistical test on fixed gene sets, to determine the individual gene set has a significant proportion of genes within this subnetwork. Hence these techniques assume that a gene set is significant only if it has a substantial proportion of its genes significant. This assumption might not be valid because there are instances where only part of a gene set becomes significant, and it would probably go noticed if the rest of the genes are unaffected. Our ability to create connected components based on the microarray data of the phenotypes ensures that we have sufficient granularity to capture portions of pathways or gene sets are affected.

With reference to Chapter 1, these contributions enable us to successfully make consistent biological inferences from microarray experiments and prior biological information.

8.2 Future Work

Biologically descriptive analysis of microarrays is currently an active of research, providing many opportunities for novel ideas. While this thesis has contributed some new concepts in this area, it contains its own shortfalls and there remains many interesting issues that necessitate further research.

8.2.1 Building an automated tool

To demonstrate the validity of our algorithm, we have chosen to use only two datasets of various diseases rather digging in depth into a single disease but across many datasets. Though this has shown that we are able to obtain significant results across two datasets of the same disease, we would have achieved an interesting biological result if we were able to find consistent results of similar nature if we had used numerous datasets of the same disease. If biological conclusions from a single disease are consistently significant over all the other datasets, it is very possible that such a biological conclusion is heavily supported.

To carry out such large-scale analysis, it will require us extending the algorithm towards building a more automated tool. This tool will require us to revise the following components:

1. Microarray input formats The current method of accepting DNA microarray inputs into the algorithm is still very manual. The extension will allow the algorithm to accept the microarray data with its annotation information and automatically link it back to the biological pathway inputs.

2. Biological input formats Only biological data from PathwayAPI is accepted currently. One of the extension will be to extend this to other biological formats like SBML, BioPaX and SGML. This will allow the tool to be more pervasive and not be tied down to any particular biological repository.

3. Statistical processing For statistical processing, we will have to replace the hypothesis test with the ANOVA so as to allow connected components from multiple datasets of the same disease to be tested at the same time.

These will allow us to carry out wide and large scale studies within GEO and other important databases, allowing us to detect significant subnetworks that are consistent within more databases. As these subnetworks are significant across more datasets, they will definitely give a lot more insights for the researcher.

8.2.2 RNA Seq

An extension to use RNA-Seq as the choice of experimental input is possible. However the design of the algorithm will have to cater to the vast amounts of additional data when RNA-Seq is being used and there being a greater level of biological granularity within RNA-Seq experiments. Together with a biological database that supports this level of granularity, the algorithm will be able to increase the amount of descriptive information that the algorithm will be able to provide (without a large change in the fundamental algorithm).

8.2.3 Over-reliance on biological information

The main conceptual shortfall of the algorithm involves our reliance on biological data. As pointed out also in (Sivachenko et al., 2007), there are two scenarios where our solution will find it challenging to find any biologically significant results. One such situation occurs when the biological information available is not sufficiently comprehensive. Naturally when the information is not comprehensive, the connected components created will not be sufficiently large / comprehensive to be significant and hence will be unable to provide proper biological descriptions. The second situation occurs when the biological information provided is inaccurate. In such cases, we will create the wrong components within the pathways and hence draw incorrect conclusions from the results. We understand such a problem and have tried to alleviate such issues by creating our own integrated biological database. However we realise that such issues are inevitable and there is always a probability of it occurring. Naturally in such scenarios, the system must at least be able to differentiate the difference between the situation where “no significant results are found because of a lack of biological a priori information” and “no significant results are found because of a lack of correlating microarray data”.

8.2.4 Pairwise connections within components

In addition, although biological information is being integrated into the analysis, we are utilising gene pairwise connections within components instead of their pairwise relationships (For instance, instead of taking into account the relationship p53 activates MDM2 or MDM2 inhibits p53, we merely use the connection between the two genes, simplifying their relationship to one that p53 is connected to MDM2). This simplification greatly reduces the granularity of our analysis. For instance, if in the scenario where both p53 and MDM2 are activated, we could deduced that the relationship “p53 activates MDM2” is significant and in the scenario where only MDM2 is activated, we could deduce that “MDM2 inhibits p53” is significant. However with our simplification, we can only deduce that the connected between p53 and MDM2 is significant in the former scenario and we can deduce nothing significant in the second scenario.

Finally, although we have used our best efforts to ensure a professional standard in all our experimental studies, there are still some limitations of the project due to its scale. For instance, the biological repository we created came from only three independent pathway repositories. Although the number of pathways combined together is already considered substantial (as most analysis only use a single database), we understand that the more databases we include in our study, the more complete the information we would hold and the more complete the study would be.

8.3 Discussions

8.3.1 Comparison with previous techniques

We are able to perform better than GSEA because we have managed to concentrate on a more focused granularity, finding connected components which are significant and not entire pathways. This gives us an extremely good size for biological analysis because it is large enough to make proper biological inferences, yet small enough such that the possibility of false positives and false negatives is kept as low as possible. This especially goes true in the future as lines between pathways become blurred (deciding which genes should be allowed in which pathways is already a subjective issue). In addition, as more data gets available, pathways tend to increase in size as well and not all genes within it are actually relevant to the analysis. (We have tried running the data on Pathway Express (PathwayExpress, 2009) but unfortunately were unable to obtain any results. This is probably due to the size of the data as Pathway Express works better on smaller datasets.)

8.3.2 Irregularity of microarray experiments

Though microarray experiments have seen vast improvements over the last few years, they are still relatively rudimentary as compared to the exact binary computational systems that we are so used to. For instance, it is a known fact that there is a natural fluctuation in the expression level of genes within a cell. This natural fluctuation may cause the natural ranges of expression of the same gene from different cell phenotypes (eg normal vs disease) to overlap. A sudden rise of the expression level of a gene therefore may not suggest the cell phenotype. To alleviate the impact of such scenarios, our solution was to chose only genes who appear highly expressed in more than 50% of the patients of a phenotype.

8.3.3 Metrics Used

The final version of our algorithm used the t-test to find components which are activated more in one component and less in the other component. Other than the t-test we had tested a numerous host of techniques such as chi-square, gini coefficient, entropy, etc. The other techniques unfortunately could not differentiate components as well as the t-test. For instance, in using the chi-square test, some of the minimal values required for the parameters of the chi-square test are often not fulfilled. Entropy failed as a metric as well because the (absolute) value of the entropy is high when the components behave similarly across phenotypes (a case of total information) as well as when the components behave totally differently across phenotypes (a case of totally no information). Ultimately, we found the t-test to still be the best in capturing the different behaviour of components across different phenotypes.

8.3.4 Size of Subnetwork

In the beginning of the project we had attempted to carry out our pathway analysis on a smaller scale (using only < 5 pathways) using an extremely manual process. Although such a technique would provide us with results of a very high granularity, reproducing such techniques on the many pathways available would not have been a feasible option. In addition, it would run contrary to the technological trend that requires us to process more data within a shorter period of time. After the decision was made to expand the analysis to a technique that is scalable, we managed to test our algorithm in rapidly across many pathways.

8.3.5 Overlap of Components vs Overlap of Genes

We recognise that the percentage overlap of components is consistently not as high as compared to the overlap of genes. We argue that such a phenomena occurs because there is a key group of genes (which belong to a few key components) acting specifically for that phenotype. However this key group of genes (being more influential) occur in various other (non significant) components as well. Hence when spurious genes or false positives (possibly due to batch effect) occur, they may make the non-significant components to be deemed as significant as well. This probably means that the technique is still not refined enough to capture the nuances within the components since it is unable to reject such false positives.

8.3.6 Selection of m and n

We have chosen the value of m and n through empirical means and found 10% of the top genes, 50% of patients as suitable parameters to be used in the experiment. A more methodological way is to assess based on the distribution of gene expression values. Specifically, we can:

1. Optionally do a log transform of the gene expression values.
2. Divide the gene expression values into 100 intervals (buckets).
3. (iii) Plot the frequency distribution of gene expression values into these intervals. It should form a normal-like distribution.
4. Most of the samples (in our experience, 8̃0%) will be near the mean, x̃% (in our experience, 1̃0%) near the left (i.e. low) extreme. x̃% (in our experience, 1̃0%) near the right (i.e. high) extreme. This suggests x% to be the threshold.

It is also possible to consider an optimization approach, first seeding the values of n and m, and then carry out a simple hill climbing process, tweaking the values of n and m and stopping when the score starts to deteriorate (this score can be the number of consistent subneworks obtained). The seeding of n and m with the hill climbing can be repeated several times to ensure that we do not hit into a local maximum. However, this approach may risk overtuning of the n and m parameters.

8.3.7 Extension to other datasets

As mentioned in the previous section, one of the extensions would be to create a large-scale tool which will be able to read in both microarray data and biological data of different formats. This is a non-trivial engineering issue and a possible solution will be to standardize two fixed formats. One format for the microarray data and the other for the biological data. Software connectors will be written to convert both the microarray data as well as the biological data into one of these fixed formats. The algorithm then accepts only these two fixed formats as inputs. This framework will allow other formats to be easily added and make the tool more scalable.

8.3.8 Specificity of the Algorithm

When running the algorithm for the leukemia dataset, we failed initially to take note that the data consisted of different types of ALL leukemia. Although both datasets boasted that the samples were taken from ALL patients, we were unaware that (Armstrong et al., 2002) consisted of patients only from T-cell ALL patients while the dataset from (Golub et al., 1999) consisted of patients from B-cell ALL and T-cell ALL. In using the mixed dataset, we were unable to find a good component intersection between the datasets and the percentage overlap stuck at 30% or lower. However when this was discovered and we removed the B cell patients from the list, the overlap percentage increased to more than 70%. This shows the high level of specificity of the algorithm.

APPENDIX A

Histograms

We show here the histograms from the connected components in Chapter 7.

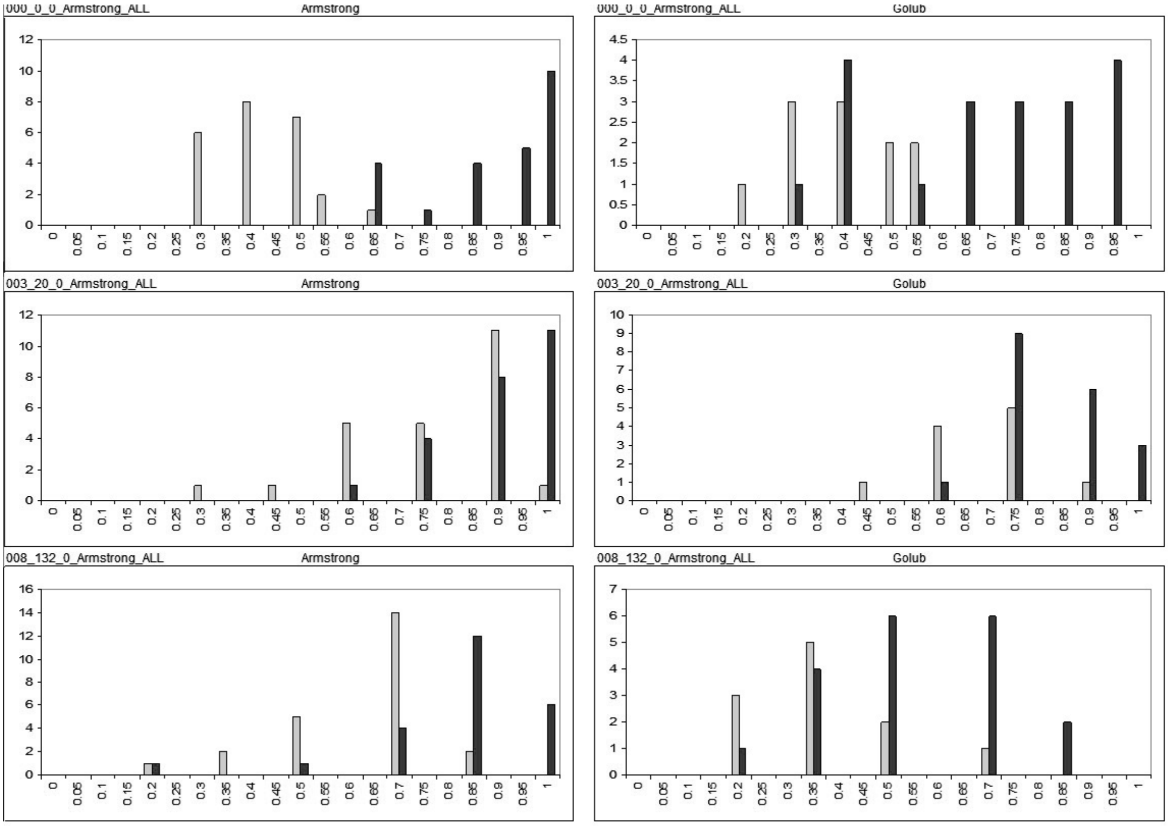


Figure A.1: Histogram of leukemia connected components (a).

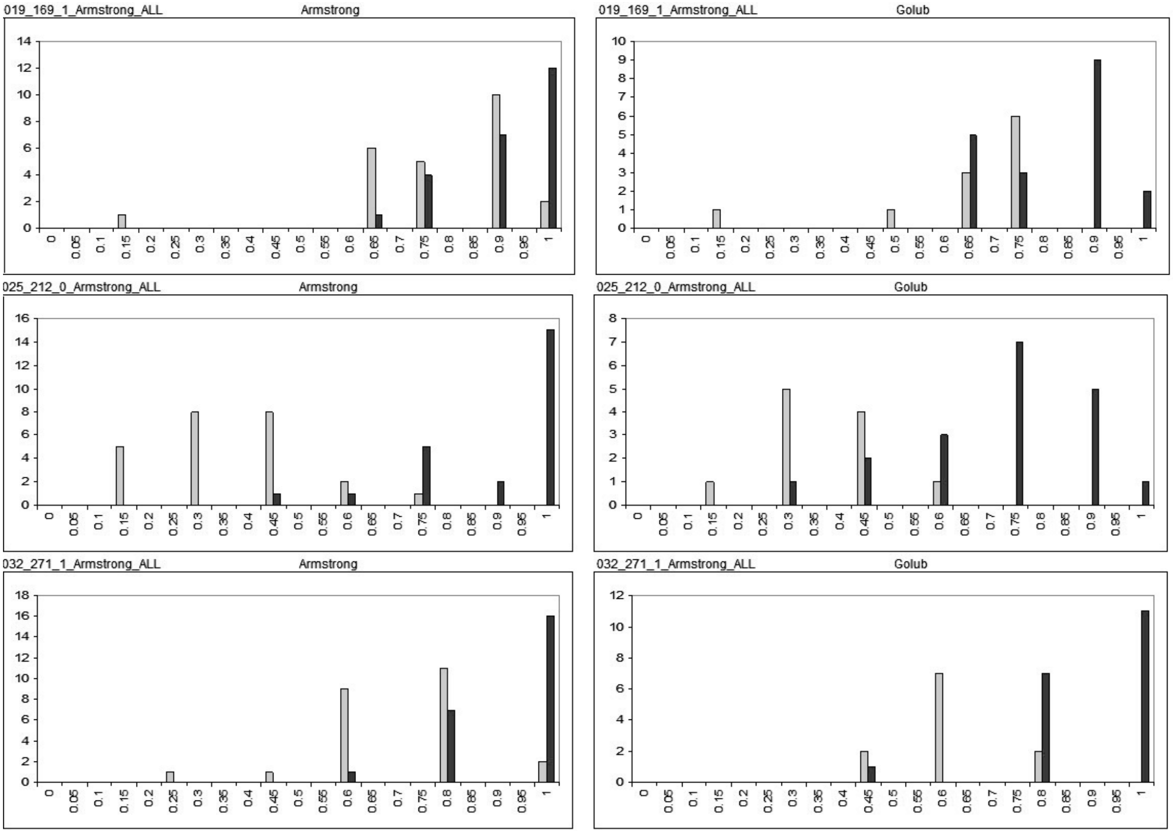


Figure A.2: Histogram of leukemia connected components (b).

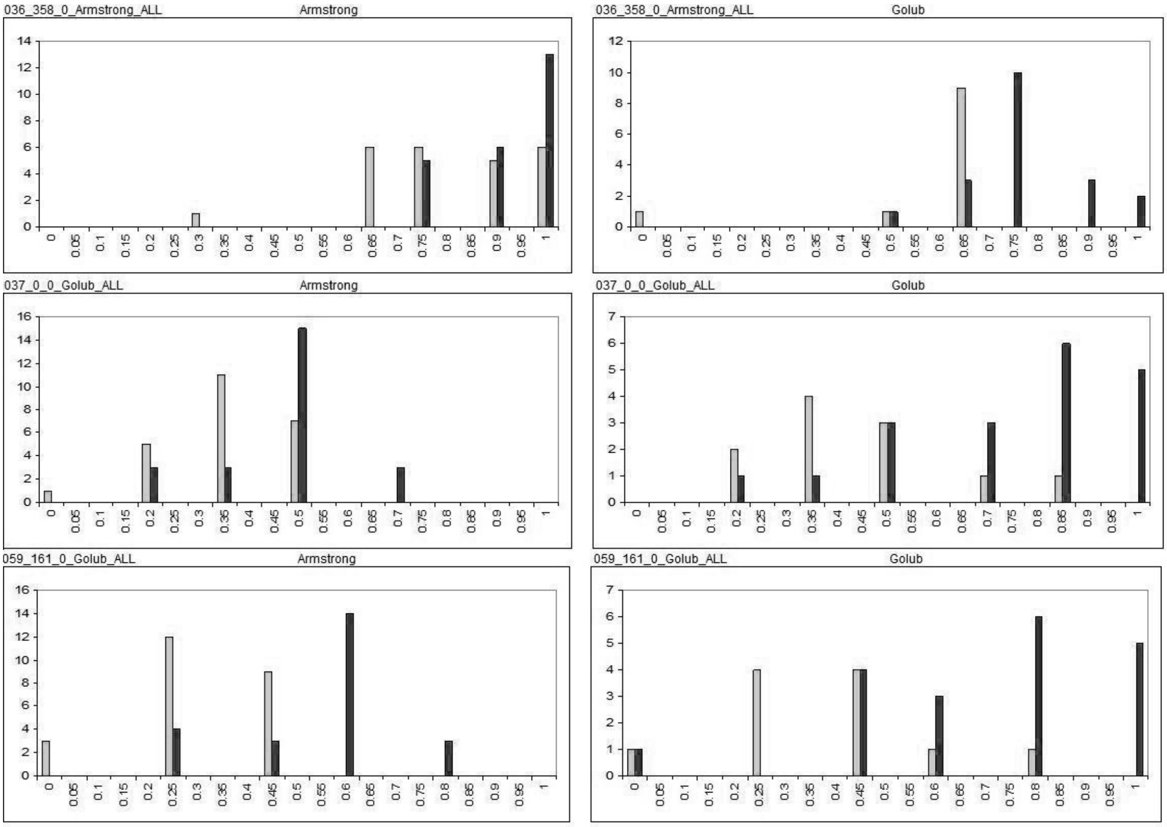


Figure A.3: Histogram of leukemia connected components (c).

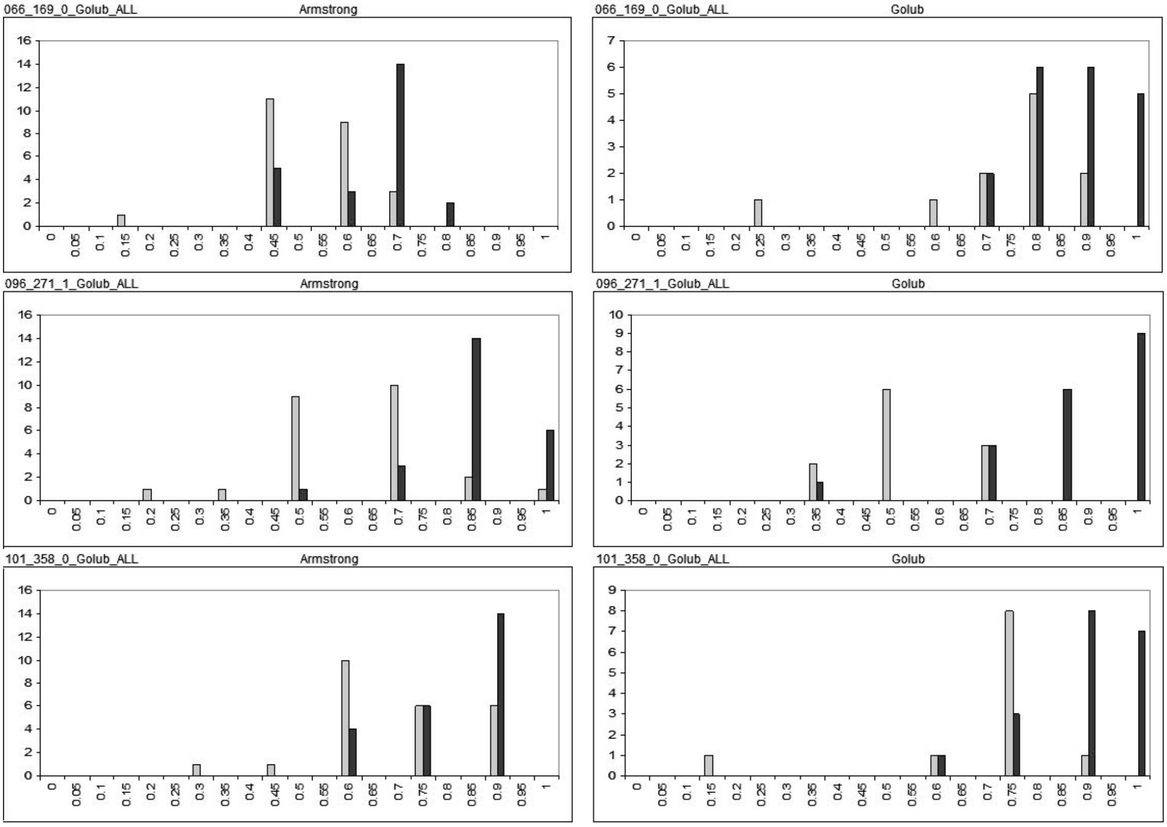


Figure A.4: Histogram of leukemia connected components (d).

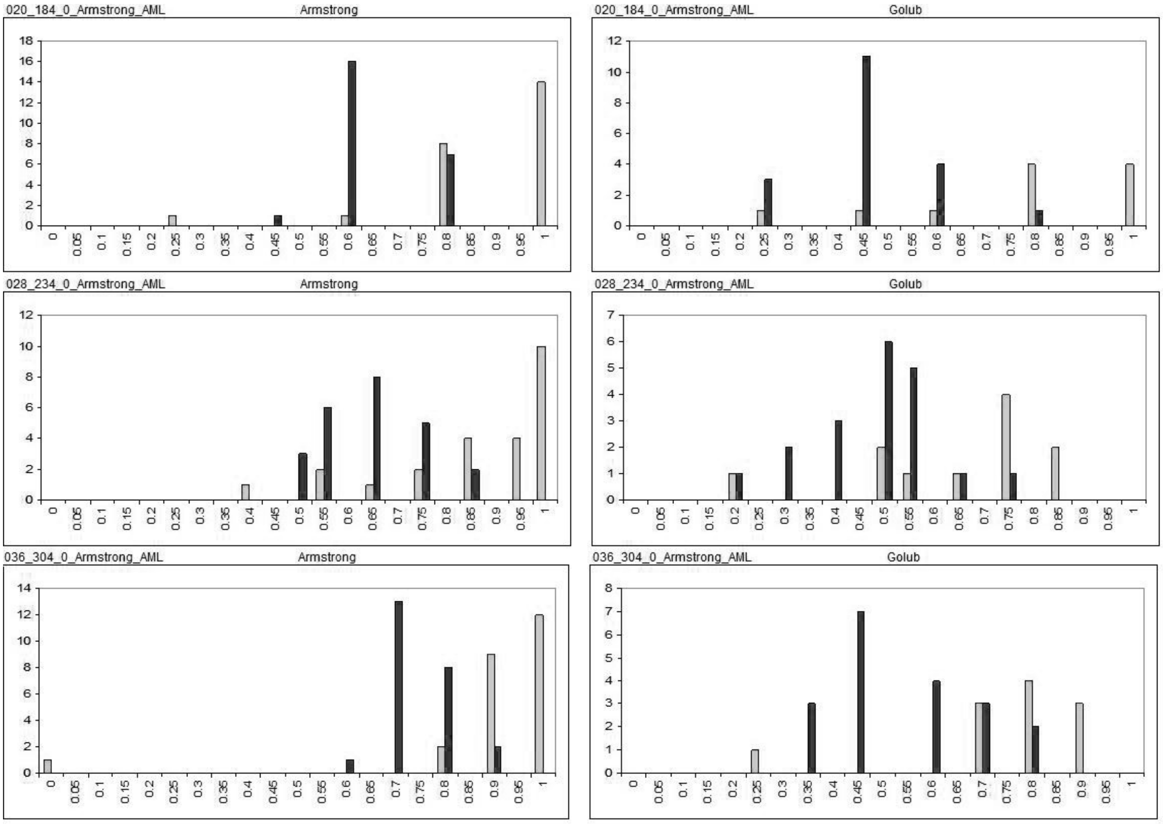


Figure A.5: Histogram of leukemia connected components (e).

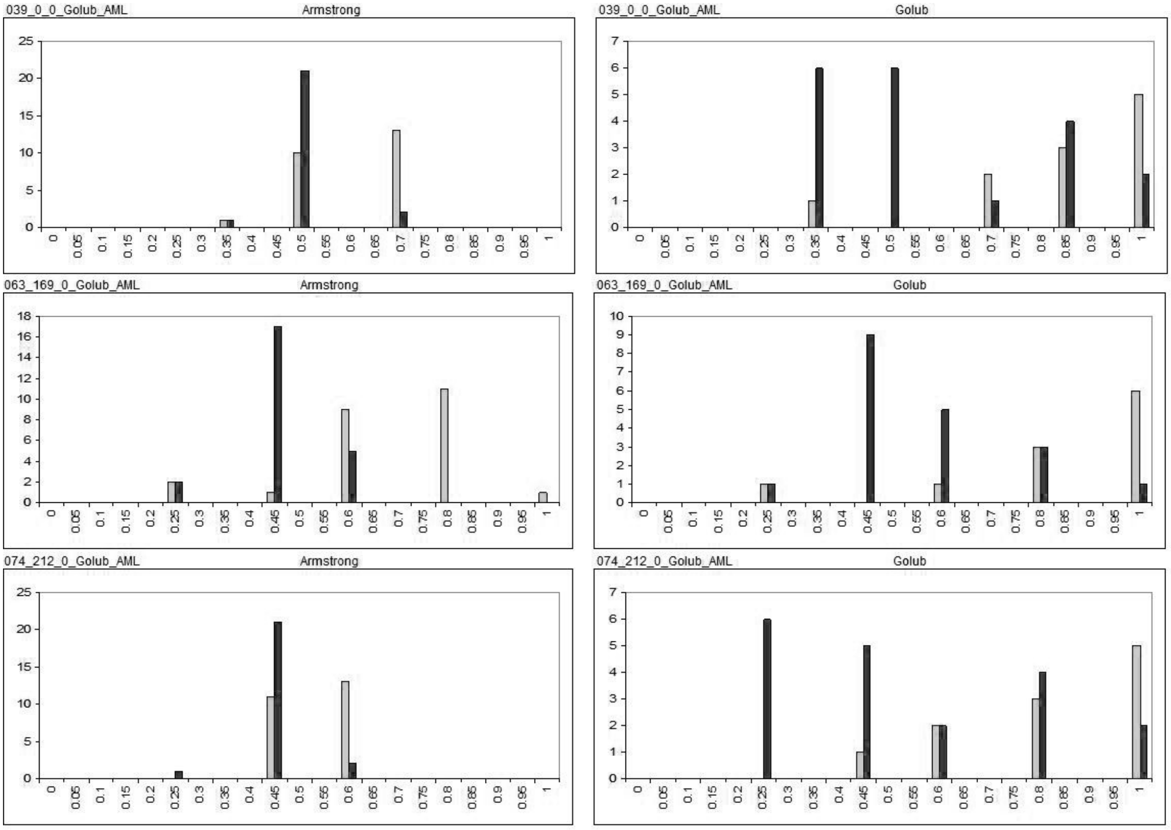


Figure A.6: Histogram of leukemia connected components (f).

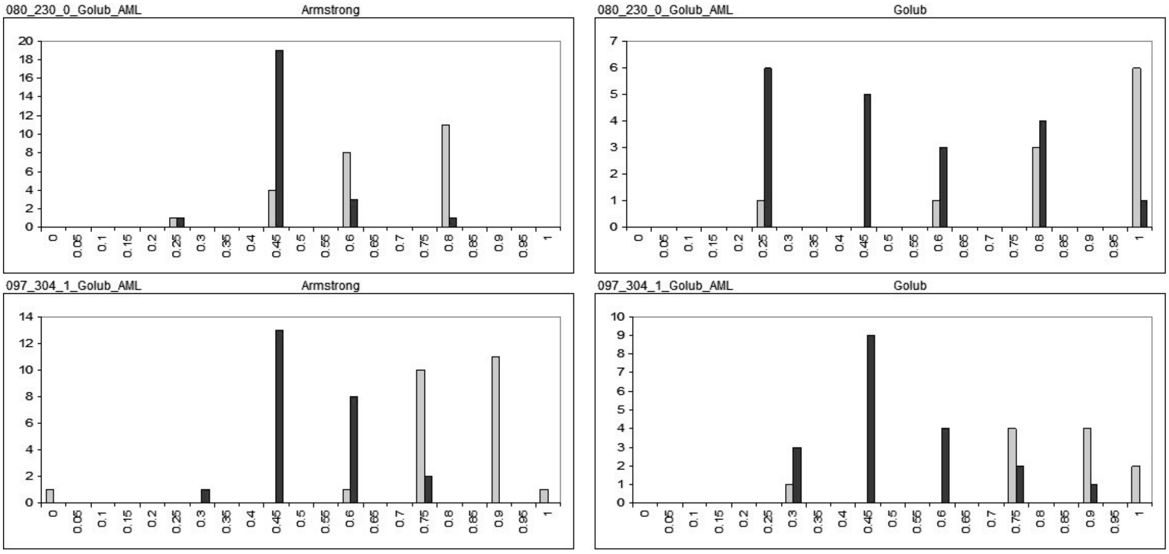


Figure A.7: Histogram of leukemia connected components (g).

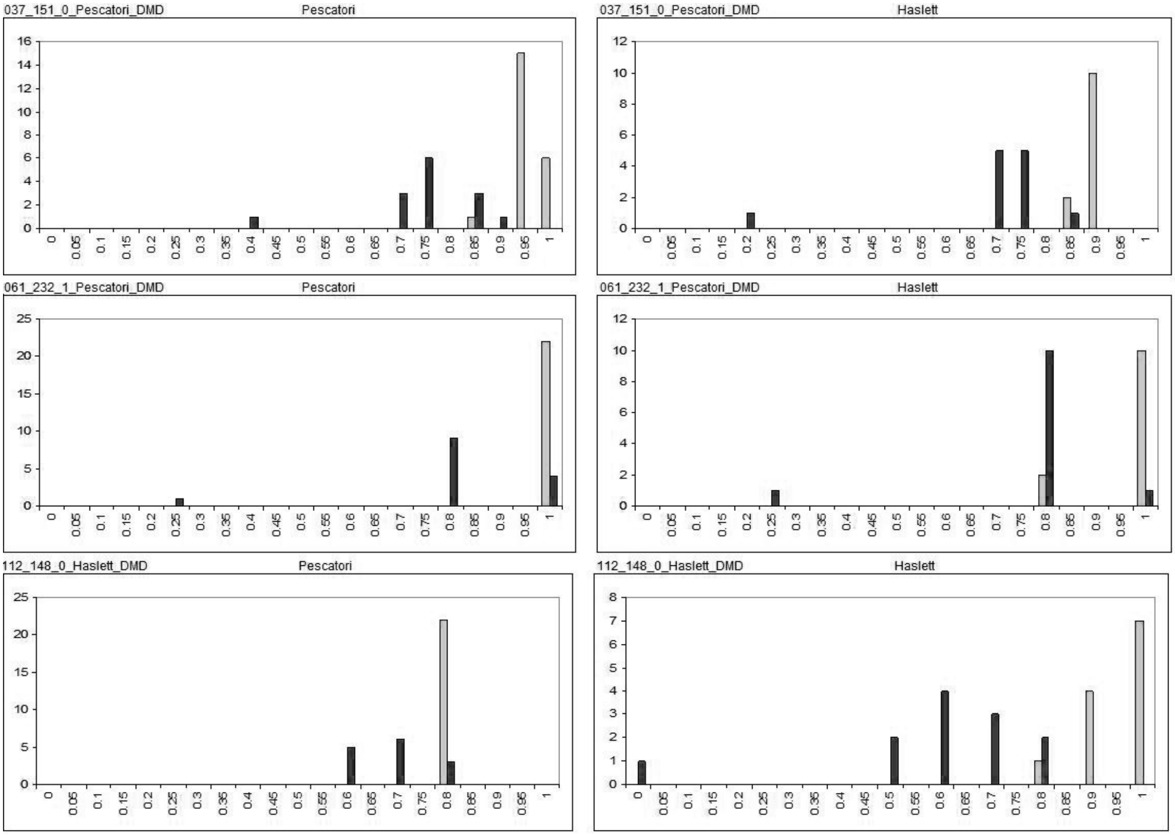


Figure A.8: Histogram of DMD connected components (a).

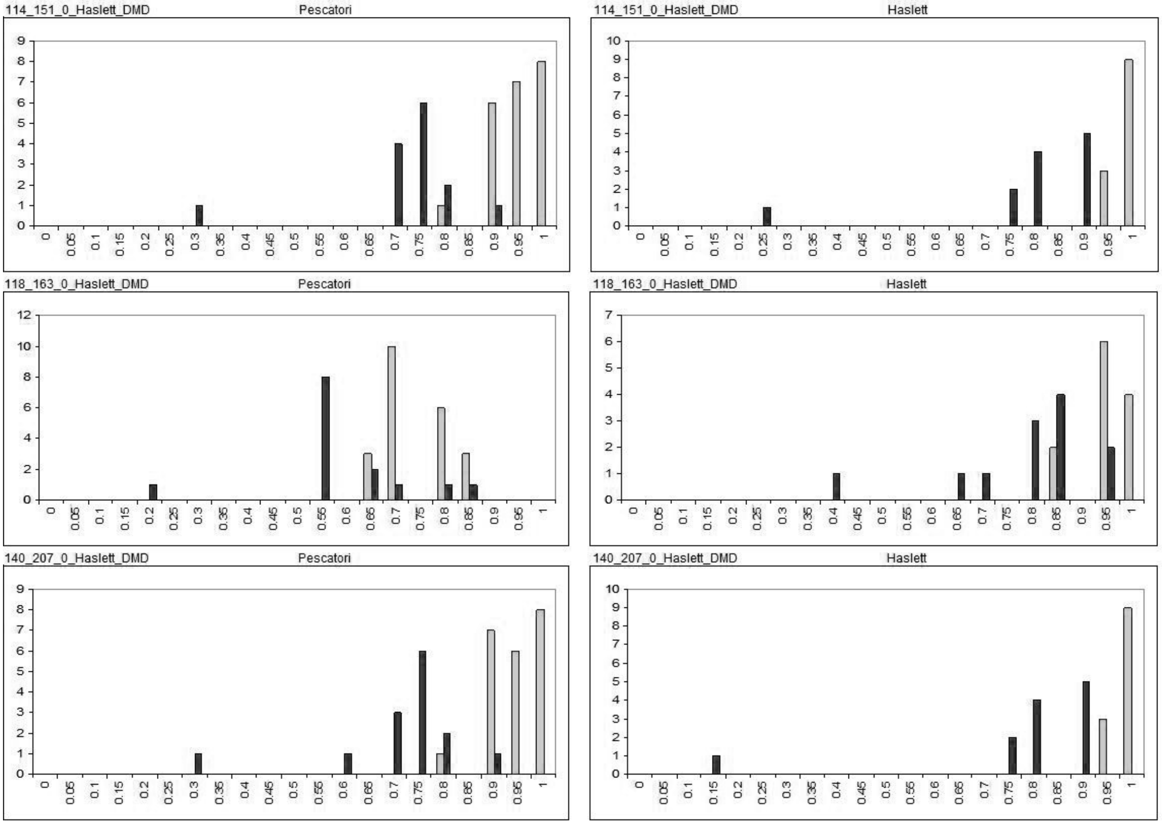


Figure A.9: Histogram of DMD connected components (b).

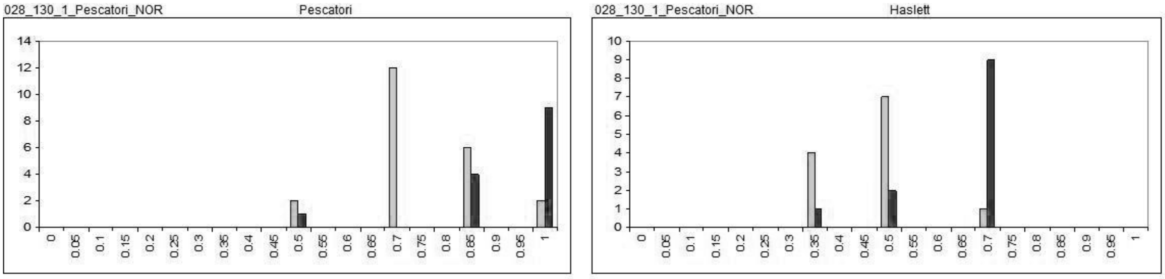


Figure A.10: Histogram of DMD connected components (c).

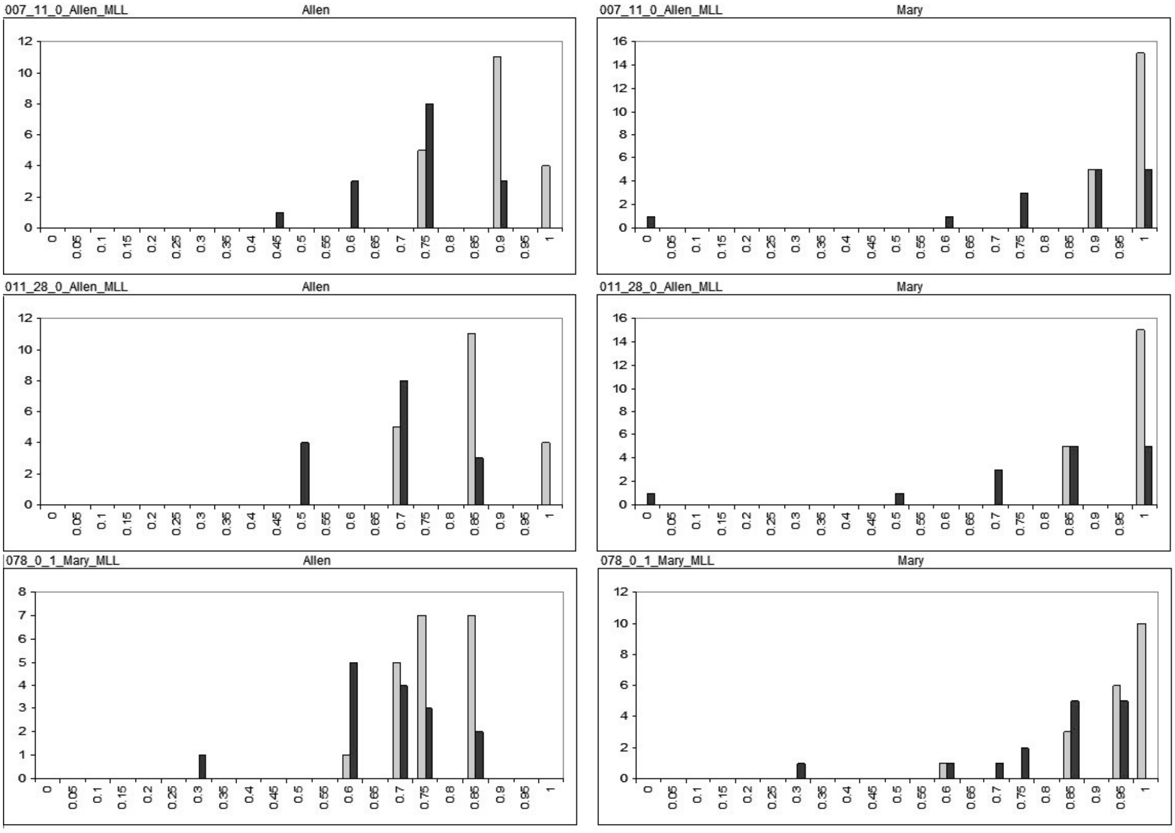


Figure A.11: Histogram of Subtype connected components (a).

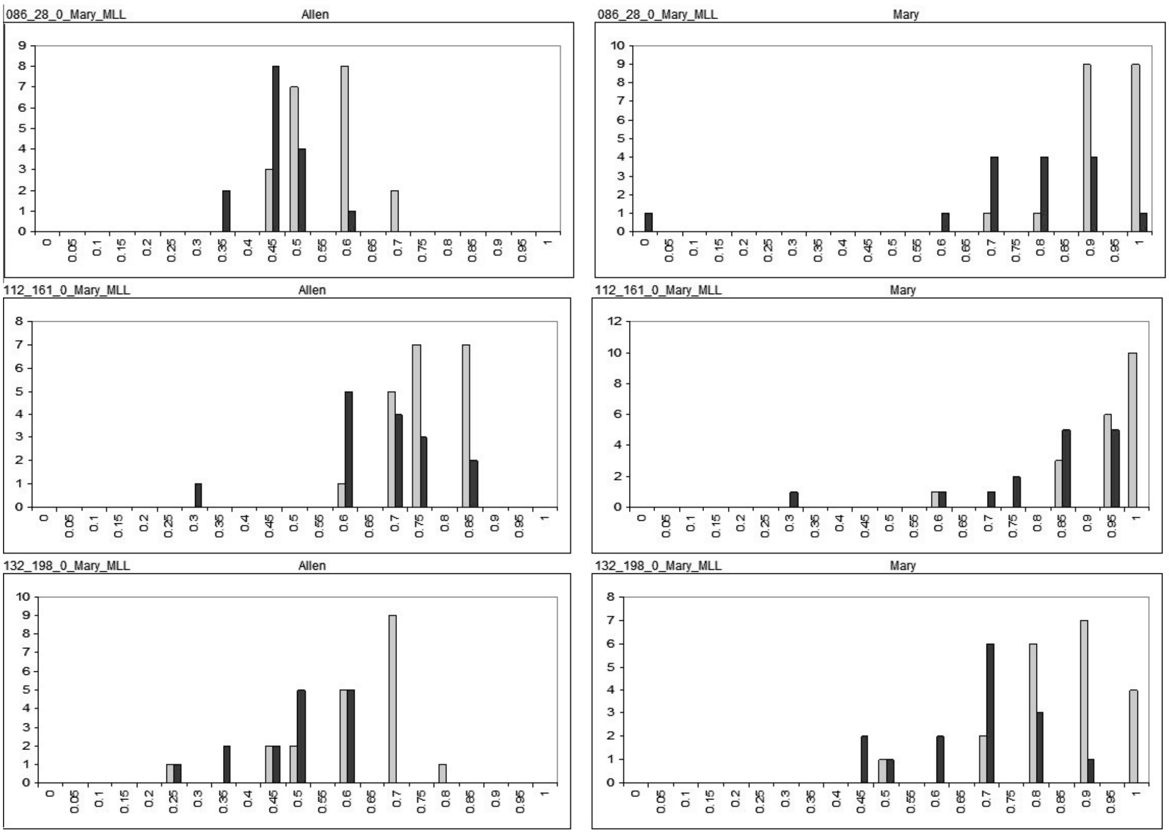


Figure A.12: Histogram of Subtype connected components (b).

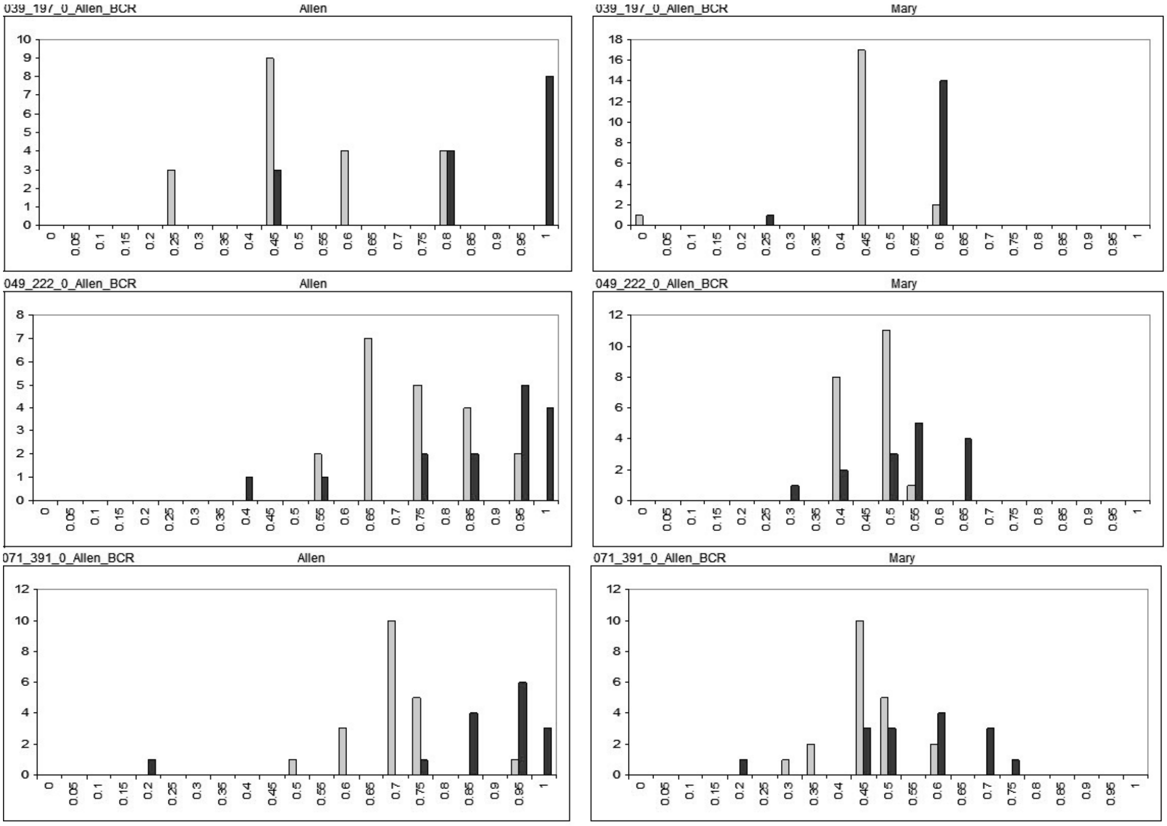


Figure A.13: Histogram of Subtype connected components (c).

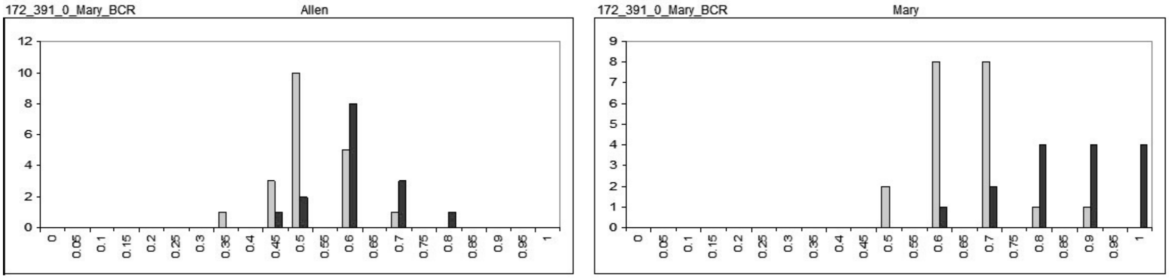


Figure A.14: Histogram of Subtype connected components (d).

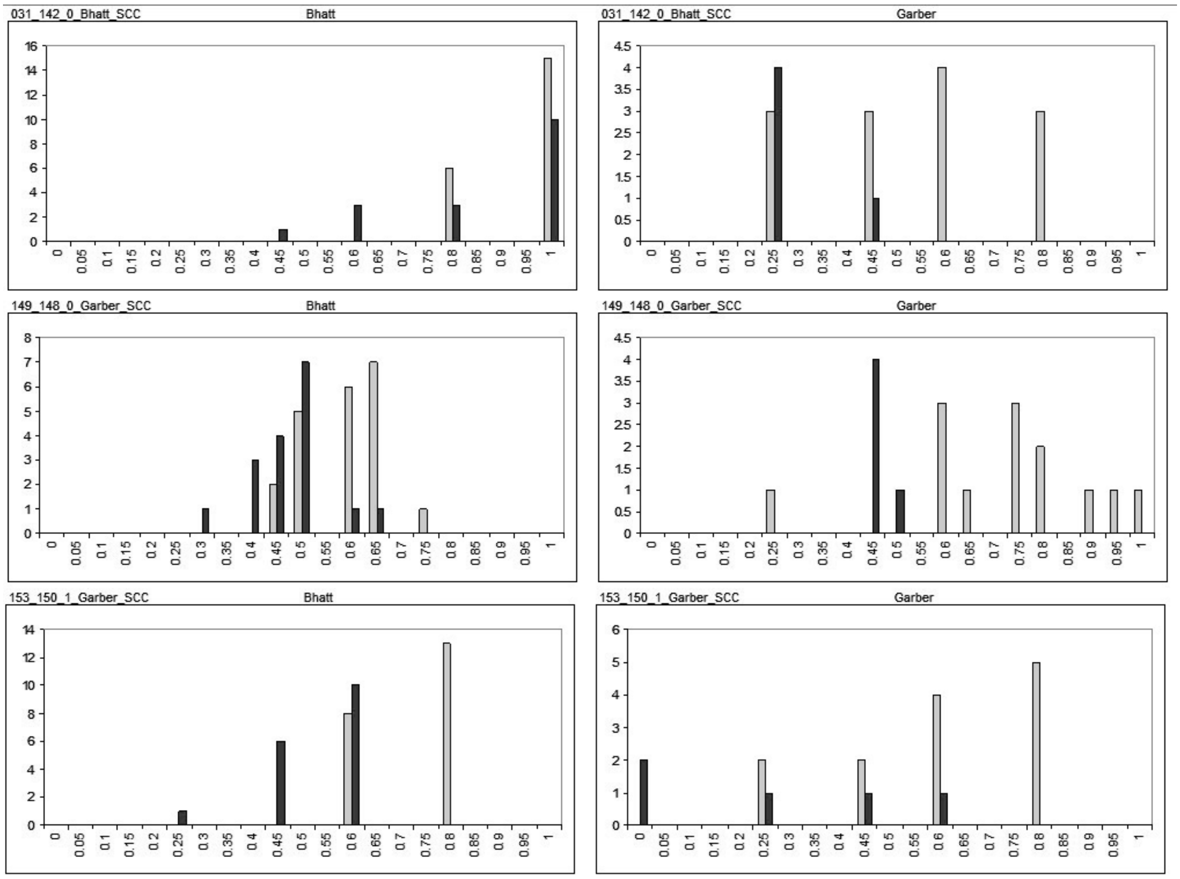


Figure A.15: Histogram of Lung connected components (a).

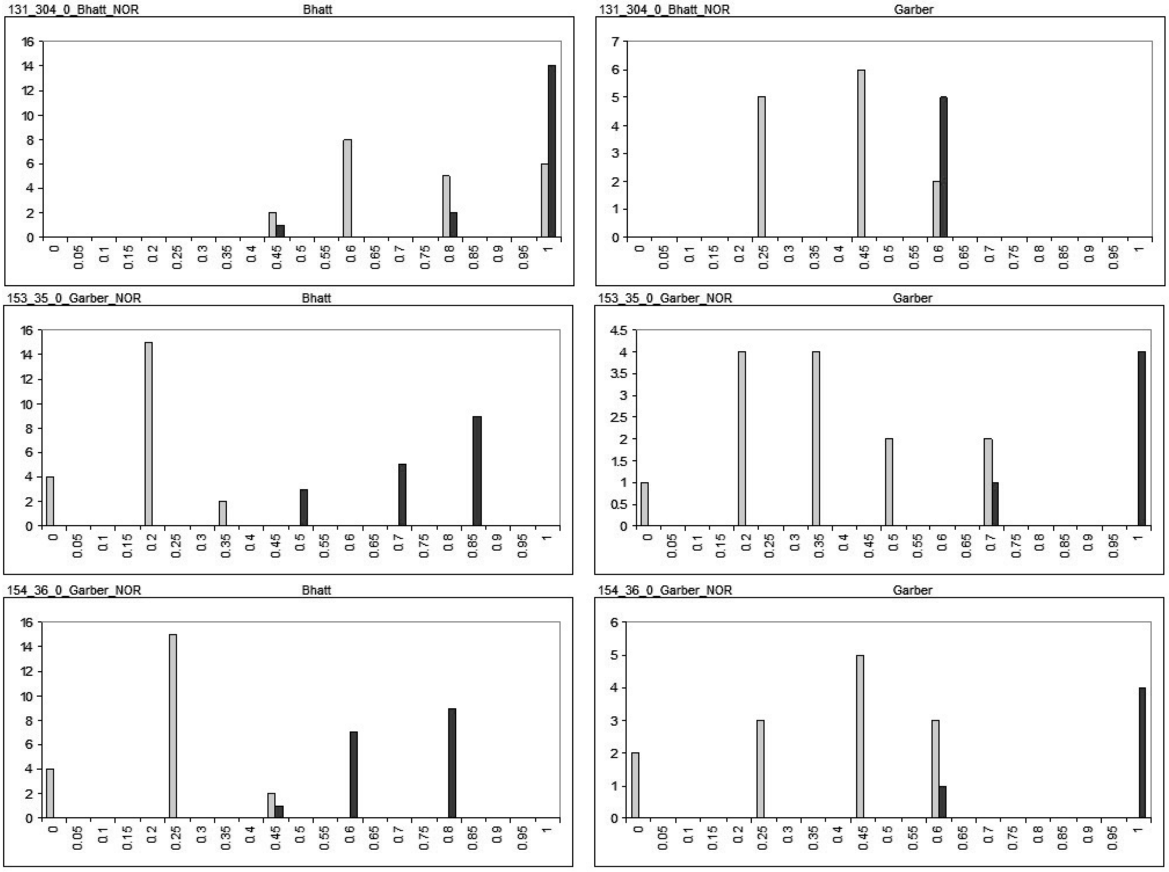


Figure A.16: Histogram of Lung connected components (b).

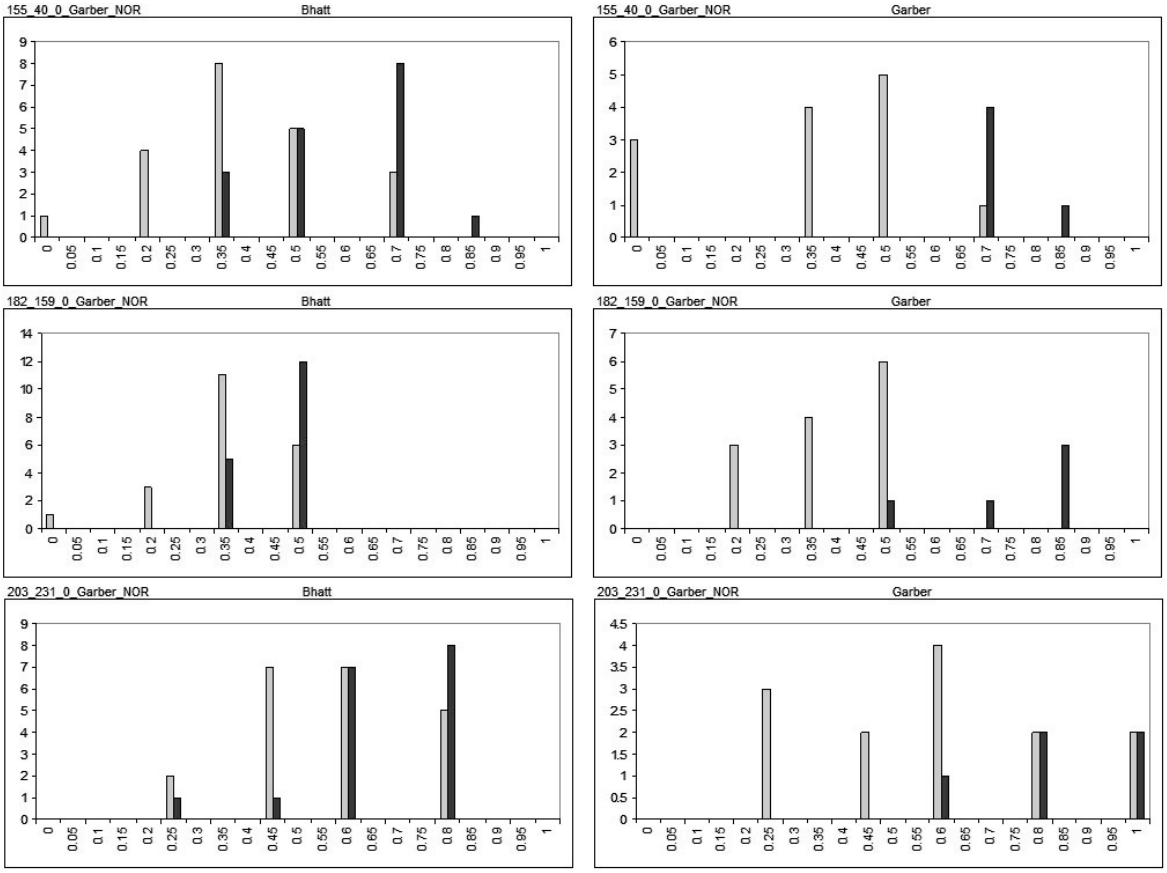


Figure A.17: Histogram of Lung connected components (c).

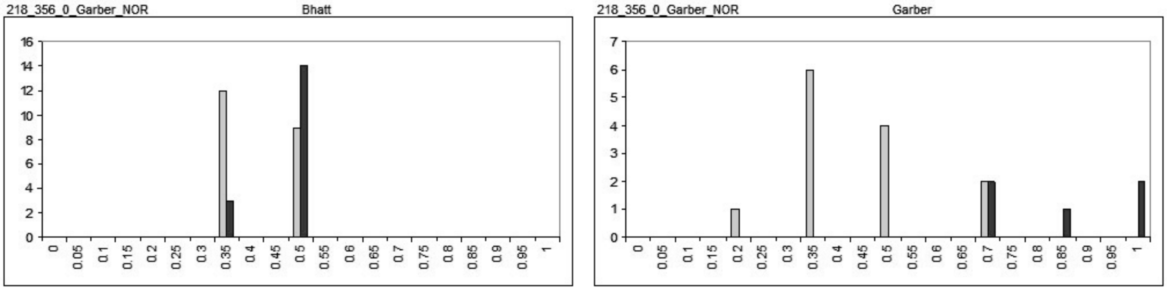


Figure A.18: Histogram of Lung connected components (d).

Bibliography

Akutsu, T., S. MIYANO, and S. KUHARA (1999). Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Pacific Symposium on Biocomputing*, pp. 17–28.

Allison, D. B., X. Cui, G. P. Page, and M. Sabripour (2006). Microarray data analysis: from disarray to consolidation to consensus. *Nature Reviews: Genetics 7*, 55–65.

Alter, O., P. O. Brown, and D. Botstein (2000). Singular value decomposition for genome-wide expression data processing and modeling. *PNAS 97(18)*, 10101–10106.

Armstrong, S. A., J. E. Staunton, L. B. Silverman, R. Pieters, M. L. den Boer, M. D. Minden, S. E. Sallan, E. S. Lander, T. R. Golub, and S. J. Korsmeyer (2002, 2002/01//). Mll translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics 30*, 41 – 7.

Auliac, C., V. Frouin, X. Gidrol, and F. D'alche-Buc (2008, February). Evolutionary approaches for the reverse-engineering of gene regulatory networks: a study on a biologically realistic dataset. *BMC Bioinformatics 9*, 91+.

Baker, P., J. Kearney, B. Gong, A. Merriam, D. Kuhn, J. Porter, and J. Rafael-Fortney (2006). Analysis of gene expression differences between utrophin/dystrophin-deficient vs mdx skeletal muscles reveals a specific upregulation of slow muscle genes in limb muscles. *Neurogenetics 7(2)*, 81–91.

Balagopal, P., R. Olney, D. Darmaun, E. Mougey, M. Dokler, G. Sieck, and D. Hammond (2006). Oxandrolone enhances skeletal muscle myosin synthesis and alters global gene expression profile in duchenne muscular dystrophy. *Am J Physiol Endocrinol Metab. 290(3)*, E530–9.

Baldi, P. and A. D. Long (2001). A bayesian framework for the analysis of microarray expression data: Regularized t-test and statistical inferences of gene changes. *Bioinformatics 17*, 509–519.

Benito, M., J. Parker, Q. Du, J. Wu, D. Xiang, C. M. Perou, and J. S. Marron (2004). Adjustment of systematic microarray data biases. *Bioinformatics 20*, 105–114.

Benjamini, Y. and D. Yekutieli (2001). The control of the false discovery rate in multiple testing under dependency. *Annals of Statistics 29*, 1165–1188.

Berger, S. I. and R. Iyengar (2009). Network analyses in systems pharmacology. *Bioinformatics 25*(19), 2466–2472.

Bhattacharjee, A., W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. J. Mark, E. S. Lander, W. Wong, B. E. Johnson, T. R. Golub, D. J. Sugarbaker, and M. Meyerson (2001). Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses. *Proceedings of the National Academy of Sciences of the United States of America 98*(24), 13790–13795.

Bilban, M., L. Buehler, S. Head, G. Desoye, and V. Quaranta (2002). Normalizing dna microarray data. *Curr Issues Mol Biology 4*, 57–64.

Booden, M. A., D. P. Siderovski, and J. D. Channing (2002). Leukemia-associated rho guanine nucleotide exchange factor promotes g alpha q-coupled activation of rhoa. *Molecular and cellular biology 22(12)*, 4053–61.

Butte, A. J. and I. S. Kohane (2000). Mutual information relevance networks: Functional genomic clustering using pairwise entropy measurements. *Pacific Symposium on Bioinformatics 5*, 415–426.

Chee, M., R. Yang, E. Hubbell, A. Berno, X. C. Huang, D. Stern, J. Winkler, D. J. Lockhart, M. S. Morris, and S. P. A. Fodor (1996). Accessing genetic information with high-density dna arrays. *Science 274(5287)*, 610–614.

Cheng, Y. and G. M. Church (2000). Biclustering of expression data. *Proceedings / ... International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology 8*, 93–103.

Choe, S. E., M. Boutros, A. M. Michelson, G. M. Church, and M. S. Halfon (2005). Preferred analysis methods for affymetrix genechips revealed by a wholly defined control dataset. *Genome Biol 6*, R16.

Chowbina, S. R., X. Wu, F. Zhang, P. M. Li, R. Pandey, H. N. Kasamsetty, and J. Y. Chen (2009). Hpd: an online integrated human pathway database enabling systems biology studies. *BMC bioinformatics 10*(Suppl 11), S5.

Cooper, G. F. and E. Herskovits (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning 9*, 309–347.

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein (2001). *Introduction to Algorithms*. MIT Press and McGraw-Hill.

Cui, X., J. Hwang, J. Qiu, N. Blades, and G. Churchill (2005). Improved statistical tests for differential gene expression by shrinking variance components estimates. *Biostatistics 6(1)*, 59–75.

Dahlquist, K., N. Salomonis, K. Vranizan, S. Lawlor, and B. Conklin (2002). Genmapp, a new tool for viewing and analyzing microarray data on biological pathways. *Biostatistics 31(1)*, 19–20.

de Hoon, M., S. Imoto, and S. Miyano (2002). Statistical analysis of a small set of time-ordered gene expression data using linear splines. *Bioinformatics 18*(11), 1477–1485.

D'Haeseleer, P. (2000). *Reconstructing gene networks from large scale gene expression data*. Ph. D. thesis, The University of New Mexico. Adviser-Forrest, Stephanie.

Dhaeseleer, P., S. Liang, and R. Somogyi (2000). Genetic network inference: From co-expression clustering to reverse engineering. *Bioinformatics 16(8)*, 707–26.

D'haeseleer, P., X. Wen, S. Fuhrman, and R. Somogyi (1998). Mining the gene expression matrix: inferring gene relationships from large scale gene expression data. In *IPCAT '97: Proceedings of the second international workshop on Information processing in cell and tissues*, New York, NY, USA, pp. 203–212. Plenum Press.

Dilip, R. and A. Pankaj (2005, March). Inferring pathways from gene lists using a literature-derived network of biological relationships. *Bioinformatics 21*(6), 788–793.

Djebbari, A. and J. Quackenbush (2008). Seeded bayesian networks: Constructing genetic networks from microarray data. *BMC Systems Biology 2*(1), 57+.

Dong, D., C.-Y. Cui, B. Mow, and L. Wong (2009). Deciphering drug action and escape pathways: An example on nasopharyngeal carcinoma. In *BICoB '09: Proceedings of the 1st International Conference on Bioinformatics and Computational Biology*, Berlin, Heidelberg, pp. 199–210. Springer-Verlag.

Doniger, S. W., N. Salomonis, K. D. Dahlquist, K. Vranizan, S. C. Lawlor, and B. R. Conklin (2003). Mappfinder: using gene ontology and genmapp to create a global gene-expression profile from microarray data. *Genome biology 4*(1), R7.

Draghici, S., P. Khatri, A. L. Tarca, K. Amin, A. Done, C. Voichita, C. Georgescu, and R. Romero (2007, September). A systems biology approach for pathway level analysis. *Genome Res. 17*, 1537–1545.

Efron, T. R., S. J.D., and T. V. (2001, December). Empirical bayes analysis of a microarray experiment. *Journal of the American Statistical Association 96*, 1151–1160.

Efroni, S., C. F. Schaefer, and K. H. Buetow (2007). Identification of key processes underlying cancer phenotypes using biologic pathway analysis. *PLoS ONE 2*(5), e425.

Ein-Dor, L., I. Kela, G. Getz, D. Givol, and E. Domany (2005). Outcome signature genes in breast cancer: is there a unique set? *Bioinformatics 21*(2), 171–178.

Eisen, M. B. and P. O. Brown (1999). Dna arrays for analysis of gene expression. *Methods Enzymol 303*, 179–205.

Eisen, M. B., P. T. Spellman, P. O. Brown, and D. Botstein (1998, December). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the United States of America 95*(25), 14863–14868.

Elliott, B., M. Kirac, A. Cakmak, G. Yavas, S. Mayes, E. Cheng, Y. Wang, C. Gupta, G. Ozsoyoglu, and Z. M. Ozsoyoglu (2008, August). Pathcase: Pathways database system. *Bioinformatics 24(21)*, 2526–2533.

Friedman, N., M. Linial, I. Nachman, and D. Pe'er (2000). Using bayesian networks to analyze expression data. *J Comp Biol 7*, 601–620.

Garber, M. E., O. G. Troyanskaya, K. Schluens, S. Petersen, Z. Thaesler, M. Pacyna-Gengelbach, M. van de Rijn, G. D. Rosen, C. M. Perou, R. I. Whyte, R. B. Altman, P. O. Brown, D. Botstein, and I. Petersen (2001). Diversity of gene expression in adenocarcinoma of the lung. *Proceedings of the National Academy of Sciences of the United States of America 98*(24), 13784–13789.

Garvey, S., C. Rajan, A. Lerner, W. Frankel, and G. Cox. (2002). The muscular dystrophy with myositis (mdm) mouse mutation disrupts a skeletal muscle-specific domain of titin. *Genomics 79(2)*, 146–9.

Gerull, B., M. Gramlich, J. Atherton, M. McNabb, K. T. K, S. Sasse-Klaassen, J. Seidman, C. Seidman, H. Granzier, S. Labeit, M. Frenneaux, and L. Thierfelder (2002). Mutations of ttn, encoding the giant muscle filament titin, cause familial dilated cardiomyopathy. *Nat Genet. 30(2)*, 201–4.

Ghosh, D. and A. Chinnaiyan (2004). Classification and selection of biomarkers in genomic data using lasso. The University of Michigan Department of Biostatistics Working Paper Series 1041, Berkeley Electronic Press.

Goeman, J. J., S. A. van de Geer, F. de Kort, and H. C. van Houwelingen (2004). A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics 20*(1), 93–99.

Golub, T. R., D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander (1999, October). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science 286*(5439), 531–537.

Google (2009). Google. `http://code.google.com/apis/chart/`.

Green ML, K. P. (2006). The outcomes of pathway database computations depend on pathway ontology. *Nucleic Acids Res Aug 7;34(13)*, 3687–97.

Hackman, P., A. Vihola, H. Haravuori, S. Marchand, J. Sarparanta, J. D. Seze, S. Labeit, C. Witt, L. Peltonen, I. Richard, and B. Udd (2002). Tibial muscular dystrophy is a titinopathy caused by mutations in ttn, the gene encoding the giant skeletal-muscle protein titin. *American Journal of Human Genetics Sep 71(3)*, 492–500.

Hanisch, D., A. Zien, R. Zimmer, T. Lengauer, and Thomas (2002). Co-clustering of biological networks and gene expression data. *Bioinfomatics 18 (Supp. 1)*, S145–S154.

Haslett, J. N., D. Sanoudou, A. T. Kho, R. R. Bennett, S. A. Greenberg, I. S. Kohane, A. H. Beggs, and L. M. Kunkel (2002). Gene expression comparison of biopsies from Duchenne muscular dystrophy (DMD) and normal skeletal muscle. *Proceedings of the National Academy of Sciences of the United States of America 99*(23), 15000–15005.

He, X. and J. Zhang (2006, June). Why do hubs tend to be essential in protein networks? *PLoS genetics 2*(6), e88+.

Henegar, C., R. Cancello, S. Rome, H. Vidal, K. Clément, and J.-D. Zucker (2006). Clustering biological annotations and gene expression data to identify putatively co-regulated biological processes. *J. Bioinformatics and Computational Biology 4*(4), 833–852.

Hoopes, L. (2008). Genetic diagnosis: Dna microarrays and cancer. *Nature Education 1*, 1.

Ideker, T., O. Ozier, B. Schwikowski, and A. F. Siegel (2002, July). Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics (Oxford, England) 18 Suppl 1*(suppl_1), S233–240.

Ingenuity (1998). Ingenuity. `http://www.ingenuity.com/`.

Ishii, M., S. Hashimoto, S. Tsutsumi, Y. Wada, K. Matsushima, T. Kodama, and H. Aburatani (2000). Direct comparison of genechip and sage on the quantitative accuracy in transcript profiling analysis. *Genomics 68(2)*, 136–43.

Itoh-Satoh, M., T. Hayashi, H. Nishi, Y. Koga, T. Arimura, T. Koyanagi, M. Takahashi, S. Hohda, K. Ueda, T. Nouchi, M. Hiroe, F. Marumo, T. Imaizumi, M. Yasunami, and A. Kimura (2002). Titin mutations as the molecular basis for dilated cardiomyopathy. *Biochem Biophys Res Commun. 291(2)*, 385–93.

Johnson, W. Evan, Li, Cheng, Rabinovic, and Ariel (2007, January). Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics 8*(1), 118–127.

Joshi-Tope, G., M. Gillespie, I. Vastrik, P. D'Eustachio, E. Schmidt, B. de Bono, B. Jassal, G. R. Gopinath, G. R. Wu, L. Matthews, S. Lewis, E. Birney, and L. Stein (2005, January). Reactome: a knowledgebase of biological pathways. *Nucleic Acids Res 33*(Database issue), D619–622.

Joshi-Tope, G., I. Vastrik, G. Gopinathrao, L. Matthews, E. Schmidt, M. Gillespie, P. D'Eustachio, B. Jassal, S. Lewis, G. Wu, E. Birney, , and L. Stein (2003). The genome knowledgebase: A resource for biologists and bioinformaticists. *Quant Biol. 68*, 237–43.

Juliana, L., Y. Andrew, F. S. A. Jr, Y. Bing, K. de los Santos, and S. P. Goff (2006). Interaction of moloney murine leukemia virus matrix protein with iqgap. *The EMBO Journal 25*, 2155 – 2166.

Kanehisa, M. and S. Goto (2000). Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research 28*, 27–30.

Kanehisa, M., S. Goto, M. Hattori, K. F. Aoki-Kinoshita, M. Itoh, S. Kawashima, T. Katayama, M. Araki, and M. Hirakawa (2006). From genomics to chemical genomics: new developments in kegg. *Nucleic Acids Research 34*, 354–357.

Karp, P. D., C. A. Ouzounis, C. Moore-Kochlacs, L. Goldovsky, P. Kaipa, D. Ahren, S. Tsoka, N. Darzentas, V. Kunin, and N. Lopez-Bigas (2005). Expansion of the biocyc collection of pathway/genome databases to 160 genomes. *Nucleic Acids Research 19*, 6083–89.

Katzav, S. (2007). Flesh and blood: the story of vav1, a gene that signals in hematopoietic cells but can be transforming in human malignancies. *Cancer Lett 255(2)*, 241–54.

Kauffman, S. A. (1969). Metabolic stability and epigenisis in randomly construced genetic nets. *Journal of Theoretical Biology 22*, 437–467.

Kauffman, S. A. (1993, June). *The Origins of Order: Self-Organization and Selection in Evolution* (1 ed.). Oxford University Press, USA.

Kawashima, S., T. Katayama, Y. Sato, and M. Kanehisa (2003). Kegg api: A web service using soap/wsdl to access the kegg system. *Genome Informatics 14*, 673–674.

Khatri, P. and S. Draghici (2005). Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics 21*(18), 3587–3595.

Kotecha, N., K. Bruck, W. Lu, and N. Shah (2008). Pathway knowledge base: An integrated pathway resource using biopax. *Appl. Ontol. 3*(4), 235–245.

Krishna, D. and J. LeDoux (2006, May). Murine leukemia virus particles activate rac1 in hela cells. *Biochem Biophys Res Commun 345*(3), 1184–93.

Krishnamurthy, L., J. Nadeau, G. Ozsoyoglu, M. Ozsoyoglu, G. Schaeffer, M. Tasan, and W. Xu (2003, May). Pathways database system: an integrated system for biological pathways. *Bioinformatics 19*(8), 930–937.

Kristelly, R., G. Gao, and J. J. G. Tesmer (2004, November). Structural determinants of rhoa binding and nucleotide exchange in leukemia-associated rho guanine-nucleotide exchange factor. *The Journal of Biological Chemistry 279*, 47352–47362.

Kuo, W., T. Jenssen, A. Butte, L. Ohno-Machado, and I. Kohane (2002). Analysis of matched mrna measurements from two different microarray technologies. *Bioinformatics 18(3)*, 405–12.

Lähdesmäki, H., S. Hautaniemi, I. Shmulevich, and O. Yli-Harja (2006). Relationships between probabilistic boolean networks and dynamic bayesian networks as models of gene regulatory networks. *Signal Process. 86*(4), 814–834.

Lander, E. (1999). Array of hope. *Nat. Genetics 21*, 3–4.

Lapointe, J., C. Li, J. P. Higgins, M. van de Rijn, E. Bair, K. Montgomery, M. Ferrari, L. Egevad, W. Rayford, U. Bergerheim, P. Ekman, A. M. DeMarzo, R. Tibshirani, D. Botstein, P. O. Brown, J. D. Brooks, and J. R. Pollack (2004). Gene expression profiling identifies clinically relevant subtypes of prostate cancer. *Proceedings of the National Academy of Sciences of the United States of America 101*(3), 811–816.

Lee, H. K., W. Braynen, K. Keshav, and P. Pavlidis (2005). Erminej: tool for functional analysis of gene expression datasets. *BMC Bioinformatics 6*, 269.

Liang, S., S. Fuhrman, and R. Somogyi (1998). Reveal: a general reverse engineering algorithm for inference of genetic network architectures. *Pacific Symposium Biocomputing 1*, 18–29.

Liu, M., X.-w. Chen, and R. Jothi (2009, October). Knowledge-guided inference of domain-domain interactions from incomplete protein-protein interaction networks. *Bioinformatics 25*(19), 2492–2499.

Liu, M., A. Liberzon, S. W. Kong, W. R. Lai, P. J. Park, I. S. Kohane, and S. Kasif (2007, June). Network-based analysis of affected biological processes in type 2 diabetes models. *PLoS Genet 3*(6), e96+.

Madeira, S. C. and A. L. Oliveira (2004). Biclustering algorithms for biological data analysis: A survey. *IEEE Transactions on Computational Biology and Bioinformatics 1*(1), 24–45.

Michiels, S., S. Koscielny, and C. Hill (2005, February). Prediction of cancer outcome with microarrays: a multiple random validation strategy. *The Lancet 365*(9458), 488–492.

Murphy, K. (2001). Learning bayes net structure from sparse data sets. technical report.

NCBI (2009). Ncbi. `http://www.ncbi.nlm.nih.gov/`.

Ogata, H., S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa (1999). Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res 27*, 29–34.

PathwayAPI (2009). Pathwayapi. `http://www.pathwayapi.com`.

PathwayExpress (2009). Pathwayexpress. `http://vortex.cs.wayne.edu/projects.htm`.

Pavlidis, Lewis, and Noble (2002). Exploring gene expression data with class scores. In *Pac. Symp. Biocomput*, pp. 474–485.

Pavlidis, J. Qin, V. Arango, J. Mann, and E. Sibille (2004a). Using the gene ontology for microarray data mining: A comparison of methods and application to age effects in human prefrontal cortex. *Neurochemical Research 29(6)*, 1213–1222.

Pavlidis, P., J. Qin, V. Arango, J. J. Mann, and E. Sibille (2004b, June). Using the gene ontology for microarray data mining: A comparison of methods and application to age effects in human prefrontal cortex. *Neurochemical Research 29*(6), 1213–1222.

Pescatori, M., A. Broccolini, C. Minetti, E. Bertini, C. Bruno, A. D'amico, C. Bernardini, M. Mirabella, G. Silvestri, V. Giglio, A. Modoni, M. Pedemonte, G. Tasca, G. Galluzzi, E. Mercuri, P. A. Tonali, and E. Ricci (2007). Gene expression profiling in the early phases of DMD: a constant molecular signature characterizes DMD muscle from early postnatal life throughout disease progression. *FASEB J. 21*(4), 1210–1226.

Prasad, T. S., K. Kandasamy, and A. Pandey (2009). Human protein reference database and human proteinpedia as discovery tools for systems biology. *Methods in Molecular Biology 577*, 67–79.

Quackenbush, J. (2002). Microarray data normalization and transformation. *Nature Genetics 32*, 496–501.

Rhodes, D. R., J. Yu, K. Shanker, N. Deshpande, R. Varambally, D. Ghosh, T. Barrette, A. Pandey, and A. M. Chinnaiyan (2004). Large-scale meta-analysis of cancer microarray data identifies common transcriptional profiles of neoplastic transformation and progression. *PNAS 101(25)*, 9309–9314.

Rogers, S. and M. Girolami (2005). A bayesian regression approach to the inference of regulatory networks from gene expression data. *Bioinformatics 21(14)*, 3131–7.

Rogojina, A., W. Orr, B. Song, and E. G. Jr (2003). Comparing the use of affymetrix to spotted oligonucleotide microarrays using two retinal pigment epithelium cell lines. *Mol Vis 6:9*, 482–96.

Romero, P., J. Wagg, M. L. Green, D. Kaiser, M. Krummenacker, and P. D. Karp (2005). Computational prediction of human metabolic pathways from the complete human genome. *Genome Biol 6(1)*, R2.

Ross, M. E., R. Mahfouz, M. Onciu, H. C. Liu, X. Zhou, G. Song, S. A. Shurtleff, S. Pounds, C. Cheng, J. Ma, R. C. Ribeiro, J. E. Rubnitz, K. Girtman, W. K. Williams, S. C. Raimondi, D. C. Liang, L. Y. Shih, C. H. Pui, and J. R. Downing (2004, December). Gene expression profiling of pediatric acute myelogenous leukemia. *Blood 104*(12), 3679–3687.

Rousseeuw, P. (1987, November). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math. 20*(1), 53–65.

Salomonis, N., K. Hanspers, A. C. Zambon, K. Vranizan, S. C. Lawlor, K. D. Dahlquist, S. W. Doniger, J. M. Stuart, B. R. Conklin, and A. R. Pico (2007, June). Genmapp 2: New features and resources for pathway analysis. *BMC Bioinformatics 8*, 217+.

Segal, E., M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman (2003, June). Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics 34*(2), 166–176.

Segal, E., H. Wang, and D. Koller (2003). Discovering molecular pathways from protein interaction and gene expression data. In *ISMB (Supplement of Bioinformatics)*, pp. 264–272.

Shannon, C. and W. Weaver (1963). The mathematical theory of communication. *University of Illinois Press 1*, 1.

Shannon, P., A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker (2003, November). Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res 13*(11), 2498–2504.

Singh, D., P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, E. S. Lander, M. Loda, P. W. Kantoff, T. R. Golub, and W. R. Sellers (2002, 2002/03//). Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell 1*, 203 – 9.

Sivachenko, A. Y., A. Yuryev, N. Daraselia, and I. Mazo (2005). Identifying local gene expression patterns in biomolecular networks. In *CSBW '05: Proceedings of the 2005 IEEE Computational Systems Bioinformatics Conference - Workshops*, Washington, DC, USA, pp. 180–184. IEEE Computer Society.

Sivachenko, A. Y., A. Yuryev, N. Daraselia, and I. Mazo (2007). Molecular networks in microarray analysis. *J. Bioinformatics and Computational Biology 5*(2b), 429–456.

Smyth, G. K. and G. K. Smyth (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Stat. Appl. Genet. Mol. Biol 3*.

Soh, D., D. Dong, Y. Guo, and L. Wong (2007). Enabling more sophisticated gene expression analysis for understanding diseases and optimizing treatments. *ACM SIGKDD Explorations 9*, 3–14.

Soh, D., D. Dong, Y. Guo, and L. Wong (2009). Consistency, comprehensiveness, and compatibility of pathway databases. *BMC Bioinformatics 2nd revision*, manuscript.

Sohler, F., D. Hanisch, and R. Zimmer (2000). New methods for joint analysis of biological networks and expression data. *Bioinformatics 20*, 1517–1521.

Subramanian, A., P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Elbert, M. A. Gillete, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *PNAS 102(43)*, 15545–15550.

Tan, P., T. Downey, E. J. Spitznagel, P. Xu, D. Fu, D. Dimitrov, R. Lempicki, B. Raaka, and M. Cam (2003). Evaluation of gene expression measurements from commercial microarray platforms. *Nucleic Acids Res 31(19)*, 5676–84.

Tarca, A. L., S. Draghici, P. Khatri, S. S. Hassan, P. Mittal, J.-S. Kim, C. J. Kim, J. P. Kusanovic, and R. Romero (2008, November). A novel signaling pathway impact analysis (spia). *Bioinformatics 25*(1), 75–82.

Tavazoie, S., J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church (1999, July). Systematic determination of genetic network architecture. *Nature genetics 22*(3), 281–285.

Thomas, P. D., M. J. Campbell, A. Kejariwal, H. Mi, B. Karlak, R. Daverman, K. Diemer, A. Muruganujan, and A. Narechania. (2003). Panther: a library of protein families and subfamilies indexed by function. *Neurochemical Research 13*, 2129–2141.

Tian, L., S. A. Greenberg, S. W. Kong, J. Altschuler, I. S. Kohane, and P. J. Park (2005). Discovering statistically significant pathways in expression profiling studies. *PNAS 102*, 13544–13549.

Tusher, V. G., R. Tibshirani, and G. Chu (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences of the United States of America 98*, 5116–5121.

van Iersel, M., T. Kelder, A. Pico, K. Hanspers, S. Coort, B. Conklin, and C. Evelo (2008). Presenting and exploring biological pathways with pathvisio. *BMC Bioinformatics 9*, 399.

Wang, J., Q. Rao, M. Wang, H. Wei, H. Xing, H. Liu, Y. Wang, K. Tang, L. Peng, Z. Tian, and J. Wang (2009, Sept). Overexpression of rac1 in leukemia patients and its role in leukemia cell migration and growth. *Biochem Biophys Res Commun 386*(4), 769–74.

Wen, X., S. Fuhrman, G. S. Michaels, D. B. Carr, S. Smith, J. L. Barker, and R. Somogyi (1998). Large-scale temporal gene expression mapping of central nervous system development. *PNAS 95*(1), 334–339.

Wen-Son, H., S. Ross, P. Bee-Keow, L. Thomas, D. Difeng, S. Donny, W. Limsoon, G. Simon, C. Judy, C. Chun-Ying, L. Yoke-Fong, L. Soo-Chin, M. Benjamin, S. Richie, M. Salto-Tellez, and G. Boon-Cher (2009, Feb). Pharmacodynamic effects of seliciclib, an orally administered cell cycle modulator, in undifferentiated nasopharyngeal cancer. *Clinical Cancer Research 15*(4), 1435–1442.

Wikipathways (2004). Wikipathways. `http://www.wikipathways.com`.

Yang, Y. H., S. Dudoit, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T. P. Speed (2002, February). Normalization for cdna microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res 30*(4), e15.

Yeoh, E., M. Ross, S. Shurtleff, W. Williams, D. Patel, R. Mahfouz, F. Behm, S. Raimondi, M. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C. Pui, W. Evans, C. Naeve, L. Wong, and J. Downing (2002). Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell 1*(2), 133–143.

Yi, S.-G., Y.-J. Joo, and T. Park (2009). Rank-based clustering analysis for the time-course microarray data. *J. Bioinformatics and Computational Biology 7*(1), 75–91.

Yoo, C., V. Thorsson, and G. F. Cooper (2002). Discovery of causal relationships in a gene regulation pathway from a mixture of experimental and observational dna microarray data. *Paci c Symposium on Biocomputing 1*, 498–509.

Yuen, T., E. Wurmbach, R. Pfeffer, B. Ebersole, and S. Sealfon (2002). Accuracy and calibration of commercial oligonucleotide and custom cdna microarrays. *Nucleic Acids 30(10)*, e48.

Zhang, M., L. Zhang, J. Zou, C. Yao, H. Xiao, Q. Liu, J. Wang, D. Wang, C. Wang, and Z. Guo (2009). Evaluating reproducibility of differential expression discoveries in microarray studies by considering correlated molecular changes. *Bioinformatics 25*(13), 1662–1668.