The Budgeted Biomarker Discovery Problem

by

Sheehan Veikko Khan

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

University of Alberta

Abstract

Researchers conduct association studies to discover biomarkers in order to gain new biological insight on complex diseases and phenotypes. Although most researchers have intuitions about what defines a biomarker and how to assess the results of an association study, there is neither a formal definition for what a biomarker is, nor objective goal for association studies. As a result, the literature is full of association studies with conflicting results – *e.g.*, studies on the same phenotype that produce lists of biomarkers with little to no overlap.

This thesis presents the "Budgeted Biomarker Discovery (BBD) problem", which clearly defines (1) what a biomarker is, and (2) rewards for correctly identifying biomarkers and penalties for incorrectly identifying biomarkers. Furthermore, the BBD problem allows researchers to use a mixture of high- and low-throughput technologies. In the context of discovering biomarkers from gene expression data, we show how future association studies can use both microarrays and qPCR data to objectively find the genes that are biomarkers in a cost efficient manner.

We present several algorithms for solving the BBD problem, and show that good algorithms must make use of both microarrays and qPCR. Also, they must be able to adapt to the data as it is collected. For example, when solving a new BBD problem, we must begin by collecting microarrays because we do not yet know how many biomarkers we expect to identify, or which qPCR arrays would be most informative. Thus, we use the high-throughput microarrays to survey the problem, until we can identify which specific low-throughput qPCR arrays to use for focusing on those genes that are potentially biomarkers. To identify when this transition should occur, we present the problem of estimating the density of univariate statistics in high-throughput data, and we present our Fused Density Estimation (FDE) algorithm as a solution. We use FDE as the backbone of our adaptive algorithms for solving BBD problems. In a series of experiments on real microarray data and realistic synthetic data, we show that our BBD1 algorithm is the most robust solution, amongst those considered, to the BBD problem.

Preface

This thesis is an original work by Sheehan Veikko Khan. It builds of his previously published works:

1. **S. Khan**, R. Greiner. ”Budgeted Transcript Discovery: A Framework For Joint Exploration And Validation Studies”, BIBM 2014

2. **S. Khan**, R. Greiner. ”The Budgeted Biomarker Discovery Problem: A Variant of Association Studies”, AAAI workshop on Modern Artificial Intelligence for Health Analytics (MAIHA), 2014

3. **S. Khan**, R. Greiner. ”Finding Discriminatory Genes: a methodology for validating microarray studies”, ICDM workshop on Data Mining for Biomedical Applications (BioDM), 2013

4. C. Stretch, **S. Khan**, N. Asgarian, R. Eisner, S. Vaisipour, S. Damaraju, O. Bathe, H. Steed, R. Greiner, V. Baracos. ”Effects of sample size on differential gene expression, rank order and prediction accuracy of a gene signature”, PLoS One, 2013

The first three papers are related to the thesis because they build upon the BBD problem definition. In particular, our TNAS-FDR, BBD1 and BBD-Greedy are clear improvements of our previous RR, RR+RR, and mLUCB algorithms from the BIBM paper (#1 above) respectively. The workshop papers (#2 and #3 above) also had other algorithms, but they were very specific to the models we used at the time, and as those models have become outdated so too have the algorithms.

The PloS One paper (#4 above) addresses the issue of statistically underpowered microarray studies present in the literature. It uses a sub-sampling procedure on a very large dataset (to date the largest human skeletal muscle dataset) to show the sample size required for reliable results. In this thesis we re-use the dataset and the sub-sampling approach to generate realistic synthetic data for our experimental results.

The average Ph.D. thesis is nothing but a transference of bones from one graveyard to another.

– J. Frank Dobie, “A Texan in England”, 1945.

Acknowledgements

First and foremost I would like to acknowledge my family for their endless love and support. My parents, Shams and Eila Khan, and my sister, Tanya Parrag. Without them none of this would have been possible.

Academically, I definitely must thank my supervisor, Russ Greiner. Not only did he teach me about machine learning, bioinformatics and how to write this thesis, but he also taught me about the irrelvant show and gave me other various good tips and pointers over the years. Also, our colleagues Vickie Baracos, Cynthia Stretch, Csaba Szepesvári, Tor Lattimore, and Rick Valenzano, helped to make this thesis by offering their time and expertise along the way.

I would like to thank NSERC, AITF, PIMS, MITACS, and the government of Alberta for their generous scholarships, which enabled me to focus more on research and less on financials.

Lastly I would like to say that it takes more than support, supervision and funding to get a PhD. The process is long and its important to take breaks and live a little along the way. I would like to thank all my Edmonton friends for being there to have a coffee/beer, or go climbing/curling/roller-blading/running, or hanging out on the weekends and play games. There are far too many of you for me to name here, and I will not even try to pick and choose amongst you to fill the space. Just know that if you are reading this now, than you are one of those people I would have liked to acknowledge.

“So long, and thanks for all the fish.”
– Douglas Adams, ”he Hitchhiker’s Guide to the Galaxy”, 1979

Table of Contents

List of Figures

List of Symbols

Symbol	Description
α	Level of FDR control applied to a t-test
B	The total experimentation budget
β	The importance of recall, relative to precision
C_t	The cost of test t
Δ_i, $\hat{\Delta}_i$	The standardized effect size of expression values for gene g_i, (empirical estimate is $\hat{\Delta}_i$)
$\Delta_i^{(t)}$, $\hat{\Delta}_i^{(t)}$	The standardized effect size of expression values for gene g_i when observed by test t (empirical estimate is $\hat{\Delta}_i^{(t)}$)
Δ^*	The minimal effect size of relevant genes
ε	Random variable modelling additive noise
F_x, $\hat{F}_x$	The CDF of random variable x, (empirical estimate is $\hat{F}_x$)
f_x, $\hat{f}_x$	The PDF of random variable x, (empirical estimate is $\hat{f}_x$)
Φ_x, $\hat{\Phi}_x$	The characteristic function of f_x, (empirical estimate is $\hat{\Phi}_x$)
$\mathcal{G}$	The set of all genes, $\mathcal{G} = \{g_1, \ldots, g_N\}$
$\mathcal{G}_t$	The subset of genes covered by test t
j	The square root of negative one, $j = \sqrt{-1}$
N	The total number of genes, $N = \lvert\mathcal{G}\rvert$
N_{PCR}	The number of genes covered on a custom qPCR array
R	The set of relevant genes, $R = \{g_i : \Delta_i \geq \Delta^*\}$
$\mathcal{T}$	The set of all tests
$\psi_{i,s}^{(t)}$	Expression value of gene g_i, of a patient in class s, using test t
$\psi_{i,s,m}^{(t)}$	The m'th observed expression value of gene g_i, of a patient in class s, using test t
	Parameterized Distributions
$\mathcal{N}\left(\mu, \sigma^2\right)$	The normal distribution with mean μ, and variance σ^2
Laplace (μ, b)	The Laplace distribution with location and scale parameters, μ, and b respectively
Cauchy (x_0, γ)	The Cauchy distribution with location and scale parameters, x_0, and γ respectively
$\mathcal{U}\left(a, b\right)$	The uniform distribution over the interval $[a, b]$
$T\left(\nu, \mu\right)$	The non-central t-distribution, with ν degrees of freedom, and non-centrality parameter μ
$\chi^2(\nu)$	The chi-squared distribution with ν degrees of freedom

List of Abbreviations

Abbreviation	Meaning	Page
AUC	Area Under the receiver operating Curve	30
BBD	Budgeted Biomarker Discovery	2
BBD-Greedy	Greedy algorithm for BBD	66
BBD1	first BBD algorithm using only qPCR	69
BBD2	second BBD algorithm using only qPCR	71
CDF	Cumulative Density Function	36
DAVID	Database for Annotation Visualization and Integrated Discovery	11
ER	Estrogen Receptor	30
FDE	Fused Density Estimation	4
FDR	False Discovery Rate	12
FFT/iFFT	(inverse) Fast Fourier Transform	37
GEO	Gene Expression Omnibus	8
GO	Gene Ontology	11
GRN	Gene Regulatory Network	30
GSEA	Gene Set Enrichment Analysis	27
IPA	Ingenuity Pathway Analysis	11
ITR	Individualized Treatment Rules	31
KEGG	Kyoto Encyclopedia of Genes and Genomes	11
LASSO	Least Absolute Shrinkage and Selection Operator	28
LUCB	Lower Upper Confidence Bound	34
MSE	Mean Squared Error	55
NGS	Next Generation Sequencing	24
PAM	Predictive Analysis for Microarrays	28
PDF	Probability Density Function	43
qPCR	quantitative Polymerase Chain Reaction	6
SAM	Significance Analysis of Microarrays	28
SNP	Single-Nucleotide Polymorphism	25
SAR	Successive Accepts and Rejects	34
SPRT	Sequential Probability Ratio Test	32
TNAS	Traditional Naive Association Study	64
TNAS-FDR	TNAS algorithm with FDR control	65
TNAS-Omniscient	TNAS oracle algorithm	72
TNAS-Top$K$	TNAS algorithm returning top K genes	66

Chapter 1

Introduction

Many researchers in bioinformatics are concerned with finding "biomarkers", which are features that can be used to identify and separate case specimens (*i.e.*, those that exhibit the phenotype of interest) from control specimens (*i.e.*, those that do not exhibit the phenotype of interest). For example, a gene that is overexpressed in patients with a certain type of cancer, in comparison to expression levels of healthy people, would be called a biomarker for that cancer. There are several reasons that researchers are interested in biomarkers:

1. Biomarkers can be used as diagnostic tools – *e.g.*, 23andMe[1] is a company that uses biomarkers to create tests for: inherited conditions, drug response, genetic risk factors, etc.

2. They can be used for personalized medicine – *e.g.*, biomarkers can be used to identify the type of breast cancer a woman has, which can help determine the treatment that will work best for her [11].

3. Once identified, pharmaceutical companies can develop drugs that specifically target biomarkers to suppress or promote their expression [89].

4. There are many features that could be biomarkers for a specific disease, many of which have yet to be identified. Identifying these biomarkers could initiate follow-up research on them, which could lead to new biological insights about those features, and how they are related to the phenotype.

[1] `https://www.23andme.com`

Biomarkers are typically found by using a high-throughput technology to compare many features in a case versus control experiment. We refer to these experiments as "association studies". While most researchers have intuitions about association studies, currently there is no universally accepted consensus about the specific details, including the goals, of these studies.

In this thesis, we define the Budgeted Biomarker Discovery (BBD) problem. Solving this problem provides a precise way to find biomarkers in the context of gene expression association studies. The BBD framework extends the standard approach of association studies by providing a clear definition for what it means for a gene to be a biomarker, and defines how to reward algorithms for correctly identifying a genes that are biomarkers, and to penalize algorithms for falsely identifying genes as biomarkers when they are not. Furthermore the BBD framework incorporates the use of both high and low-throughput technologies. Thus, when operating under a fixed experimentation budget, the BBD framework allows researchers to determine the most cost effective way to collect their data with the goal of discovering biomarkers.

The terms "biomarker" and "biomarker discovery" have been overloaded in the literature. We are explicitly *NOT* using the term "biomarker discovery" to mean the process of identifying markers for use in clinical tests as described by Pepe *et al.* [69]. As hinted above, our definition of biomarker is based the ability of a feature, independent of the technology used to observe it, to separate the cases from the controls; see Allison *et al.* [2]. Our definition of biomarker discovery includes the use of a high-throughput technology to identify candidate biomarkers that are then checked by a more accurate low-throughput technology. In the context of gene expression, we will provide formal and objective definitions of our interpretation of these terms in Chapter 2. In Section 2.6 we will show how our model can be adapted to other types of 'omics technologies, and beyond bioinformatics we show that the BBD problem can be generalized to other machine learning problems.

1.1 Contributions

This section provides a succinct list of the contributions made in this thesis.

1. *Claim*: Current microarray association studies can benefit from an appropriate and clearly defined objective.

   We formalize the Budgeted Biomarker Discovery (BBD) problem, which incorporates both high- and low-throughput technologies to discover biomarkers within a given experimentation budget. The problem definition itself is a contribution.

   - In current association studies, biomarkers are obtained from either ranked lists, or null hypothesis testing. Unfortunately neither approach defines what it means to be a biomaker and it is therefore impossible to objectively evaluate the study in terms of the statements made about the biomakers. In other words, without a definition of a biomarker we cannot reward a study for correctly finding them, or penalize it for incorrectly finding things that are not biomarkers. Without an objective evaluation criteria for the study, many strange statements can be made. For example, we could say that "all genes are biomarkers", or "no genes are biomarkers", and without penalties for incorrectly labelling genes both statements are perfectly fine. In the BBD problem we provide a clear definition of what a biomarker is, and thus we can partition all genes into the set of genes that are biomarkers and the set of genes that are not. We also provide an evaluation function that compares the set of genes returned as biomarkers to the ground truth based on our definition.

     - Microarray association studies are special case of BBD problems. By treating them as BBD problems, they benefit from our precise definitions.
     - We show that, while there is a growing number of microarray as-

sociation studies, the number of microarrays used per study remains the same. Thus, studies remain underpowered and in need of follow-up checking studies. Because the BBD problem incorporates both high- and low-throughput technologies, it allows us to combine association and checking studies with a single objective. We conjecture that one of the reasons many researchers currently do not perform checking studies, is the lack of a proper definition, but posing the problem as a BBD problem provides that definition.

2. *Claim*: With only a few observations, we can learn useful plate models for gene expression data from high-throughput technologies.

   We present a plate model for gene expression values, and show it can be used to tune algorithms for specific BBD problems, *i.e.*, the algorithm can use the parameters of the plate model to tune its parameters for the BBD problem that it is currently solving. In order to use these plate models, we showed that we must estimate the distribution of the effect size of genes f_Δ; knowing this distribution, we can generate all other parameters in the plate model. We showed that we can get good approximations of f_Δ with very few examples, by exploiting the very high-dimensionality of the problem. To do this we presented and analyzed two simple estimators, which we then combined to make our Fused Density Estimation (FDE) algorithm.

3. *Claim*: Algorithms for solving BBD problems should make use of our plate model.
   Claim: Our BBD1 algorithm is a very robust solution for BBD problems.

   We show that our BBD1 algorithm is the most robust solution, amongst the algorithms we present, for solving BBD problems. BBD1 uses a mixture of microarray and custom PCR data, and can tune its parameters to adapt to new BBD problems to help solve them effectively. Thus BBD1 performs very well across all of our experiments; in most experiments it has the best performance.

4. *Claim*: Greedy algorithms do not provide good solutions for BBD problems.

   We show that BBD problems have a direct analogy to adaptive submodular maximization problems. While previous works have shown that greedy algorithms typically perform well on submodular problems, we show that they do not work well for our BBD problems. The problem with greedy algorithms is that they must compute some measure of utility for all possible tests, and then perform the one with highest utility. Whereas our other algorithms, utilizing only custom PCR arrays, can use our plate model to tune parameters for a good policy, and then behave according to that policy, *i.e.*, Claim 3.

1.2 Outline

In Chapter 2 we present our definitions and models for biomarkers and their discovery, *i.e.*, we formally define the BBD problem. Section 2.5 highlights the benefits of adopting the BBD approach in comparison to previous approaches. Chapter 3 follows with a discussion of related problems in bioinformatics and computing science.

As a precursor to developing our BBD algorithms, we present the problem of density estimation for univariate statistics of high-throughput data in Chapter 4. Solving this problem is not only a crucial step towards solving the BBD problem, but it can also be used for other problems such as determining the sample size needed for specific research outcomes. For example, after analyzing some microarray data, we may be able to show that it is statistically impossible to reliably identify the top 100 genes (that are biomarkers), but this method allows us to predict how many more microarrays would be required to do so. While this may sound similar to the development of realistic simulation models of microarray data, Section 3.1.5 shows that is an entirely different problem. We show that it is impossible to solve this density estimation problem in general, and we highlight the approximation noise introduced by reasonable solutions. We then construct our hybrid FDE algorithm that combines those solutions.

In Chapter 5, we present our algorithms for solving BBD problems. Our algorithms cover two types of BBD problems: 1) where microarray data has already been collected, and our algorithms must analyze it to find the biomarkers, and 2) where the algorithms have the ability to analyze data as it is collected, and thus choose between collecting microarrays or quantitative Polymerase Chain Reaction (qPCR) so that they may discover as many biomarkers as possible under a given experimentation budget. Experimental results in Chapter 6 show that our BBD1 algorithm is a particularly attractive solution for BBD problems. It incorporates both microarray and qPCR data, and it has very robust performance in experiments based on real microarray data and across a spectrum of synthetic datasets. We believe that it corresponds to the solution sought after by researchers interested in objectively validating microarray studies [21, 73].

We close this chapter with a brief description of high- and low-throughput technologies for readers unfamiliar with methods of observing gene expression values. We also provide a description of the current approach to association studies in gene expression, highlighting the issues that are problematic and ill-defined.

1.3 High- Versus Low-Throughput Methods

In bioinformatics, there are two different types of tools used for data collection: high- and low-throughput. As their names imply, high-throughput methods are capable of observing many features simultaneously, and low-throughput methods are capable of observing only a few features simultaneously. For example, we can use a single microarray to measure the expression values of all of Mrs. Smith's genes, or we can use a single qPCR array to measure the values of a small subset of her genes. These two approaches typically trade-off cost for accuracy. High-throughput methods are much more cost effective in terms of cost per feature observed, but low-throughput methods have much less observation noise – microarray costs approximately \$0.01 per gene, and qPCR costs approximately \$1 per gene [28, Table 1].

In the context of studies on gene expression, microarrays are used as the high-throughput device. Current microarrays are capable of measuring most, if not all, RNA transcripts in the human genome.[2] Low-throughput observations are often done via qPCR arrays, which typically measure the expression levels of 100 genes.[3] In practice qPCR is used as the gold standard for measuring gene expression levels [19, 21, 63, 73, 75, 91].

While both microarray and qPCR provide measurements of gene expression, they capture that information very differently. In a microarray, the RNA transcripts are hybridized to the array, such that the transcripts for gene g_i bond at a specific location, (x_i, y_i), on the array. The expression value for gene g_i is then measured as the proportion of RNA at (x_i, y_i) relative to the total amount of RNA across the array. By contrast, qPCR operates by isolating the RNA transcripts specific to gene g_i, and reverse transcribes them into cDNA. The cDNA then undergoes a series of amplification cycles. During each amplification cycle, the amount of cDNA doubles, and for each strand of cDNA produced a photon of light is emitted. Thus, by counting the number of cycles required to observe a fixed amount of light, it is straightforward to calculate the amount of RNA present in the original sample. For more details on microarrays and qPCR (as well as other high- and low-throughput technologies) we refer the reader to Reece & Campbell [19, Chapter 20].

Clearly, we cannot expect that the expression values for gene g_i to be from the same distribution in both technologies. But, if gene g_i really is a biomarker, we assume that both methods should be able to detect a difference between the distributions for the cases and controls.

The impact of these differences is that when we use both high- and low-throughput technologies, we cannot directly mix the data – they produce different numbers, from different distributions. However, in Section 2.1 we will provide a way to mix appropriate summaries of the data.

One last point of interest is that, while there are many readily available "off-the-

[2] The most common brand on the market is the Affymetrix HG-U133-Plus-2, which measures 54 675 mRNA transcripts.

[3] For example, SABiosciences currently offers qPCR arrays with either 100 genes, or 400 genes. `http://www.sabiosciences.com/PCRArrayPlate.php`

shelf" qPCR products on the market, that each test a set of genes that are known to have specific interactions (*e.g.*, genes associated with breast cancer, or genes associated with cell growth, *etc.*), most vendors also offer custom qPCR solutions, which allow researchers to specify the 100 genes they would like to test. In general, such custom qPCR arrays will cost slightly more than purchasing an "off-the-shelf" solution, but will be much more efficient as researchers can avoid purchasing multiple qPCR products in order to test all the candidate genes suggested by their analysis.

1.4 Traditional Association Studies and Their Pitfalls

When performing an association study on gene expression, researchers will collect several microarrays from patients in the case group, and several from patients in the control group. After the data has been collected, the genes are ranked based on their observed difference between the two groups, and the top genes are declared to be the biomarkers.[4] Here we will quickly outline the pitfalls of this approach.

The first problem with microarray association studies is that they are terribly "underpowered", *i.e.*, the number of microarrays used is typically very small in relation to the number of genes measured in each microarray.[5] The community has been very good in mandating the public release of the microarray data collected in these studies when publishing their results. One of the largest public databases of microarray data is the Gene Expression Omnibus (GEO) [32]; currently[6] it has 53 959 datasets composed of 1 313 826 microarrays in total. Figure 1.1[top] shows the number of datasets submitted to GEO per year. Note that the number of microarray datasets submitted has continued to increase since 2002, with approximately 10 000 being submitted in 2013. However, Figure 1.1[bottom] shows that the number of microarrays used per dataset has not increased over time; essentially all have less

[4] Note that we have been intentionally vague here in stating that the genes are ranked based on their difference, without defining difference. This is one of the problems we will discuss in this section.

[5] The statistical power of a test is defined to be the probability that a specific event is detected [36]. For example, if performing a t-test for the null hypothesis that $\mu = 0$ or $\mu \neq 0$, we may define the power to be the probability that we correctly reject the null hypothesis when $\mu = 1$, *i.e.*, $P(\text{reject null hypothesis}|\mu = 1)$. Thus, statistical power is subject to the null hypothesis being tested, and the alternative event of interest.

[6] On January 1, 2015.

than 100 microarrays and the majority have only 10–12.

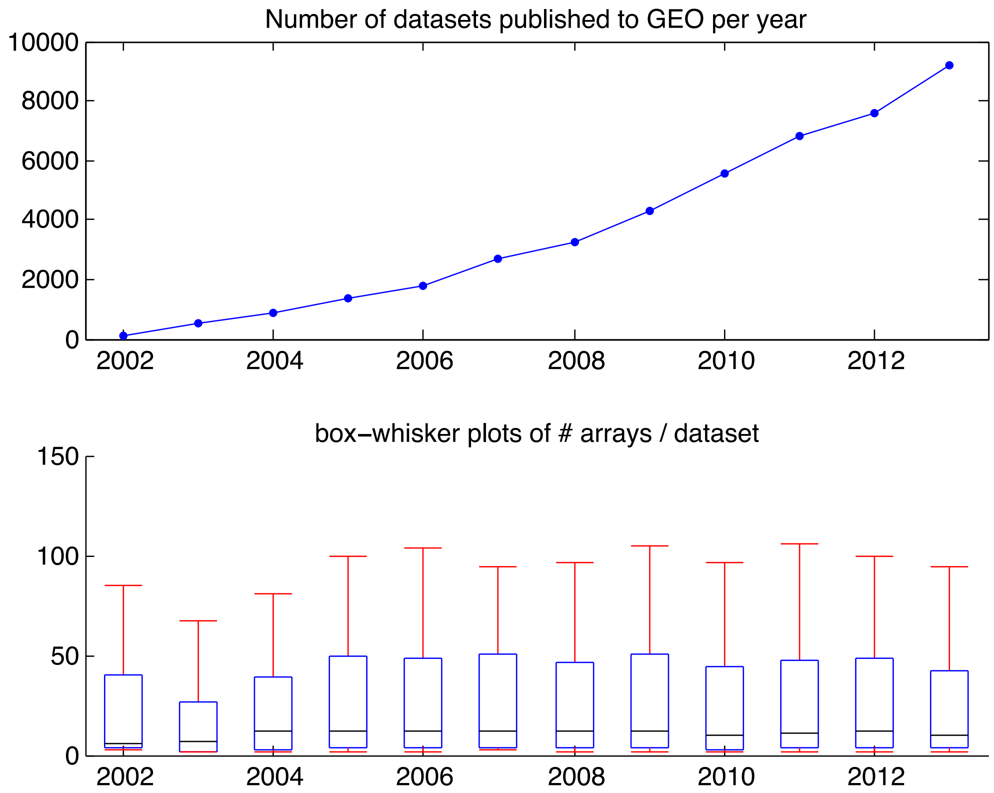

Figure 1.1: [top] Summary of the dataset submitted to GEO per year. [bottom] Box and whisker plots for the number of microarrays per dataset.

As a result of being underpowered, it is difficult to properly identify the genes that are actually biomarkers. This means it is difficult to reproduce the results of an association study [45]; even if the same tissue samples are given to two different labs for analysis, each lab may produce different biomarkers [95]. To alleviate these issues, some researchers will perform a follow-up checking study using qPCR to confirm that the genes implicated by the microarray study truly are differentially expressed. However, there is yet to be a consensus as to which genes require confirmation, and what defines confirmation [2, 73].

The community is still in the process of standardizing how qPCR data should be made publicly accessible [18]. We believe that if more researchers adopted the BBD framework, it would help push towards mandating the release of qPCR data in a way similar to what is done for microarray data. If qPCR data were made public, the analyses could be independently verified[7], and moreover this data could be used as testbeds to develop and analyze new BBD algorithms.

The second problem is that there are many ways to measure the difference in gene expression levels between the cases and the controls. There are several commonly used measures, each of which will catch different trends in the data, and they all claim to be measures of "differential expression" [22, 94]. The results of a study can be wildly different based on which statistic is chosen to measure differential expression [14]. For example, suppose our data for a particular gene has sample means for the cases and controls $\hat{\mu}_1$ and $\hat{\mu}_0$ respectively. We could measure the difference by the difference of the estimates $m_1 = \hat{\mu}_1 - \hat{\mu}_0$ or we could use their ratio $m_2 = \hat{\mu}_1/\hat{\mu}_0$. If the true values are $\mu_1 = 1$ and $\mu_0 = 0.01$ then based on m_1 it would seem there is no difference[8] but by m_2 it would seem there is a large difference. Without a biological rational to favour one of these measures *a priori*, it is highly tempting for researchers to retroactively pick the statistic under which their favourite genes receive high ranks. Clearly, we should define our association studies to prevent this from occurring.

Coupled with the ambiguity in defining the measure of differential expression is the problem of deciding which of the top ranked genes will be called "biomarkers".[9] There are two common approaches taken to decide which genes are biomarkers.

Top K The first approach is to simply declare the top K features to be biomarkers, *i.e.*, those with the K largest measures of differential expression. The problem with this approach is selecting the appropriate value for K. Statistically, it is difficult to associate the choice of K with any measure of quality of, or confidence in, the results (if K must be set *a priori*). Although we do know that, as a result of studies being underpowered, statistical variations

[7] Surprisingly, many gene expression papers are published with errors in the statistical analysis [31] – errors include misinterpreting statistical tests, and training on the testing data (in supervised learning scenarios). Perhaps the same could be said for results based on qPCR, but without the data it is difficult to tell.

[8] This value is below typical levels of observation noise.

[9] Fundamentally this is a very difficult problem, because we have not specified what determines if a gene is or is not a biomarker, the rewards for discovering them, nor the penalties for falsely identifying genes as biomarkers.

often cause the top K genes in two different studies on the same phenotype to be very different [35].

One of the motivations for reporting the top K genes is that they can be "verified" by looking up their relevance in previously published studies. Or they can be compared with known functional groups of genes from databases like Kyoto Encyclopedia of Genes and Genomes (KEGG) [50], Gene Ontology (GO) [4], Ingenuity Pathway Analysis[10] (IPA), the Database for Annotation Visualization and Integrated Discovery (DAVID) [42], *etc.* This kind of validation can be consider as the "biological plausibility" of the results. If assessing the biological plausibility is the goal of the study, we may question, "If we are only interested in finding what we already know, why bother doing the study at all?", as we could just datamine those databases. Also what does it mean if the biological relevance of the top K features is not already known? Arguably, the most interesting discovery would be to find genes that are only expressed with phenotype, *i.e.*, there are no co-morbidities affecting the expression level of the genes.

Hypothesis test The second approach is to use null hypothesis testing principles. In this approach, we compute the distribution of the summary statistic under the null hypothesis that the gene is not differentially expressed, *i.e.*, the distribution of expression levels is the same in both cases and controls. After the data has been collected, we compute the summary statistic for each gene, and those with statistics that are unlikely to be drawn from the distribution of the null hypothesis are considered biomarkers. We use the p-value to measure how unlikely the observed statistic is, where the p-value is the probability of seeing the observed statistic, or one more extreme, when taking a random draw from the null distribution. The standard assumptions for the null hypothesis are:

1. When conditioned on the class, $s \in \{1, 0\}$, the gene expression values of gene g_i are normally distributed, $\mathcal{N}\left(\mu_{i,s}, \sigma_{i,s}^2\right)$.

[10] `www.qiagen.com/ingenuity`

2. The conditional means are the same, $\mu_{i,1} = \mu_{i,0}$.

The benefit of adopting this approach is that there is a rich literature in statistics that we can use to make statements about the quality of the genes reported as biomarkers. Specifically, we can quantify the number of genes we have reported as biomarkers, but expect not to be biomarkers, *i.e.*, amongst those genes identified as biomarkers, we can quantify the number of genes for which we expect that the null hypothesis is true.

We believe that the second assumption does not hold for any genes. Therefore given enough data, we will confidently determine all genes are biomarkers. However, in the statistically underpowered scenarios we actually encounter in real association studies this is not likely to be an issue. In other words, in practice we expect that because of the small sample size it is more likely that we fail to identify genes that are biomarkers than we falsely report genes that are not biomarkers.

We also note there are others that use a naive bayes approach to to detecting biomarkers [34, 77]. In the naive bayes process, we fit a mixture model, where each gene is either a biomarker or not biomarker. Then select the genes that strongly fit the biomarker model to be biomarkers. However, it has been shown that this is equivalent to using a False Discovery Rate (FDR) controlled t-test, *i.e.*, naive bayes is a variation of the hypothesis testing approach [34]. Controlling the FDR at level α, means that we guarantee that the null hypothesis is true for at most α percent of the genes identified as biomarkers.

Thus, the biggest issue with association studies is that they fail to explicitly define what makes a gene a biomarker. Also needed are rewards and penalties for (mis)reporting genes as biomarkers. Without rewards and penalties, we can have some very strange results, *e.g.*, declaring all genes as biomarkers is only a bad idea if there is a penalty for falsely claiming a gene is a biomarker when it is not. By asking the right questions of the data, very relevant information can still be discovered from statistically underpowered association studies. For example, [78] presents a sub-sampling study on a large microarray dataset, and shows that with reasonable sample sizes robust classifiers can be learned from the data to predict the sex of a patient. Here, the classifiers have a very specific performance measure (predictive accuracy measured on hold out data), and sex is a very incontrovertible[11] phenotype. We will take care to be equally objective when we define the BBD problem in Chapter 2.

[11] See footnote 7 in Section 4.4, on page 55.

Chapter 2

The Budgeted Biomarker Discovery Problem

In this thesis, we will explore the "Budgeted Biomarker Discovery" (BBD) problem. While biomarkers can be any type of feature, we will explicitly consider the case of genes as biomarkers. Focusing solely on genes allows us to be more concrete with our descriptions. It is straightforward to adopt our subsequent methods and analyses to apply to any type of biomarker; we will elaborate on this in Section 2.6.

Before we formally define our BBD problem, we first need to introduce some notations. We begin by assuming that we have a set of genes denoted by $\mathcal{G} = \{g_1, \ldots, g_N\}$. We use the proportion of RNA transcripts present in a sample as a measure of the gene expression level of the patient. When conditioned on the binary phenotype, $s \in \{1, 0\}$, the class specific mean and standard deviations for the expression values of gene g_i are denoted as $\mu_{i,s}$ and $\sigma_{i,s}$ respectively. We will assess genes based on their standardized effect size,

$$\Delta_i = \frac{\mu_{i,1} - \mu_{i,0}}{\sqrt{\left(\sigma_{i,1}^2 + \sigma_{i,0}^2\right)/2}} \quad . \tag{2.1}$$

Genes with positive Δ_i are said to be up-regulated with the phenotype and those with negative Δ_i are said to be down-regulated. Our goal is to find the set of genes, R, that are relevant, *i.e.*, either strongly up-regulated, or strongly down-regulated,

$$R = \{g_i : |\Delta_i| \geq \Delta^*\} \quad . \tag{2.2}$$

The terms "relevant gene" and "biomarker" are interchangeable within the BBD framework. However, we prefer to use the term "relevant gene" going forward as it is explictly defined in Equation (2.2), whereas "biomarker" may be confused with a previous definition (outside of this thesis) [69].

Here we have used Δ^* as a threshold for the minimal effect size genes that are relevant. For now, we can consider this as some fixed value that has been given to us as a definition. Later, in Section 2.2 we will discuss how to properly set this value in general.

2.1 Plate Model

As previously discussed in Section 1.3, in BBD problems we will make use of both high- and low-throughput technologies. In general we assume that we have a collection of possible tests, $\mathcal{T}$. Each test, $t \in \mathcal{T}$, will cover a specific subset of genes, $\mathcal{G}_t \subseteq \mathcal{G}$, for a cost, C_t. For example, one test is a microarray that measures expression levels of all genes in a patient for cost C_{array}, another test is a qPCR array that measures $N_{PCR} = 100$ genes associated with breast cancer for cost C_{PCR}, and another test is a custom qPCR array that measures $N_{PCR} = 100$ genes for our choice for cost C_{custom}.

In Section 1.3 we showed that our tests will use different approaches to measuring the gene expression values, and thus our definition of the effect size in Equation (2.2) is slightly ambiguous because it does not specify to which test the values of $\mu_{i,s}$, and $\sigma^2_{i,s}$ correspond. However, as the effect size is invariant to scale and shift operations on the underlying distributions[1] we claim that the value of Δ_i will be comparable across different tests. For example, the effect size of a gene observed in microarray data is comparable to that observed in qPCR, even though both methods observe fundamentally different data. We make the assumption that all tests have

[1] A shift operation would add a constant bias to the $\mu_{i,s}$ terms (raising the overall expression values for the gene) that would be lost when we compute the difference in means. A scale operation would multiply the $\mu_{i,s}$ and $\sigma_{i,s}$ terms by a constant (scaling all the expression values by the same amount) that would be lost when we divide the mean difference by the average standard deviation.

the same effect size, *i.e.*,

$$\forall t \in \mathcal{T} : \forall g_i \in \mathcal{G}_t : \Delta_i = \Delta_i^{(t)} = \frac{\mu_{i,1}^{(t)} - \mu_{i,0}^{(t)}}{\sqrt{\left(\sigma_{i,1}^{(t)2} + \sigma_{i,0}^{(t)2}\right)/2}} \quad , \tag{2.3}$$

where we have added the superscript (t) to denote the quantities specific to the test t. In general, we know that this assumption is not true, but it allows us to define relevance as an inherent property of the gene, *i.e.*, a gene is relevant regardless of which technology we use to observe its values.[2]

We use $\psi_{i,s,m}^{(t)}$ to denote the m'th gene expression value, of patients from class s, of gene g_i, using test t. We model these expression values as random draws from the distribution $f_{\psi_{i,s}^{(t)}}\left(\psi\,; s, \theta_i^{(t)}\right)$, where $\theta_i^{(t)}$ is a tuple of parameters, specific to gene g_i, and test t. For example, if expression values are normally distributed, and $\Delta_i = 1$, then a possible parameterization would be,

$$\begin{aligned}
\theta_i^{(t)} &= \left(\mu_{i,1}^{(t)} = 1, \sigma_{i,1}^{(t)2} = 1, \mu_{i,0}^{(t)} = 0, \sigma_{i,0}^{(t)2} = 1\right) \\
\psi_{i,1,m}^{(t)} &\sim f_{\psi_{i,1}^{(t)}}\left(\psi\,; 1, \theta_i^{(t)}\right) = \mathcal{N}(1,1) \\
\psi_{i,0,m}^{(t)} &\sim f_{\psi_{i,0}^{(t)}}\left(\psi\,; 0, \theta_i^{(t)}\right) = \mathcal{N}(0,1) \quad .
\end{aligned}$$

Figure 2.1 presents a plate model that summarizes our model of gene expression. Plate models are convenient tools for visualizing data with repeated structure [55, Chapter 6.4.1]. The idea is to view the repeated structure as a stack of identical plates. The plates are distinct, but have the same descriptions so it is sufficient to describe the top plate, and apply all statements to the plates below it.

From the top down, our plate model has a stack of plates corresponding to genes. The plate for gene g_i contains the true effect size of the gene, Δ_i, which we model as a random variable drawn from the distribution f_Δ, an indicator variable to denote if the gene is relevant, and a stack of plates corresponding to the tests used to observe that gene. The plate for each test, t, contains the distributions for the gene expression values using that test (which is encoded by $\theta_i^{(t)}$)[3], and two stacks of

[2] In practice, we expect that qPCR will be more accurate than microarray, and thus $\forall g_i \in R$: $\Delta_i^{(microarray)} \leq \Delta_i^{(qPCR)}$.

[3] The values of $\theta_i^{(t)}$ are constrained according to our assumption that all tests have the same effect size, Equation (2.3).

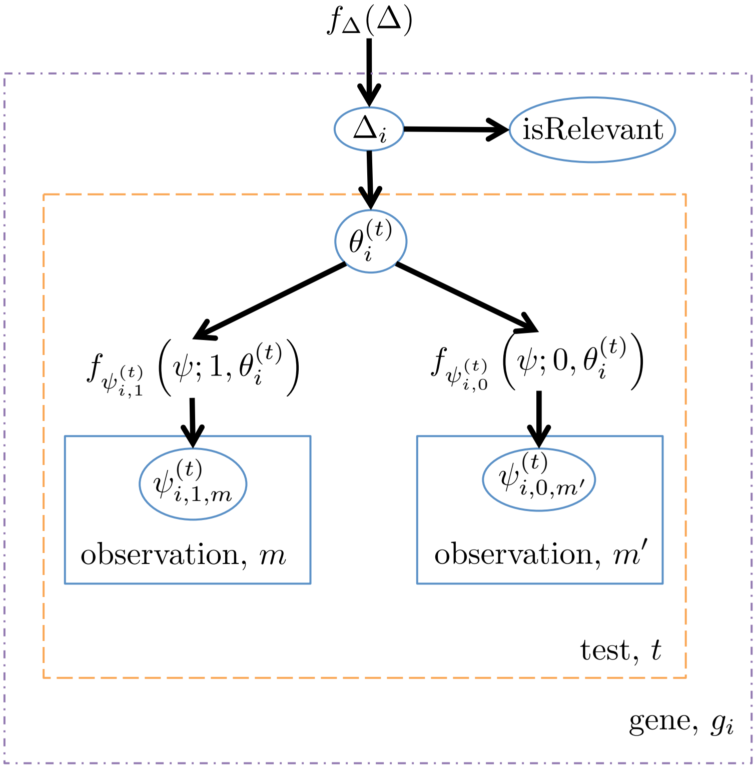

Figure 2.1: Plate model representing how the gene expression data is distributed. The dash-dot rectangle encloses all the variables and observations pertaining to gene g_i. The dashed rectangle encloses all the variables associated with test t on gene g_i. The solid rectangles enclose all the actual expression values observed (for each class). The isRelevant variable denotes an indicator of the relevance of the gene, *i.e.*, $g_i \in R \rightarrow \text{isRelevant} = 1$.

plates corresponding to the observed gene expression values, $\psi_{i,s,m}^{(t)}$, one stack per class, s.

The arrows in the plate model represent the natural flow of information in the model, *i.e.*, knowing the term at the tail of the arrow it is straightforward to reason about the term at the head of the arrow. Unfortunately, in BBD problems we can only observe the expression values, $\psi_{i,s,m}^{(t)}$, at the bottom of the model – those are what we see when collecting data. Thus, we must use the expression values to make inferences about the unknown distributions, $f_{\psi_{i,s}^{(t)}}$, which we then use to infer the unknown value of Δ_i, and ultimately make a prediction about the relevance of the gene. Using $M_s^{(t)}$ to denote the number of observations collected, from class s, of test t, we compute the following estimates of the unknown model parameters,

$$\begin{aligned}
\hat{\mu}_{i,s}^{(t)} &= \frac{1}{M_s^{(t)}} \sum_{m=1}^{M_s^{(t)}} \psi_{i,s,m}^{(t)} \\
\hat{\sigma}_{i,s}^{(t)2} &= \frac{1}{M_s^{(t)} - 1} \sum_{m=1}^{M_s^{(t)}} \left(\psi_{i,s,m}^{(t)} - \hat{\mu}_{i,s}^{(t)} \right)^2 \\
\hat{\Delta}_i^{(t)} &= \frac{\hat{\mu}_{i,1}^{(t)} - \hat{\mu}_{i,0}^{(t)}}{\sqrt{\left(\hat{\sigma}_{i,1}^{(t)2} + \hat{\sigma}_{i,0}^{(t)2} \right) / 2}} \\
\hat{\Delta}_i &= \frac{\sum_{t \in \mathcal{T}} \left(M_1^{(t)} + M_0^{(t)} \right) \hat{\Delta}_i^{(t)}}{\sum_{t \in \mathcal{T}} \left(M_1^{(t)} + M_0^{(t)} \right)} \quad .
\end{aligned} \tag{2.4}$$

Another useful property of plate models is that they are generative, which means that we can use the model to generate realistic synthetic data. If we know the distribution of the effect sizes, f_Δ, we can draw N values, $\{\Delta_i\}_{i=1}^N$, from it to create a hypothetical set of genes. Then by assuming a parametric form of the distributions $f_{\psi_{i,s}^{(t)}}\left(\psi\ ; s, \theta_i^{(t)}\right)$ we can draw synthetic data to represent the observations.

In Chapter 4 we will show how to exploit high-throughput data to produce a good estimate of the distribution of effect sizes, $\hat{f}_\Delta \approx f_\Delta$, by assuming that the $f_{\psi_{i,s}^{(t)}}\left(\psi\ ; s, \theta_i^{(t)}\right)$ are normal distributions. In other words we assumed,

$$\forall t : \theta_i^{(t)} = \left(\mu_{i,1}^{(t)} = \Delta_i, \sigma_{i,1}^{(t)2} = 1, \mu_{i,0}^{(t)} = 0, \sigma_{i,0}^{(t)2} = 1 \right) \quad .$$

In Chapter 5 will show how BBD algorithms can use the the estimated distribution, $\hat{f}_\Delta$, to tune their internal parameters by simulating their performance on synthetic data.

2.2 Setting an Appropriate Definition of Relevance

The previous section presented a new definition of biomarkers based on the relevance of genes as measured in Equation (2.2), and presented a plate model from which we can simulate data to evaluate and tune algorithms designed to find the relevant genes, but it remains unclear how to set our threshold on effect sizes, Δ^*, to define the relevant genes. Here we seek for an intuitive manner that we can use to set Δ^*.

We posit that a natural approach to setting Δ^* is to consider the performance of a case versus control classifier induced by a gene. Biomarkers should be able to separate the cases from the controls at a reasonably high accuracy. Keeping our assumption of normally distributed expression values, and assuming we know the parameters of the class specific distributions, $\theta_i^{(t)}$, we can construct a simple classifier that predicts the unknown phenotype of a patient, s, based on the observed expression value, $\psi_i^{(t)}$, of gene g_i,

$$\hat{s} = \arg\max_{s \in \{1,0\}} \left\{ f_{\psi_{i,s}^{(t)}} \left(\psi_i^{(t)} \, ; s, \theta_i^{(t)} \right) \right\} \quad .$$

Further assuming a 50/50 case/control distribution, and a common variance in both classes, $\sigma_{i,1}^{(t)2} = \sigma_{i,0}^{(t)2}$, the accuracy of this classifier is then,

$$\text{accuracy}(\Delta_i) = \int_{-\infty}^{|\Delta_i|} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx \quad . \tag{2.5}$$

Given an understanding of the biology associated with the phenotype under study, it should be reasonable for a biologist to state a minimal accuracy for a gene to be relevant. For example, in a study to find genes that are relevant when identifying the tumour type for women with breast cancers, we may set a high accuracy (perhaps 90%) because we know that the estrogen receptor proteins are strongly related to breast cancers [26]. However, if our goal is to find genes that are relevant when comparing stage II tumours to stage III tumours, we may use a lower threshold (perhaps 60%), as we expect that it would be difficult to determine the stage of a tumour based on gene expression values alone.

Once we have established a desired level of accuracy, it is a straightforward exercise to use Equation (2.5) to back-calculate the appropriate Δ^* to define the set of relevant genes in Equation (2.2). Figure 2.2 shows Δ^* as a function of the desired accuracy.

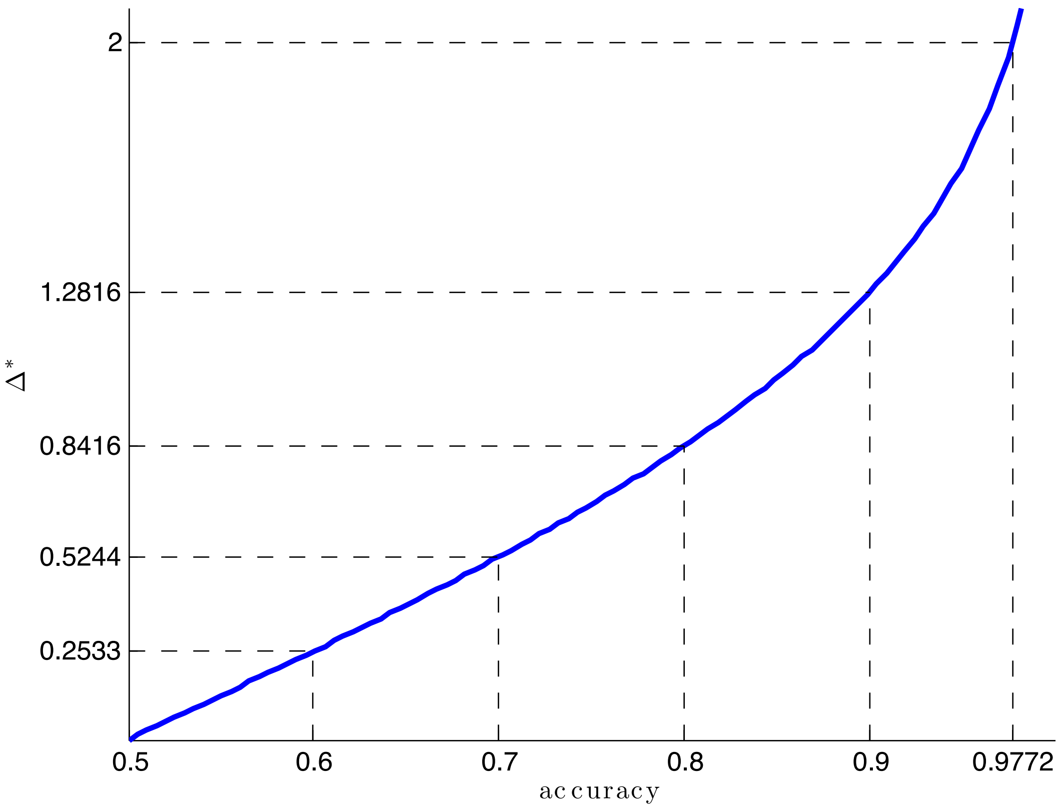

Figure 2.2: The relationship between the accuracy of a gene as a univariate classifier, and the Δ^* used to determine which genes are relevant.

2.3 Data Collection Model

In practice there are a variety of tests that can be used to collect gene expression data; different tests will have different costs. As the name implies, the BBD problem will incorporate a budget requirement, such that we spend at most a fixed budget, B, while collecting data. In other words, knowing the budget, a BBD algorithm should select and perform the most cost effective tests to discover the relevant genes.

Unfortunately, when we start a new study we have no idea how effective a particular test, t, will be at discovering relevant genes, as we have not yet collected any data. Thus, we consider a sequential data collection model wherein the data is analyzed as soon as it is collected, and thus can be used to determine which test, t, should be performed next. In practice we expect that as data is collected, many genes will appear to be obviously irrelevant, some may appear to be extremely relevant, and others will be difficult to assess. In other words, we can clearly identify many genes that are definitely are either in, or not in, the set R, defined by Equation (2.2). We believe that an intelligent algorithm should notice this and focus on collecting data for those genes that are on the borderline of being labelled relevant. In other words, there is little utility in collecting additional data for genes that we have already been strongly convinced are either relevant or irrelevant because it is unlikely that the data will change our decisions. For example, in a study on a new disease, an algorithm may begin by collecting a few microarrays to get a general sense of the data, but then notice that there are many genes on the apoptosis pathway that are borderline relevant, and so decide is more cost efficient to switch to using qPCR arrays for apoptosis. In other more general cases, the most cost effective solution may be to collect data from a variety of different qPCR arrays (including custom ones) that collectively include the genes that have been implicated by the initial microarrays.

2.4 BBD Definition

Formally, the BBD problem is to spend an experimentation budget, B, by selecting tests from $\mathcal{T}$ to collect data on the genes in $\mathcal{G}$ and return an estimate of the set of relevant genes $\hat{R} \subseteq \mathcal{G}$. Each test, $t \in \mathcal{T}$, will provide a measurement of the expression values for a specific subset of the genes, $\mathcal{G}_t$, at a specific cost, C_t.[4]

In order to evaluate the performance of algorithms, or objectively compare the results of different studies, we require an evaluation function that accepts as inputs the estimated set of relevant genes, $\hat{R}$ and the true set of relevant genes, R, and returns a number measuring their similarity, with higher scores being preferable. Ideally we desire $\hat{R}$ to have both high precision and recall, where precision means that genes in $\hat{R}$ are also in R, and recall means that genes in R are found in $\hat{R}$.

$$\text{precision} = \frac{\left|R \cap \hat{R}\right|}{\left|\hat{R}\right|}$$

$$\text{recall} = \frac{\left|R \cap \hat{R}\right|}{|R|}$$

To improve one of these measures, after the data has been collected, we must sacrifice the other. For example, we could get perfect recall by declaring all genes are relevant, $\hat{R} = \mathcal{G}$, but the precision would be poor, and reversely we could get perfect precision[5] by declaring no genes are relevant, $\hat{R} = \{\}$, but then we would have a recall of 0. As we require a single number to represent the similarity between R and $\hat{R}$, we propose the generalized F score as an appropriate evaluation function,

$$\text{evaluation}(R, \hat{R}) = \left(1 + \beta^2\right) \frac{\text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}} \quad . \tag{2.6}$$

We use the parameter $\beta \in \mathbb{R}$ to trade-off between precision and recall. Specifically, β allows us to weight the partial derivatives of the evaluation function.

$$\frac{\partial \text{evaluation}(R, \hat{R})}{\partial \text{recall}} \Bigg/ \frac{\partial \text{evaluation}(R, \hat{R})}{\partial \text{precision}} = \beta^2 \frac{\text{precision}^2}{\text{recall}^2}$$

Thus, β is the relative importance of recall over precision, *e.g.*, $\beta = 2$ means that we prefer recall twice as much as precision. In practice we expect β to be fairly low, perhaps $\beta \approx 1/10$, as it is much more important that we be precise when reporting relevant genes, than that we report all the relevant genes.

[4] This definition of the BBD problem is a generalized version of our previous presented definitions in [52, 53, 54].

[5] For this we should define $\frac{0}{0} = 1$.

2.5 Comparison to Traditional Association Studies

We now summarize the main differences between our BBD problem and traditional approaches to association studies.

1. In Section 1.4 we showed that a pitfall of association studies was that they lacked both a clear definition of what a biomarker was, and how much they valued their discovery. The BBD problem has formally defined biomarkers to be the relevant genes, *i.e.*, those that can separate the cases from the controls to a high degree of accuracy, where the user specifies accuracy. The BBD framework also provides the function, evaluation$(R, \hat{R})$, which sets an objective goal for the study – to get a high evaluation. The immediate benefits of this framework are:

    (a) Because there is a clear objective to the study, there is no uncertainty with respect to choosing which set of genes should be reported as biomarkers. In order to do well, algorithms should report the set of genes that they believe would give the highest evaluation possible. [6]

    (b) This framework helps to remove the perception of conflict that occurs when multiple studies discover different sets of genes on the same phenotype. For example, if two studies on the same phenotype discovered different sets of genes, $\hat{R}_1$ and $\hat{R}_2$, we would expect them to be different based on the statistical arguments of [35], but if they followed the same experimental and analytical procedures then we would expect that the quality of their sets be similar, *i.e.*, evaluation$(R, \hat{R}_1) \approx$ evaluation$(R, \hat{R}_2)$.

2. Our definition of relevance based on the effect size is very similar to previous approaches using t-tests – Equation (2.1) is just a scaled t-statistic. The key distinction is that for gene g_i to be a biomarker the BBD framework requires $|\Delta_i| \geq \Delta^*$ for a gene g_i to be a biomarker, whereas the t-test only requires

[6] Note that an algorithm cannot directly compute its evaluation, as it does not know the genes that are truly relevant, R. We will show that effective algorithms will find a good approximation for the evaluation.

$\Delta_i \neq 0$. We believe that in reality there will be a continuum of Δ_i values, and no genes will have $\Delta_i = 0$.

3. Traditional association studies focus only on microarrays, or in general high-throughput data. Thus, they require a follow up checking study in qPCR (or other low-throughput data) to confirm that specific genes are biomarkers. Unfortunately there is no clear consensus on how to properly perform an association plus checking study [2, 73]. Our BBD framework clearly merges these concepts into a single study, and considers how to effectively spend the experimentation budget – the best strategy is to combine microarrays and qPCR so that the evaluation function is maximized.

2.6 Extensions to the BBD Model

We now describe how our BBD framework can be generalized to cover a broader range of problems.

While Figure 1.1 showed that microarray studies are increasing in popularity we acknowledge that they are soon to be replaced by Next Generation Sequencing (NGS) methods [44]. To update our model, we can simply add these new technologies to the set of available tests, $\mathcal{T}$, as they come out. All of our algorithms and analyses will hold for the new technologies.

Outside of gene expression data, it is relatively straight forward to adopt our plate model to cover other types of 'omics data. For example, in metabolomics there are different techniques that can be used to collect a metabolomic profile (*i.e.*, concentration levels of all metabolites in a sample), each with its own accuracy and cost [27]. Furthermore, some metabolites will have concentration levels that are below the noise thresholds of certain technologies, and thus we see a similarity to the use of different qPCR arrays to cover all genes. By mapping the appropriate metabolomic parameters into our plate model we can identify metabolites by posing a BBD problem.

We can also extend our model to cover problems in genomics, where the goal is to find loci in our DNA where the allele can be linked to a specific phenotype.

This is currently done by multistage Single-Nucleotide Polymorphism (SNP) studies [81]. In a multistage study, the first stage uses a SNP array that observes $O(1\ 000\ 000)$ SNPs to identify candidate SNPs. Each subsequent stage then refines the number of candidate SNPs by using SNP arrays that are at least an order of magnitude smaller than those in the previous stage. Thus, when the study is completed the reported SNPs will have been selected out of a very large candidate pool, and have been checked by multiple SNP arrays. This multistage framework is very similar to the idea of switching from high- to low-throughput technologies in the BBD problem. The key distinction between SNPs and gene expression is that SNPs are discrete random variables whereas gene expression and metabolites are real numbers. We anticipate that our analysis would confirm the standard practice of running comprehensive SNP arrays for a small number of subjects , then running small arrays for much larger sets of subjects, *etc*.

Beyond bioinformatics, we can also use our BBD model to describe other problems in machine learning. Crowd-sourcing on sites like Amazon's Mechanical Turk[7] has become a very popular way to get menial tasks done. But before assigning work to a "turker" or group of "turkers" an employer may wish to ensure that they have some minimal level of proficiency at the task. For example, a company with a large collection of images to label may create a small test set of expertly labelled images which they then pay the potential turkers to re-produce those labels. Turkers that do well on the test set can be immediately employed to begin labelling the rest of the data. Turkers that perform terribly on the test set can be discarded. Turkers with borderline performance can be re-tested by constructing a second test set, to decide whether or not to hire them. The task of identifying good turkers is analogous to finding relevant genes in the BBD problem, with the different test datasets corresponding to the high- and low-throughput observations across the turkers.

[7] `https://www.mturk.com`

Chapter 3

Related Works

3.1 Similar Problems in Bioinformatics

3.1.1 Checking Studies

The microarray community has mandated that data be made public as a prerequisite for publishing association studies [9]. They have even set guidelines mandating the minimal amount of information to be included when releasing the data – enough that others can replicate the analyses [15]. Unfortunately, they have not set a standard for validating the results of an association study with qPCR in a checking study [21, 73]. Regrettably, most studies only release the microarray data, and give only summary statistics of the qPCR data (if it exists). It would seem that the community has yet to see the same value in qPCR data that it has with microarray data. For example, the GEO database stores many properties in relation to each microarray dataset posted, including associated papers and various high level summaries of the data, but it does not contain links to, or information about, the associated qPCR data – it does not even have an indicator for the existence of associated qPCR data.

Hopefully this will change in the future, as there has been a move to begin standardizing qPCR data in a way similar to microarrays [18]. If qPCR data were made public the analyses could be independently verified, and the data could be used as testbeds to develop and analyze new BBD algorithms.

3.1.2 Set Enrichment

As an alternative to searching for individual biomarkers in an association study, there is also a large amount of research focused on finding sets of features that are differentially expressed with the phenotype [42, 43, 79]. One of the motivations for such set analyses is that individual features may not be strongly differentially expressed in a study due to statistical noise, but if the set contains functionally relevant features that are related to the phenotype, then it is unlikely that none of those features will appear as biomarkers. In other words, it is highly likely that for at least some of the features of important sets will be identified as biomarkers. Thus, it is more likely that similar studies will agree on the feature sets that are important even if they disagree on which individual features are biomarkers. For example, if two studies reported genes from the apoptosis pathway as biomarkers, even if the the genes reported are different, we might infer that apoptosis is related to the phenotype.

Gene set analyses also employ the null hypothesis testing approach of association studies, but rather than testing a statistic about a particular gene, they employ a summary statistic that measures expression levels of multiple genes. By far the most common method for scoring a feature set is the Gene Set Enrichment Analysis algorithm (GSEA) [79]. GSEA is very similar to the Kolmogorov-Smirnov test used to check if an observed probability distribution is different from a reference one. In GSEA the reference distribution is the distribution of correlation scores measured for every feature with the phenotype. The enrichment score is a distance measure of the distribution of the correlation scores of the features within the set to the reference distribution. The distance measure will be high if the set contains features with correlation scores found at the tails of the reference distribution, *i.e.*, the set contains genes that either highly correlated with the phenotype, or those that are highly anti-correlated with the phenotype. Note, this score does not necessarily mean that all features within the set are correlated with each other, *i.e.*, the genes in the set need not be biologically related to get a good score.

Analyzing sets by a summary statistic falls into the same trappings we have already discussed with association studies in Section 1.4 – we lack a formal definition of what it means for a set to be “differentially expressed” and an objective evaluation criteria comparing the sets reported to be differentially expressed, to those that truly are. Also, another important issue is to find appropriate, pre-defined, sets to analyze. Common choices are to use known pathways from databases like KEGG [50], or ontologies and hierarchies from GO [4], *etc*. Note these are some of the same sources that are being used for biological interpretations of discovered biomarkers. Thus, we argue that turning to set analysis has really just complicated the original problem rather than simplifying it.

3.1.3 Class Prediction

An alternative method to finding biomarkers is to build a simple classifier to predict the phenotype. If the classifier works well then it must then have found informative features, and by definition[1] those features should be called biomarkers. While there is an extensive literature on learning classifiers for high-dimensional data, only a handful of the methods have been successfully adopted by the bioinformatics community.

The Least Absolute Shrinkage and Selection Operator (LASSO) [82], Predictive Analysis for Microarrays (PAM) [83], and Significance Analysis of Microarrays (SAM) [87] are very popular algorithms that are typically applied to microarray data. The main idea in these approaches is to use regularization methods that penalize the classifier for every gene that it includes in its model. As a result the classifier will opt for using a small set of highly informative genes, instead of using a large set of many somewhat informative genes. By working with a smaller set of genes the classifier will have less parameters to fit than had it used more genes, and given the limited amount of data, relative to the number of genes, it is more likely that those parameters can be tuned for good performance. In other words, we prefer classifiers that have fewer parameters to tune, because they will be less likely to “overfit” to the noise in the data [41].

The downside of employing sparsity inducing methods like these is that, while

[1] Here we mean the vague definition of biomarkers in general, and not our definition in the BBD framework.

there is strong evidence to suggest that the genes that used in the model are biomarkers, nothing can be said about the genes left out of the model. As the previous approaches operated based on inducing sparsity, they are limited in their ability to discover biomarkers, *i.e.*, by their design, these algorithms will have poor recall when discovering biomarkers. For example, if there are two genes that are very relevant, but are correlated with each other, the classifier will use one and ignore the other. Thus, by inspecting the genes used by the classifier we can miss some genes that are obviously biomarkers. This subtle point is the difference between learning a good classifier, and the problem of discovering biomarkers.

As an alternative to regularization methods, some researchers prefer to use the idea of recursive feature elimination [40]. Here we build a series of classifiers. The first classifier is given the freedom to use all the genes. Then the second classifier is given the freedom to use any of the genes, except for the most informative gene used in the first classifier, *i.e.*, the previously determined "best gene" is eliminated from the feature set. The third classifier is given the freedom to use any of the genes, except for the most informative gene in the previous two classifiers. Thus, each classifier is denied access to the most informative gene used by each of its predecessors. Note, the goal is *NOT* to make a good ensemble of classifiers to be used in a predictive task. The goal is to observe the order in which the genes are eliminated – genes eliminated earlier are more likely to be biomarkers. Thus, recursive feature elimination is similar to the top K approach discussed in Section 1.4.

Lastly, note that in the BBD framework, the genes are assessed based on their strength as a univariate classifier, whereas these methods assess the genes based on their strength within a multivariate classifier. These are potentially two different sets of genes.

3.1.4 Sample Size Calculations for Microarray Analysis

Many researchers are interested in knowing the sample size required to reliably analyze microarray data. Jung [48] and Muller *et al.* [64] investigate the problem of determining the sample size required to ensure that a pre-fixed amount of genes will pass an False Discovery Rate (FDR) controlled t-test for a given level of control α. However, [13, Section III] provides several strong arguments against adopting these and similar methods. We believe that a better approach would be to collect some preliminary microarray data and then apply our FDE algorithm to learn the plate model for the gene expression values. The plate model can then be used to determine the appropriate sample size for the study. This can be easily done by modifying our TNAS-FDR algorithm.[2]

A similar problem is to determine the sample size required to build robust classifiers [30, 70, 78]. Dobbin *et al.* [30] takes an analytical approach to modelling the statistical variations within microarray data, such that the sample size can be determined. Popovici *et al.* [70] and Stretch *et al.* [78] retro-analyze large microarray datasets by evaluating models built on sub-samples of the data. Strectch *et al.* [78] claims to be more objective than Popovici *et al.* [70] because they have used classification accuracy (measured by cross validation, and on external datasets) and sex as the phenotype, whereas Popovici *et al.* [70] use Area Under the receiver operating Curve (AUC) (measured by cross validation) and Estrogen Receptor (ER) status as the phenotype – sex is an objective phenotype (there is no ambiguity about the sex of a patient), and ER status is a subjective phenotype (experts may disagree on the ER status of a patient). We will use the sub-sampling approach and data from Stretch *et al.* [78] in our experiments.

3.1.5 Simulation Models for Microarray Data

Many researchers analyze microarray data to produce Gene Regulatory Networks (GRNs) [10, 62]. Unfortunately, there are no benchmarks to evaluate GRN algorithms on – as we do not know the ground truth. Thus, if different algorithms produce different GRNs on the same data, we cannot objectively determine which is better.

There is a sub-community of researchers that have been developing simulation models, wherein the ground truth is known, that produce data that seems representa-

[2] For a given sample size, n, TNAS-FDR computes the optimal level of FDR control, α^*, to apply, and thus by flipping the problem and fixing α, we can compute the value of n necessary to achieve a specific evaluation score after we have estimated $f_\triangle$.

tive of real microarray data [1, 66]; GRN algorithms can be evaluated and compared using these simulation models. This simulation modelling is similar to the problem of estimating the distribution of the true Δ values, f_Δ, which we solve as a subroutine within BBD. However, the key distinction between the two problems is that in estimating and using f_Δ we make no use of interactions between the genes (*i.e.*, we implictly assume genes are uncorrelated), and that is the entire point of these simulation models.

3.1.6 Individualized Treatment Rules

Clinical medicine often tries to identify which treatments are best for different patient sub-populations. For example, suppose there are two weight loss pills, A and B, and we wish to determine which pill is best for men and which is best for women. Here we hypothesize that there may be a difference in which treatment is more effective for each sex due to the differences in how men and women store fat. In other words, this problem is asking if the treatment has a differential effect across the patient population; this is strikingly similar to the concept of a biomarker being something that is differentially expressed patient populations. Furthermore, if we scale up the problem and consider a large pharmaceutical company with many drug candidates that it needs to screen for differential effects, in a budgeted manner, we can see an analogy to our BBD problem. This problem is referred to as the Individualized Treatment Rules (ITR) problem [7, 24, 25].

Despite being very closely related to the BBD problem, ITR is fundamentally different because the actions in ITR means giving a patient a treatment, which not only gives us information about the treatment but also influences the patient. Thus, the ITR goal is to not only find good treatments for different patient populations in a cost effective manner, but the goal is also to deliver good treatments to the patients. In BBD the evaluation only depends on the conclusions drawn after all the data has been collected.

3.2 Similar Problems in Computing Science

3.2.1 Sequential Probability Ratio Tests

The problem of testing if a *single* gene is relevant, assuming that we can collect the data sequentially, corresponds to the sequential hypothesis testing problem of collecting data until one of two hypotheses, $H0$ versus $H1$, can be decided at a pre-fixed confidence. The Sequential Probability Ratio Test (SPRT) [90] solves this problem optimally in the sense that provably no other algorithm can make the decision with the same confidence and collect less data in expectation.

When solving BBD problems it would be highly desirable to exploit this optimality result to reduce the amount of data collected. Unfortunately, it is very difficult to reduce BBD to the SPRT framework. For starters, it is not clear how to set appropriate distributions for the $H0$ and $H1$ hypotheses. One approach could be to pick values Δ_0 and Δ_1 for the $H0$ and $H1$ hypotheses and test which is a better fit for the estimated effect sized, $\hat{\Delta}$. Such a test would obviously be subject to how select Δ_0 and Δ_1.

Aside from the issue of setting up the hypothesis test, the SPRT framework also fails to model a selection criteria for deciding which tests to perform. For example, if we had two genes, g_1 and g_2, the SPRT framework has no criteria to determine if we should collect data for either g_1 at cost C_1, g_2 at cost C_2, or both at cost $C_{1\&2}$ – it can only determine, for each gene individually, if sufficient data has been collected to make the decisions at the pre-specified confidence.

3.2.2 Active Structure Learning

Active learning is a field of machine learning that explores the problem of learning a model when given access to both labelled and unlabelled data with the ability to obtain labels for the unlabelled data at a cost. For BBD problems, we consider that for all genes, for all tests, for all patients in our study there is a specific expression value $\psi^{(t)}_{i,s,m}$. If we have performed that test and observed that value then we consider that as labelled data, and if we have not then it is unlabelled data. The active learning problem is to use the labelled data to select which unlabelled data we should request to be labelled (within our budget) so that we can get a good evaluation score. There are many learning objectives and labelling mechanisms that have been previously studied [74]. Among the learning objectives, the closest to our BBD problem is active parameter learning for graphical models [84] and active structure learning for graphical models [59, 85]. We could use algorithms for active parameter learning to estimate the parameters in our plate model, which includes the Δ_i for all genes. Thus, enabling us to solve our BBD problem. However, solving this problem is much harder than our BBD problem, because along with identifying the relevant genes this solution will estimate the distribution of expression values for all tests for genes, *i.e.*, it must learn all the $\theta_i^{(t)}$ in our plate model. While it is possible to adapt an algorithm from this domain to solving BBD, note that these algorithms are designed with the goal of minimizing the KL divergence of the learned model to the true model, but BBD solutions are evaluated on the quality of their estimated relevant genes, $\hat{R}$. These are very different objectives. For example, an algorithm may have poor estimates of the distribution parameters, $\hat{\theta}_i^{(t)} \not\approx \theta_i^{(t)}$, but by aggregating across multiple tests, Equation (2.4) may produce a good estimate of the effect size $\hat{\Delta}_i \approx \Delta_i$. Thus, the learned model will have a high KL divergence to the true model, but the algorithm will score well on the BBD problem.

Alternatively we can pose a simpler active learning problem, wherein the goal is to learn a naive bayes classifier for the phenotype. In such a classifier [41, Chapter 6.6.3], we assume that given the phenotype the expression values of genes are independent of each other, and we only include the genes that are not independent of the phenotype. Active learning for such naive bayes models has been previously studied by [59]. After the algorithm has spent its budget, the biomarkers would then be those genes that are included in the classifier. The problem with this approach is that it totally ignores our definition of biomarkers in Equation (2.2) – any gene with $\Delta_i \neq 0$ will eventually be included in the model, and thus called a biomarker, if the budget is sufficiently large.

3.2.3 Bandit Problems

The n-arm bandit is a classic reinforcement learning problem [80]. This problem takes its name from its similarity to playing slot machines – slot machines are also known as one-arm bandits. Bandit algorithms are presented with several arms (*i.e.*, slot machines) that they must sequentially play, with the goal of maximizing the amount of money they win. Each arm is characterized by an unknown reward distribution, thus algorithms should play all arms to get an estimate of these distributions (explore), but should prefer to play those arms that have seemingly higher rewards more often (exploit). Relevant variants of this problem are the best arm identification [6], top K arm identification [17], and subset selection [49]. In these problems, the algorithm has an exploration phase where it can play the arms with the sole purpose of learning the reward distributions. Then once the exploration phase is over, the algorithm must identify: the best arm, the top K arms, or a set of good arms. If we consider each gene to be an arm with mean reward $|\Delta_i|$, then we can see a natural mapping between our BBD problem and these problems.

Unfortunately, the current algorithms for these problems are designed to work in sequential scenarios where the algorithm can only pull one arm at a time, and all arms have equal cost [17, 49]. The goal of the Successive Accepts and Rejects (SAR) algorithm [17] is: given a fixed number of pulls, identify the top K arms. It operates recursively, by pulling all arms equally often, until it can identify the extremes, *i.e.*, the arms that are clearly very good, or very bad, it then repeats on the unidentified arms. While it is possible to force a similar behaviour in a BBD algorithm, by having the algorithm use the set of tests such that all genes are tested (approximately) equally often, note that finding such a set of tests corresponds to solving the set cover problem, which is **NP**-complete [51].[3] Alternatively, we could use the Lower Upper Confidence Bound (LUCB) algorithm [49] which finds a set of K good arms[4] by iteratively partitioning the arms into those that appear to be in the top K and those that do not, and then pulling the arms that are closest to that

[3] While high-throughput methods could be used to observe all the genes, they will be come increasingly cost ineffective as the algorithm proceeds, and thus necessitate solving the set cover problem.

[4] Here good arms correspond to relevant genes.

decision boundary. In Chapter 5 our BBD-Greedy algorithm will adopt a similar behaviour.

There has been work in the bandit setting where the arms have different costs, and algorithms are allowed to observe the returns from multiple arms, but the focus there has been on minimizing the regret[5], rather than classifying the arms [3, 8, 29, 86].

3.2.4 Stochastic Submodular Maximization

A very reasonable solution to BBD problems would be to collect a few microarrays, and then make and execute a plan of which additional tests to collect, *i.e.*, once the plan is made the algorithm proceeds to follow that plan, and no longer uses the data to select which tests to collect. In making such a plan we would expect that every additional test gives us more information, but as we perform more tests, the information gained per test will decrease. This diminishing returns property is known as submodularity [33] and the study of maximizing problems of this nature has become quite popular in recent years.[6]

While most of this literature has focused on deterministic problems, there has been work on sensor placement that is stochastic [39]. Here the goal is to place temperature sensors across a room such that the room is fully covered by the sensors, but sensors can fail or be subject to varying amounts of noise. Thus, the proposed solution is to sequentially deploy batches of sensors, so that we may observe which fail or are obstructed, and adaptively plan accordingly. Golovin and Krause [39] proves that greedy algorithms do well on this, and similar problems. Our BBD problem shares this adaptive submodular property, where given the currently collected data one plan may look good, but as we execute that plan, a new plan may seem better, thus we should periodically revise the plan. Unfortunately we will show that greedy algorithms are not good solutions for BBD problems, as our experimental results in Section 6.2 show that our BBD1 algorithm outperforms our

[5] The regret of a bandit is a comparison of the money made while playing the slot machines, to how much money it could have made if it knew the true reward distributions.

[6] A comprehensive list of recent tutorials and workshops on submodular optimization is maintained by Andreas Krause at: `http://submodularity.org/`.

BBD-Greedy algorithm in terms of the evaluation score.

3.2.5 Density Estimation and De-Convolution

In Chapter 4 we present the problem of density estimation for the univariate statistics of high-throughput data. Our goal is to compute a good estimate of the true effect sizes of the genes, f_Δ. The problem of estimating a probability distribution from data has been well studied as the density estimation problem [76]. Unfortunately, this is a bit of a misrepresentation of our problem as we do not work on density estimation. Instead we make use of the Glivenko-Cantelli theorem [88], which states that the empirical estimate of the Cumulative Density Function (CDF), $\hat{F}_x$ converges uniformly to the true CDF, F_x, as the sample size used for the estimate increases, *i.e.*, $\sup_{x\in\mathbb{R}} |F_x(x) - \hat{F}_x(x)| \overset{a.s.}{\to} 0$. Thus, we can safely assume that the empirical estimate of the distribution of observed effect sizes, $\hat{\Delta}$ is equal to its true value, *i.e.*, $\hat{F}_{\hat{\Delta}} = F_{\hat{\Delta}}$, because the high-throughput nature of the data provides a large enough sample for the theorem to hold. Furthermore, in Chapter 4 we assume an additive noise model for $\hat{\Delta}$ where the noise distribution, f_ε is known. Thus, our problem of computing f_Δ can be posed as a de-convolution problem.

Convolution is an operation that takes two functions, $x(t)$ and $h(t)$, and combines them in a linear summation,

$$\begin{aligned} y(t) &= \int_{-\infty}^{\infty} h(t-\tau)x(t)d\tau \\ &= \left(h(t) * x(t)\right)(t) \quad . \end{aligned}$$

The convolution operation is invertible, *i.e.*, we can use the resulting function $y(t)$ in combination with $h(t)$ to recover $x(t)$. We call this inverse operation “de-convolution”, and it arises in many signal processing applications. For example, to play music a radio must solve the deconvolution problem of extracting the music signal $x(t)$, from the received signal at its antenna $y(t)$ in the presence of transmission noise $n(t)$, and unavoidable distortion from the antenna $h(t)$,

$$y(t) = \left(h(t) * x(t)\right)(t) + n(t) \quad .$$

When $h(t)$ is known and $n(t)$ is unknown, the problem is solved by the Weiner filter [93]; if $h(t)$ is unknown there is a vast literature of methods [72, Chapter 6].

By solved we mean that filters have been designed to transform signal $y(t)$ into $\hat{x}(t)$ such that the Mean Squared Error (MSE) between $x(t)$ and $\hat{x}(t)$ is minimized. Luckily, our particular de-convolution problem is somewhat easier than this general problem, as we need only de-convolve $f_{\hat{\Delta}} = (f_{\Delta} * f_{\varepsilon})$. This problem can be solved by applying the FFT/iFFT method [67, Chapter 9]. However, unlike our example of producing a music signal, where it is perfectly fine to minimize the MSE, we will be computing a distribution and are thus interested in analyzing the approximation noise of the algorithm, and their effects when the computed distribution is used in our plate model. As the goal of the de-convolution problem is to produce a MSE estimate of $x(t)$, this analysis is not done in the de-convolution literature.[7]

[7] Furthermore, de-convolutions are often used a pre-processing steps in classification problems, and thus, as long as the approximation noise is systematic the details of its effect are uninteresting. For example, in communication problems the goal is to determine if $x(t)$ represents a digital "1" or a digital "0".

Chapter 4

Density Estimation for High-Throughput Statistics

Before we present algorithms for solving the BBD problem we consider the problem of estimating the distribution of univariate statistics of high-throughput data. Our goal is to compute a good estimate of the distribution of the effect sizes, $\hat{f}_\Delta \approx f_\Delta$, to characterize our plate model from Section 2.1, thus allowing us to generate synthetic data for similar BBD problems that can be used to tune parameters for BBD algorithms. Throughout this section we will make frequent use of the '^' symbol to denote empirical estimates. Thus, for gene g_i the true effect size is Δ_i, its estimate is $\hat{\Delta}_i$, and f_Δ and $f_{\hat{\Delta}}$ are the respective distributions of the effect sizes across all genes. $\hat{f}_\Delta$ is our estimate of f_Δ.

In this chapter we will present two methods for estimating f_Δ from microarray data, and analyze their different strengths and weaknesses. We then present the Fused Density Estimation (FDE) algorithm that combines both methods.

In addition to parameter tuning for BBD algorithms, solving this particular density estimation problem is of interest in other tasks. For example:

1. In a traditional association study, where only microarray data is collected and the goal is to label biomarkers by using t-tests, or by listing the top K genes, we can use f_Δ to select appropriate control mechanisms for the t-test, or good values of K to use in the top K approaches. In fact this will be the motivating premise for our TNAS algorithms in Section 5.2.

2. We can ask budget related questions, *e.g.*, determine how many microarrays

are required to reliably find the top K genes, or some gene set $\mathcal{S}$ such that $|\mathcal{S}| = K$ and $\forall g_i \in \mathcal{S} : |\Delta_i| \geq \Delta^*$. This is similar to the work done on power analysis for microarray studies [92], but has the advantage of being done precisely for the study of interest instead of attempting to transfer results from other studies.

3. When analyzing gene set enrichment, knowing f_Δ we can assess the quality of a particular gene set by comparing the observed univariate statistics of the genes in the set versus those that would be observed in a random draw from f_Δ. Note the highly popular GSEA algorithm [79] uses a bootstrap approach to estimate the distribution of correlation scores of each gene with the phenotype. Using our FDE algorithm we can get a better estimate of this distribution. In Section 4.4 we show that, because our FDE algorithm corrects for the observation noise in the observed statistics, it outperforms the empirical estimator that does not model the noise.[1]

As a point of clarity, we note that others have developed methods of estimating the density of univariate statistics of microarray data as part of empirical bayes approaches for labelling biomarkers [34, 77]. However, the goal in those works was to do mixture modelling, where they learned (or assumed) a distribution for the statistics of the genes that are not biomarkers, and another for the genes that are. In other words, those methods are learning a good way to compute $\hat{f}_\Delta$ on two sets of genes. This is different from our goal of computing $\hat{f}_\Delta$ for all genes.

Throughout this chapter we will assume that all data has been collected using a single high-throughput technology, presumably microarrays, that simultaneously measures the expression values for all genes in $\mathcal{G}$. Thus, for simplicity we omit the use of superscripts to denote the test for which the parameters in this section are specific.

To be precise about our learning objective here: recall that we assess each gene,

[1] After applying the Fisher z-transform [37, 38] to the correlation scores, we can fit the resulting statistics to the additive noise model, Equation 4.3, that our FDE algorithm exploits to correct for the observation noise.

g_i, based on the standardized effect size,

$$\Delta_i = \frac{\mu_{i,1} - \mu_{i,0}}{\sqrt{\left(\sigma_{i,1}^2 + \sigma_{i,0}^2\right)/2}} \quad . \tag{2.1 revisited}$$

We seek to estimate the distribution of the these values, f_Δ, *i.e.*, we want a distribution such that we can view the set of all effect sizes, $\{\Delta_i\}$, as a set of N random variables drawn from f_Δ. However, we only have access to the unbiased estimates,

$$\hat{\Delta}_i = \frac{\hat{\mu}_{i,1} - \hat{\mu}_{i,0}}{\sqrt{\left(\hat{\sigma}_{i,1}^2 + \hat{\sigma}_{i,0}^2\right)/2}} \quad . \tag{4.1}$$

4.1 Raw Empirical Density Estimate

The obvious, naive, approach to estimating f_Δ is to use the distribution of the observed estimates, easily calculated via its Cumulative Density Function (CDF),

$$\begin{aligned} \hat{F}_{\hat{\Delta}}(x) &= \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left(\hat{\Delta}_i \le x\right) \\ \hat{f}_{\hat{\Delta}}(x) &= \frac{\partial}{\partial x}\hat{F}_{\hat{\Delta}}(x) \approx f_\Delta(x) \quad . \end{aligned} \tag{4.2}$$

The Glivenko-Cantelli Theorem [88] tells us that for large N, which we have with microarrays, this empirical estimate converges to the distribution of the observed effects, $\hat{f}_{\hat{\Delta}} = f_{\hat{\Delta}}$.[2] However, we question how reasonable it is to approximate $f_\Delta \approx f_{\hat{\Delta}}$. The issue is that this estimate does not account for the observation noise present in the estimated effect sizes, $\{\hat{\Delta}_i\}$.

In Appendix 8.2 we will show that, given our assumption that the gene expression values are normally distributed, the estimate $\hat{\Delta}_i$ follows a scaled non-central t-distribution. Rather than working directly with the exact distribution of $\hat{\Delta}_i$, we consider an approximation to an additive noise model of the form,

$$\begin{aligned} \hat{\Delta}_i &\approx \Delta_i + \varepsilon_i \\ \varepsilon_i &\sim \mathcal{N}\left(0, \frac{2}{n}\right) \quad . \end{aligned} \tag{4.3}$$

[2] Specifically the theorem tells us that $\sup_{x\in\mathbb{R}} \left|F_{\hat{\Delta}}(x) - \hat{F}_{\hat{\Delta}}(x)\right| \overset{a.s.}{\to} 0$ as N increases, which means that we have a good estimate of the distribution for all values of x.

The benefit of this approximation is that now, the distribution of the estimates is given by the convolution of the distribution of the true effect sizes with the additive noise distribution [68, Equation 7.7],

$$\begin{aligned} f_{\hat{\Delta}}\left(x\right) &= \int_{-\infty}^{\infty} f_{\Delta}\left(\tau\right) f_{\varepsilon}\left(x-\tau\right) d\tau \\ &= \left(f_{\Delta} * f_{\varepsilon}\right)\left(x\right) \quad . \end{aligned} \tag{4.4}$$

The convolution in Equation (4.4) will have a "smoothing" (*i.e.*, flattening) effect on the true distribution. To illustrate the effect of the smoothing, Figure 4.1 shows the effect of adding noise from a normal distribution to values drawn from a Laplace distribution, *i.e.*, $f_{\Delta} = \text{Laplace}\left(0,1\right)$ and $f_{\varepsilon} = \mathcal{N}\left(0,1\right)$. We pick the Laplace distribution for this example as it is notably peaked at $x = 0$, but after the noise has been added, the peak is noticeably less sharp, as probability mass from the peak has been pushed into the tails. If we were to use this distribution (*i.e.*, $f_{\hat{\Delta}}$) as the true distribution (rather than f_{Δ}), it would lead us to optimistically believe that many more genes have large statistics, *i.e.*, we would believe that it is easier to find relevant genes because there are seemingly more of them.

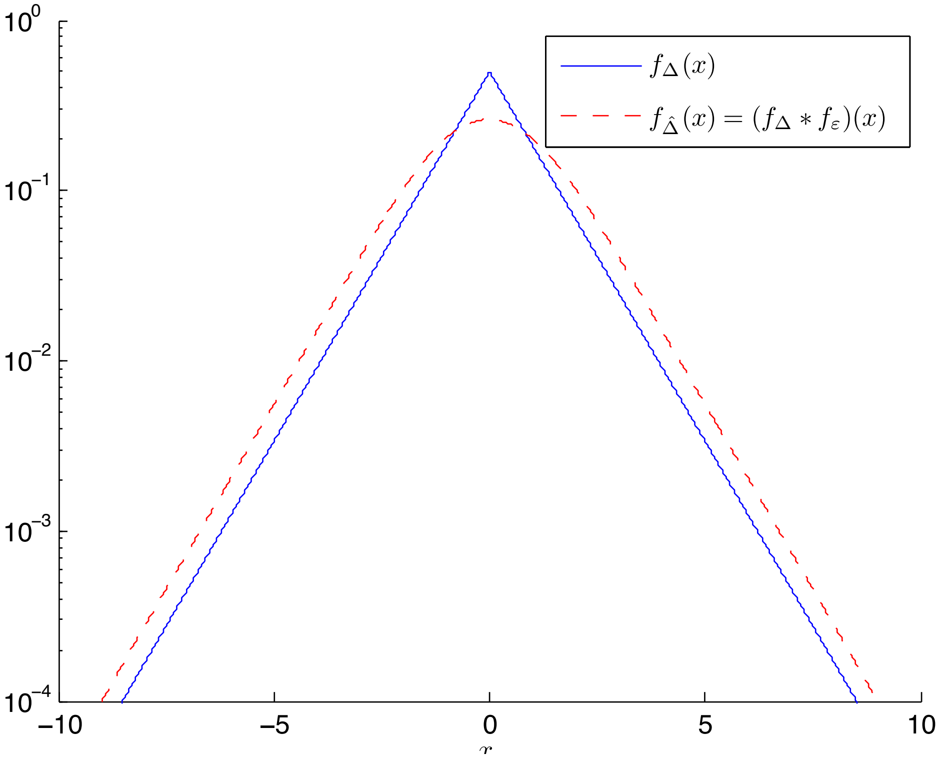

Figure 4.1: The effect of convolving a noise distribution, $f_\varepsilon = \mathcal{N}(0,1)$, with the true distribution, $f_\Delta = \text{Laplace}\,(0,1)$.

4.2 Density Estimation via Characteristic Functions

Given our assumption that we know f_ε, it would seem natural to try to undo the convolution operation in Equation (4.4) to recover f_Δ. In some special cases, when $f_{\hat{\Delta}}$ and f_ε belong to certain distributional families, we can analytically perform the deconvolution. However, in general scenarios, analytic solutions will not be tractable and we must resort to numerical integration. Unfortunately, numerical methods will introduce approximation noise that prevents us from properly recovering the distribution, but by understanding and modelling that noise we can construct algorithms that are well-suited to the task of estimating f_Δ.

The trick to understanding the noise is to consider the characteristic function representation of the distributions. Using $j = \sqrt{-1}$, the transformation from Probability Density Function (PDF) to characteristic function and its inversion are given by Equations (4.5) and (4.6).

$$\Phi_x(t) = \int_{-\infty}^{\infty} e^{jxt} f_x(x)\, dx \tag{4.5}$$

$$f_x(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-jxt} \Phi_x(t)\, dt \tag{4.6}$$

The characteristic function notation is very convenient when working with sums of random variables.

Theorem 1 (Product of characteristic functions [68, Equation 7.85])**.** *If $\hat{\Delta} = \Delta + \varepsilon$, where Δ and ε are independent random variables with characteristic functions Φ_Δ and Φ_ε respectively, then the characteristic function of $\hat{\Delta}$ is given by the product,*

$$\Phi_{\hat{\Delta}}(t) = \Phi_\Delta(t)\, \Phi_\varepsilon(t) \quad . \tag{4.7}$$

Theorem 2 (Characteristic functions of symmetric distributions)**.** *If $f_x(x) = f_x(-x)$ then,*

$$\Phi_x(t) = \Phi_x(-t) \tag{4.8}$$

$$\Phi_x(t) \in \mathbb{R} \quad . \tag{4.9}$$

From our additive noise model in Equation (4.3), the characteristic function of the noise is $\Phi_\varepsilon(t) = e^{-t^2/n}$, where n is the number of microarrays (per class). We approximate f_Δ by assuming that f_Δ is symmetric about the origin and then combining Theorems 1 and 2, and applying the reverse characteristic function transformation, Equation (4.6), to the result.

$$\begin{aligned} f_\Delta(x) &= \int_{-\infty}^{\infty} e^{jxt}\Phi_{\hat{\Delta}}(t)\,/\Phi_\varepsilon(t)\,dt \\ &= 2\int_0^{\infty} \cos(xt)\Phi_{\hat{\Delta}}(t)\,e^{t^2/n}dt \\ \hat{f}_\Delta(x;\tau,T) &= \frac{\tau}{\pi}\sum_{k\,:\,k\tau\leq T} \cos(xk\tau)\,e^{(k\tau)^2/n}\left(\frac{1}{N}\sum_{i=1}^{N}\cos\left(\hat{\Delta}_i k\tau\right)\right) \end{aligned} \tag{4.10}$$

In the event that f_Δ is not symmetric as we have assumed, our approximation will yield a symmetrized version of the true distribution, $\hat{f}_\Delta(x) \approx \frac{1}{2}(f_\Delta(x)+f_\Delta(-x))$. In this case we also point out that the difference between f_Δ and $\hat{f}_\Delta$ is of no consequence because the downstream univariate analyses operate on the absolute value of the observed statistics. In other words, we are as interested in finding genes that are up-regulated, as we are in genes that are down-regulated. Thus, it is definitely beneficial to assume the symmetry, as it leads to notable savings in computation. We need only evaluate $\hat{f}_\Delta(x)$ for $x \geq 0$ which is a factor 2 fewer calls to Equation (4.10). Furthermore, within each call to Equation (4.10), we need only sum over $k : 0 \leq k\tau \leq T$, which gives an additional factor 2 savings. We get a third factor 2 savings by noting that Theorem 2 tells us the characteristic function is real valued.[3]

We now seek to analyze how the choice of the parameters τ and T affect our approximation. Clearly we know that $\lim_{\tau\to 0, T\to\infty} \hat{f}_\Delta = f_\Delta$, but in practice we cannot use those limiting values.

4.2.1 Aliasing

In order to analyze the affect of τ on our approximation, we consider the case where $T = \infty$. Thus, the approximation noise is purely a result of τ.

[3] Arithmetic on complex numbers requires more work because we must track both the real and imaginary components.

Theorem 3 (Aliasing)**.** *If we extend our approximation in (4.10) to sum over all natural numbers then,*

$$\lim_{T\to\infty} \hat{f}_\Delta\left(x;\tau,T\right) = \sum_{k\in\mathbb{Z}} f_\Delta\left(x-\frac{k}{\tau}\right) \quad . \tag{4.11}$$

Theorem 3 is very informative as it shows us two important things:

1. From the summation of k across all integers, we can see that the estimate will be a summation of infinite "aliases" (*i.e.*, copies) of the true distribution.

2. The aliases will be spaced $\frac{1}{T}$ apart, and thus if the support of the distribution is larger than $x \in (-\frac{1}{2\tau}, \frac{1}{2\tau})$ the aliases (*i.e.*, the $f_\Delta\left(x-\frac{k}{\tau}\right)$) will overlap and we will not be able to recover the tails of the distribution.

In practice we expect that $f_\Delta\left(x\right) > 0$ for all $x \in \mathbb{R}$, but if we can make τ sufficiently small then we can push the aliases far enough apart that the aliasing effect is negligible. Figure 4.2 shows the aliasing by using Equation (4.11), for the case where the true effect distribution is Laplacian, $f_\Delta = \text{Laplace}\left(0,1\right)$. When $\tau = 0.5$ the aliases are so close that after they add up, $\hat{f}_\Delta\left(x\right) > 0.4$ for all x, resulting in a very poor approximation. But when $\tau = 0.1$ the aliases are sufficiently spaced that we have a very good approximation of $f_\Delta\left(x\right)$ on $x \in [-5,5]$.

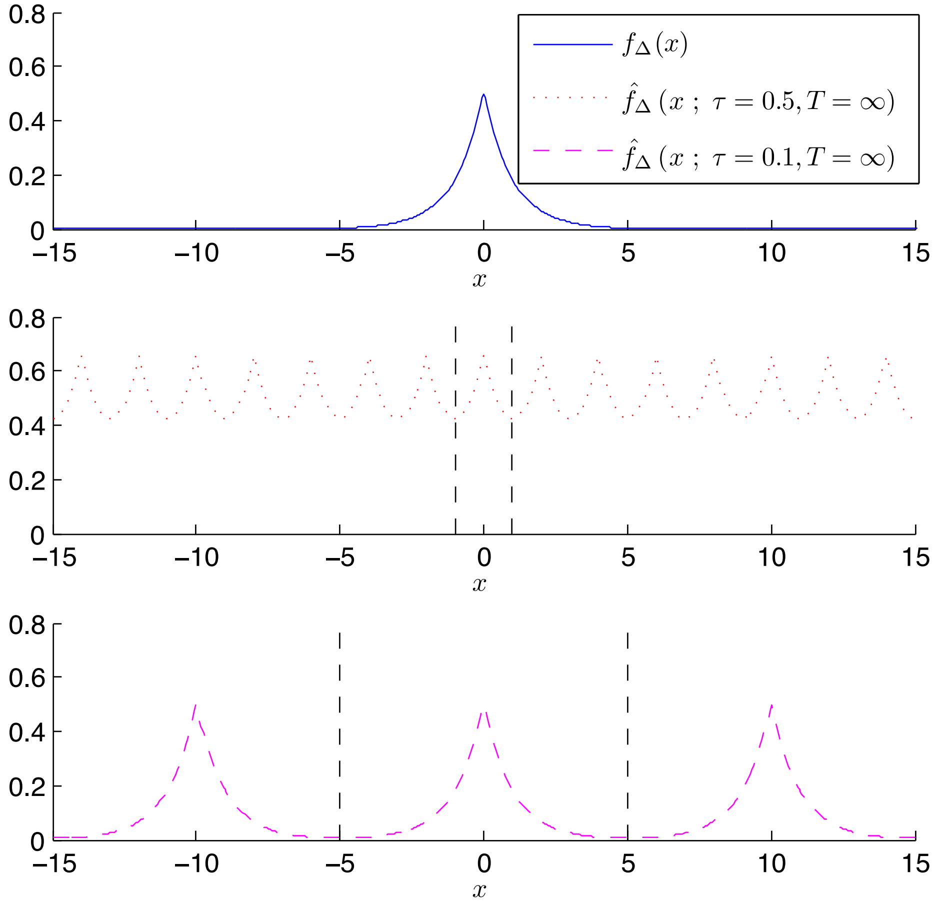

Figure 4.2: Comparison of the true distribution, $f_\Delta(x)$, and the aliased version, $f_\Delta(x;\tau,T)$, obtained from the approximation in Equation (4.11); using $\tau = \{0.1, 0.5\}$ and $T = \infty$. Vertical dashed black lines denote the interval for which the approximation is valid, $x \in (-\frac{1}{2\tau}, \frac{1}{2\tau})$.

4.2.2 Windowing

We now analyze the effect of T by considering the limit as τ approaches zero. We refer to this as the windowing effect, as T is used to control the interval (*i.e.*, window) of the characteristic function $\Phi_\triangle$ that we are integrating over.[4]

Theorem 4 (Windowing)**.** *In the limit, as we decrease τ in our approximation from (4.10),*

$$\lim_{\tau \to 0} \hat{f}_\triangle (x; \tau, T) = \left(f_\triangle (x) * \frac{\sin(xT)}{\pi x} \right) (x) \quad . \tag{4.12}$$

The effect of the convolution introduced in Equation (4.12) is to induce a "ripple" in the estimated distribution. Figure 4.3 illustrates this effect on $\hat{f}_\triangle$ with $T = \{1, 2, 4\}$. Again, we have used the Laplace distribution to model the distribution of true effect sizes, $f_\triangle = \text{Laplace}\,(0, 1)$. Note that this convolution has two very serious effects. Firstly, the tails of the estimated distribution are buried under the ripple. Secondly, the estimated distribution is not even a distribution – it does not integrate to 1, nor is it strictly non-negative. However, this is only apparent for the values of $T = \{1, 2\}$, which we have used to highlight the phenomena; when $T = 4$ the window effect is almost negligible. Clearly if we set $T = 4$ with τ small, then we can get a very good approximation of $f_\triangle$ here.

In practice our choice of T will be based on how confident we are in our empirical estimate of the characteristic function, $\hat{\Phi}_\triangle (t) = \Phi_{\hat{\Delta}} (t) / \Phi_\varepsilon (t)$. The function becomes harder to estimate as t increases, and thus we limit our integration to the window $t \in [0, T]$. In order to determine a proper value of T, we make the additional assumption that $f_\triangle$ is unimodal, and then by [5, Theorem 1] the characteristic function $\Phi_\triangle (t)$ must also be a continuous unimodal function. Thus, we can select T simply by plotting $\hat{\Phi}_\triangle (t)$ and determining the range over which it is unimodal. Figure 4.4 compares the true characteristic function, $\Phi_\triangle (t)$, versus the empirical estimate $\hat{\Phi}_\triangle (t) = \frac{1}{N} e^{t^2/n} \sum_i \cos(\hat{\Delta}_i t)$, for the same $\text{Laplace}\,(0, 1)$ distribution, with $n = 5$ microarrays per class. Note the function has a trough at $t \approx 4$ and thus $T = 4$ is a reasonable choice for this problem.

[4] Furthermore, in signal processing there is an analogous analysis for the constructing "windowing functions" for causal FIR filters [67, Chapet 7.2].

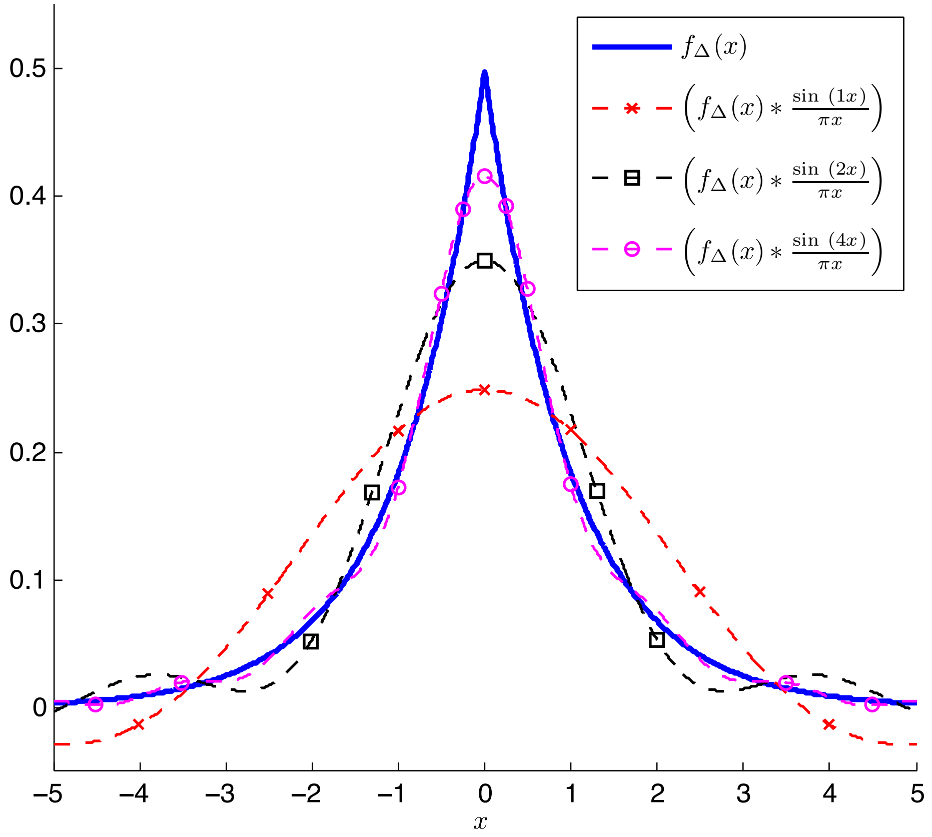

Figure 4.3: The effects of convolving a Laplace distribution with the function $\sin(xT)/\pi x$ for values $T = \{1, 2, 4\}$.

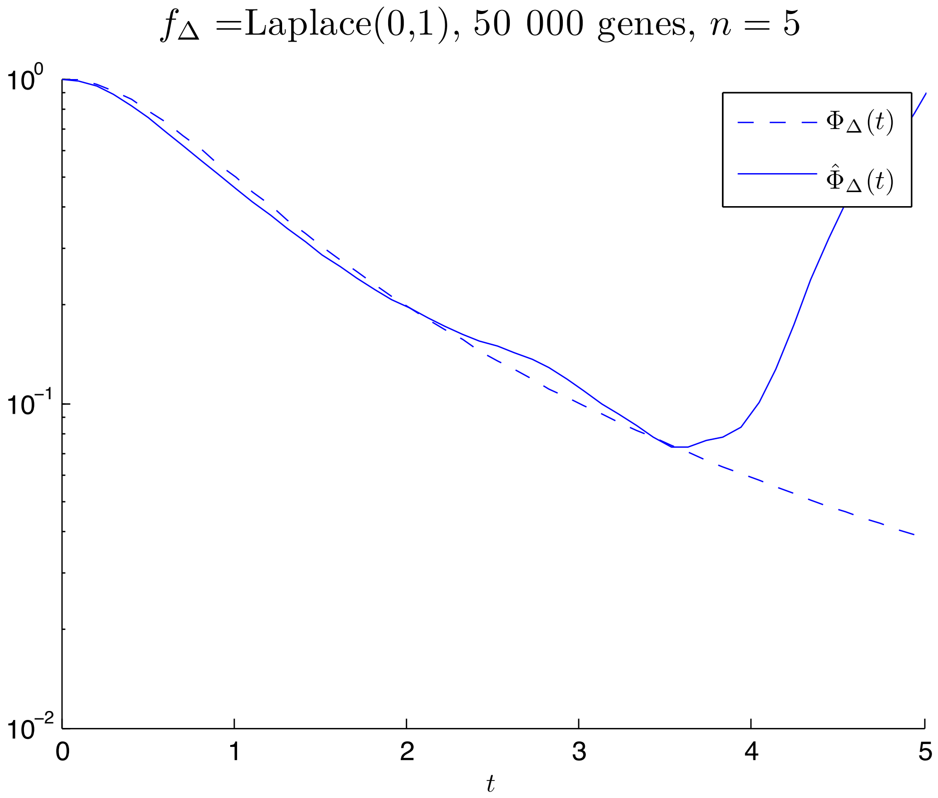

Figure 4.4: A comparison of the true characteristic function Φ_Δ versus its empirical estimate $\hat{\Phi}_\Delta$ produced from a study of 5 cases versus 5 controls, *i.e.*, $n = 5$.

The additional assumption that f_Δ is unimodal is quite reasonable, as we would expect that most genes are not relevant and this will force the distribution to be mostly unimodal. In other words, the extremely high abundance of genes with Δ_i near to 0 will force f_Δ to be mostly unimodal and symmetric[5] about the origin.

Now that we have analyzed the effect of τ and T separately, we compare the approximation from Equation (4.2) to Equation (4.10) directly. Figure 4.5 compares these approximations for our same Laplace $(0, 1)$ distribution, with $n = 5$ microarrays per class; for the comparison we use $\tau = 0.1$ and $T = 4$. Figure 4.5[left] shows that we get a better approximation of the distribution near the peak by using the characteristic function approach. Figure 4.5[right] shows that we get a better approximation of the distribution in the tails using the naive empirical approach.

[5] Recall, we previously motivated the symmetry assumption in Section 4.2.

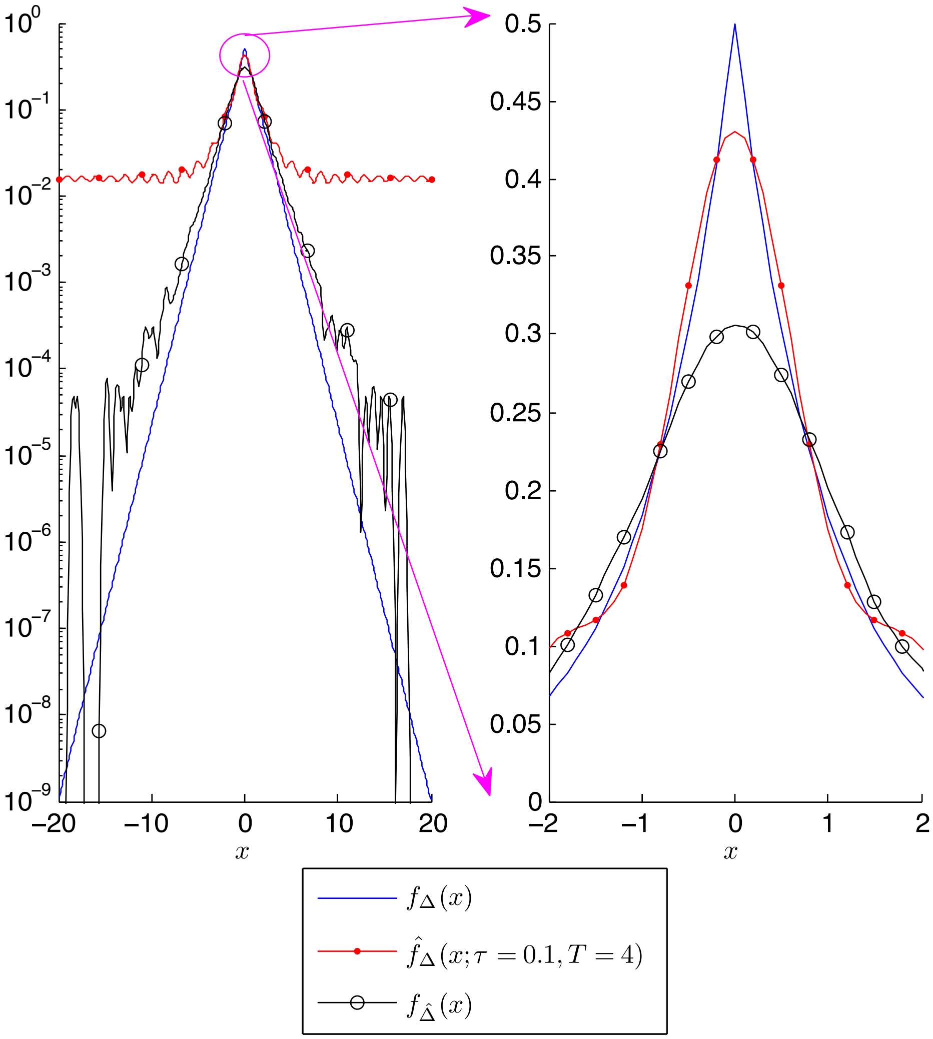

Figure 4.5: A comparison of our two approximation methods: the naive empirical estimate from Equation (4.2), and the characteristic function estimate from Equation (4.10). [left] highlights approximation in the tails. [right] highlights the approximation at the peak of the disrtibution.

4.3 FDE Algorithm

We have now shown two approaches to estimating f_Δ. The first method, Equation (4.2), naively used the empirical distribution of the observed $\hat{\Delta}_i$, *i.e.*, $\hat{f}_\Delta = f_{\hat{\Delta}}$. While this approximation was very intuitive, we showed that because it fails to model the observational noise it will produce a "smoothed" distribution, that "loses" peaks, and over-estimates tails. The second method, Equation (4.10), approximated the distribution by estimating its characteristic function and then numerically transforming it back into a PDF. We showed this method is better at recovering the bulk of the distribution but notably suffers in the tails (more so than the naive estimate). We now construct our FDE algorithm, Algorithm 1, that computes both density estimates and then fuses them to get the best of both of their properties.

FDE accepts as inputs a set of estimates $\{\hat{\Delta}_i\}$ and the assumed noise distribution $\Phi_\varepsilon(t)$. Thus, FDE works for any statistic with an additive noise model, not just our specific case where the noise is normally distributed. For our BBD problem the noise distribution has characteristic function $\Phi_\varepsilon(t) = e^{-t^2/n}$, where n is the number of microarrays used (per class).

The algorithm requires that the user set τ based on the available computational resources. Note it takes $O\left(\frac{T}{\tau}N\right)$ time to evaluate Equation (4.10). In Section 4.2.2 we showed how to select an appropriate value of T. We believe that given T, it is straightforward for the user to determine an appropriate value τ for their use. When used within our BBD algorithms in Chapter 5, we see that the runtime FDE is relatively very small in comparison to the time it takes to use the resulting estimate of $\hat{f}_\Delta$ to tune an algorithm's parameters – *i.e.*, in practice we can always set τ sufficiently small to avoid aliasing effects.

Algorithm 1 FDE($\{\hat{\Delta}_i\}$, $\Phi_\varepsilon\left(t\right)$)

```
$\Phi_{\hat{\Delta}}\left(t\right) = \frac{1}{N}\sum_{i=1}^{N}\cos\left(\hat{\Delta}_i t\right)$
$\Phi_{\Delta}\left(t\right) = \Phi_{\hat{\Delta}}\left(t\right)/\Phi_{\varepsilon}\left(t\right)$
$T = \max_{t\in\mathbb{R}^+}\{t : \frac{\partial}{\partial t}\Phi_{\Delta}\left(t\right) \leq 0\}$
Set $\tau$ based on the available computational resources
Use $\tau$ and $T$ with Equation (4.10) to produce $\hat{f}_{\Delta^{(1)}}\left(x\right)$
Use Equation (4.2) to produce $\hat{f}_{\Delta^{(2)}}\left(x\right)$
$x^* = \min_{x\in\mathbb{R}^+}\{x : \hat{f}_{\Delta^{(1)}}\left(x\right) = \hat{f}_{\Delta^{(2)}}\left(x\right)\}$
$w_1 = \int_{-x^*}^{x^*}\hat{f}_{\Delta^{(1)}}\left(x\right)dx$
$w_2 = \int_{-x^*}^{x^*}\hat{f}_{\Delta^{(2)}}\left(x\right)dx$
$f_{\Delta^{\text{(fused)}}}\left(x\right) = \begin{cases} \hat{f}_{\Delta^{(1)}}\left(x\right) & |x| \leq x^* \\ \frac{1-w_1}{1-w_2}\hat{f}_{\Delta^{(2)}}\left(x\right) & |x| > x^* \end{cases}$
**return** $f_{\Delta^{\text{(fused)}}}\left(x\right)$
```

4.4 Experiments

4.4.1 Experiments on Real Data

To show the benefit of using FDE over the naive empirical estimator, Equation (4.2), we consider the problem of estimating f_Δ on real microarray data. As we cannot truly know f_Δ for any real dataset, we take a large microarray dataset and use all the microarrays to compute $\hat{f}_{\hat{\Delta}}$ by our empirical estimator, Equation (4.2), and use that as our ground truth (*i.e.*, we use this particular $f_{\hat{\Delta}}$ as f_Δ). We then run our algorithms on sub-samples of the dataset and compare their estimates to the estimate using the full dataset. By the construction of this experiment, as the sub-samples increase in size, the empirical estimator will converge towards the estimate using the full dataset. This experiment will explore whether our FDE algorithm converges faster than the empirical distribution, $f_{\hat{\Delta}}$. For the experiment we will be using the following datasets from GEO:

GSE11882 This was a study to examine the difference in gene expression in normal brains between men and women. This dataset has 173 microarrays, 91 male versus 82 female, with 54 675 genes.

GSE19743 This was a study to examine the difference in gene expression of skin tissues of burn patients versus normal skin tissues, but we decided to use sex as the phenotype to make it consistent with our other datasets. This dataset has 177 microarrays, 120 male versus 57 female, with 54 675 genes.

GSE41726 This was a study to examine the difference in gene expression in the skeletal muscle tissues of patients with cancer cachexia. This dataset has 134 microarrays, 69 male versus 65 female, with 41 000 genes.

We chose to use GSE41726 because it is our dataset from a previous study [78]. We chose to use GSE11882 and GSE19743 because they use Affymetrix HG-U133-Plus-2[6] arrays, they had very large sample sizes, and we could extract the sex of the patients as a phenotype from the meta-data associated with the dataset. Note that

[6] This is the most used brand of microarray across all datasets on GEO.

we excluded other datasets using the same brand of microarray with larger sample sizes because upon inspection it was difficult to ascertain if the data was from a single study or was a composition of datasets from separate studies. We wanted to avoid the latter because unless properly handled, mixing microarray datasets will give rise to unwanted batch effects [47, 58]. We chose to use sex as our phenotype because it is incontrovertible[7] and common to all of our datasets.

For our experiment, we take sub-samples of sizes $n = \{5, 10, 15, \ldots, 50\}$, *i.e.*, we compare samples of n men versus n women. To evaluate the algorithms, we compare the estimate using all the data, f_Δ, to the estimate produced on the sub-sample, $\hat{f}_\Delta$, in terms of KL divergence, KS statistic, and Mean Squared Error (MSE).

$$\begin{aligned} \text{KL divergence} &= \int_{-\infty}^{\infty} f_\Delta(x) \log\left(\frac{f_\Delta(x)}{\hat{f}_\Delta(x)}\right) dx \\ \text{KS statistic} &= \max_{x\in\mathbb{R}} \left| F_\Delta(x) - \hat{F}_\Delta(x) \right| \\ \text{MSE} &= \int_{-\infty}^{\infty} \left(f_\Delta(x) - \hat{f}_\Delta(x) \right)^2 dx \end{aligned}$$

Figure 4.6 shows the results. As we have used the empirical algorithm to set the ground truth, for large n, these scores will converge to 0. However, we see that FDE has a much faster convergence rate because it models the observational noise. Note, that FDE seems to hit a performance barrier for $n \geq 30$ on GSE19743 and GSE41726. This is to be expected because ultimately both algorithms produce different estimates.

[7] Patients are clearly either male or female. While there are some rare chromosomal abnormalities such as females with XO chromosomes (Turner syndrome [61]), or males with XXY chromosomes (Klinefelter syndrome [57]), *etc.*, that could affect the analyses and these patients should be removed as statistical outliers. We know that GSE41726 does not contain any patients with these abnormalities and believe that neither GSE11882 nor GSE19743 contain them as well.

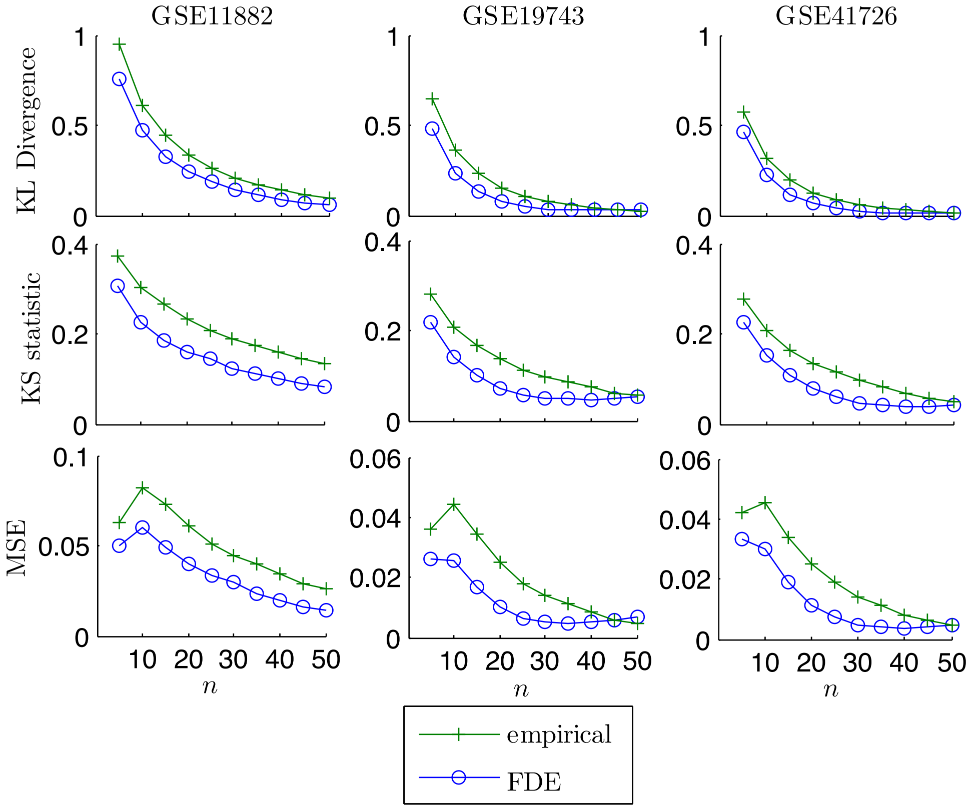

Figure 4.6: Comparison of FDE versus the empirical estimator on sub-samples from real datasets. n denotes the sample size (n men versus n women).

4.4.2 Experiments on Synthetic Data

We now show that the results of FDE should generalize over many other types of distributions. Here we use synthetic datasets where we can perform parameter sweeps on f_Δ and show that FDE outperforms the empirical estimator for all parameterizations of f_Δ. For these experiments we consider the case where we have $N = 50\ 000$ genes observed in an experiment comparing 5 microarrays from the cases versus 5 from the controls. For each gene, g_i, we draw a Δ_i from f_Δ. Then we draw 5 values from a normal $\mathcal{N}(\Delta_i, 1)$ distribution (for the cases) and 5 values from a $\mathcal{N}(0, 1)$ distribution (for the controls), from which we compute $\hat{\Delta}_i$, *i.e.*, we use our plate model from Section 2.1 to generate the data.

Figure 4.7 plots the performance of our algorithms over the class of zero mean Laplace distributions, $f_\Delta = \text{Laplace}\,(0, b)$, and the class of zero mean normal distributions, $f_\Delta = \mathcal{N}\,(0, \sigma^2)$. For all settings, we see that our FDE algorithm outperforms the empirical estimator in terms of KL divergence, KS statistic and MSE. Note that as b and σ^2 increase the problem becomes easier (for both algorithms) as the values of Δ_i become much larger than the observation noise, which makes it easier to learn their distribution.

As both of the algorithms are based on non-parametric estimators, we believe that this trend will hold for any other reasonable distribution f_Δ. In the subsequent section we will show another class of distributions where our results hold, and FDE even outperforms parametric estimators.

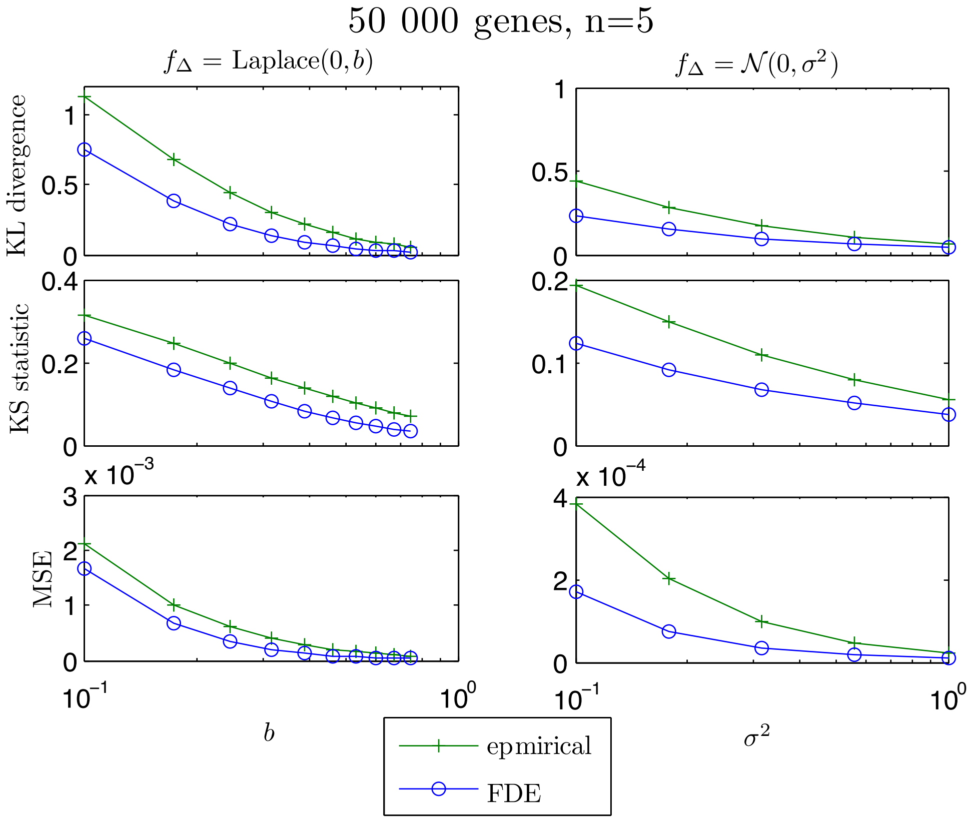

Figure 4.7: Comparison of FDE versus the empirical estimator over the class of zero mean Laplace distributions, and the class of zero mean normal distributions.

4.5 Parametric Models

The previous section showed that our FDE algorithm outperforms the naive empirical estimator. However, both of these estimators are entirely non-parametric, and so it is natural to ask, "can we do better if we know *a priori* that f_Δ falls within some parametric family?" From our additive noise model, assuming we know f_ε, we can get very reliable estimates of the mean and variance of f_Δ,

$$\begin{aligned} \mathrm{E}[\Delta] &= \mathrm{E}[\hat{\Delta}] - \mathrm{E}[\varepsilon] \\ \mathrm{Var}(\Delta) &= \mathrm{Var}(\hat{\Delta}) - \mathrm{Var}(\varepsilon) \quad . \end{aligned}$$

Thus, the short answer is yes, *IFF* there is a good mapping of the distributional parameters of f_Δ to what we can observe, *i.e.*, if the parameters can be recovered if we know the mean and variance of the distribution.

However, there are distributional families for which the parameters cannot be extracted from the mean and variance. Consider the Cauchy distribution,

$$f_x(x) = \frac{1}{\pi\gamma\left[1 + \left(\frac{x-x_0}{\gamma}\right)^2\right]} \quad ,$$

with location parameter x_0 and scale parameter γ. This distribution has undefined mean and variance and thus we cannot relate x_0 and γ to the sample mean and variance of the observed $\hat{\Delta}_i$.

Note that the Cauchy distribution is one of the well-known power law distributions, and previous works have shown that statistics of gene expression data follows power laws in practice [60]. Thus, it is a potentially very relevant distribution, and not just an obscure mathematical construction. We now address the problem of estimating γ when we know that f_Δ is a Cauchy distribution.

Traditionally the scale parameter γ is estimated by taking half the sample InterQuartile Range (IQR) [20], *i.e.*, $\hat{\gamma} = \frac{q_{3/4} - q_{1/4}}{2}$, where q_p denotes the p'th quantile, $q_p = F_\Delta^{-1}(p)$. Here the naive approach to estimating γ would be to use the IQR in the observed $\hat{\Delta}_i$. Note that this is equivalent to using the empirical distribution estimator, Equation (4.2), to compute $\hat{f}_\Delta$ and then extracting the interquartile range of that distribution. However, as shown in Section 4.1, the observation noise will push mass towards the tails of the estimated distribution, which will cause us to overestimate the interquartile range. A more robust approach would be to run our FDE algorithm to get a better estimate of the distribution, and then compute the half interquartile range from that distribution.

Here we perform an additional experiment on synthetic data where we use the class of zero median Cauchy distributions, $f_{\Delta} = \text{Cauchy}\,(0, \gamma)$, and we sweep γ. Again, we use $N = 50\ 000$ genes observed in an experiment comparing 5 microarrays from the cases to 5 from the controls. Figure 4.8 compares the difference in the estimated Cauchy distribution based on the IQR extracted from the approximate distribution using Equation (4.2) to IQR extracted from our FDE algorithm. We use IQR to denote the naive algorithm that uses half the IQR of $\hat{\Delta}_i$, and use FDE-IQR to denote the more intelligent algorithm that uses half the IQR from the distribution produced by FDE. As interesting comparison points we also include the performance of the non-parametric distribution estimates from these algorithms. Unsurprisingly, the parametric approach is always better than its non-parametric counterpart – *i.e.*, FDE-IQR is better than FDE. Interestingly the FDE algorithm generally outperforms the IQR estimate produced by the empirical estimator. Thus, by modelling the observation noise, our non-parametric distribution estimate is actually better than our naive parametric estimate that fails to model the noise.

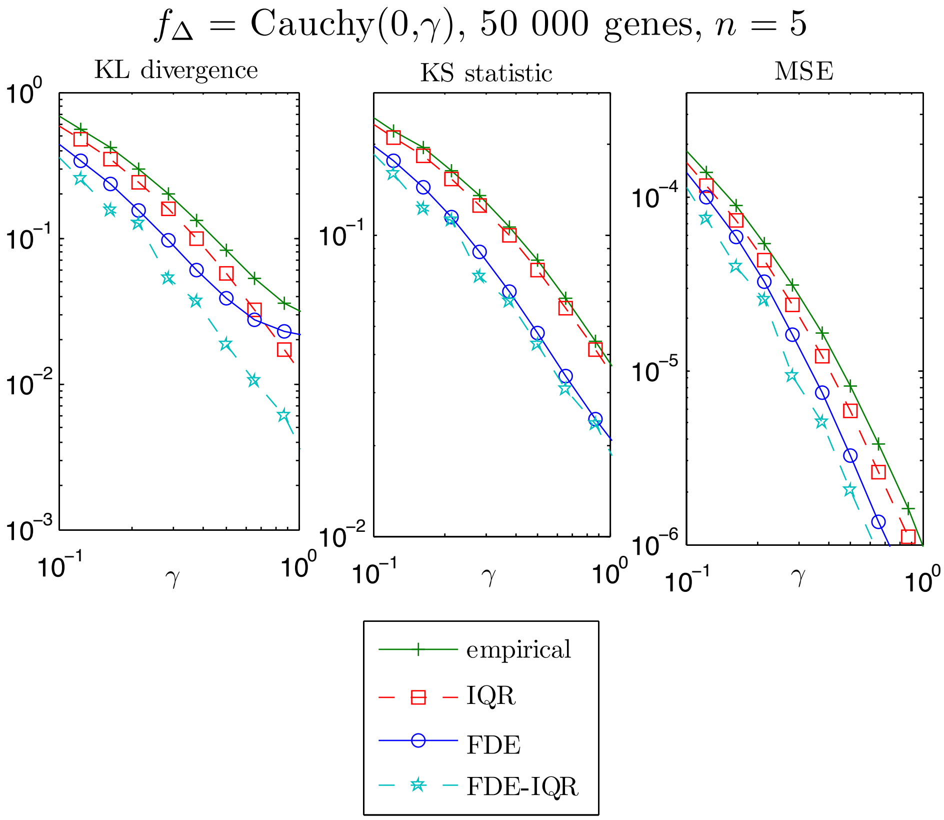

Figure 4.8: Comparison of estimators over the class of 0 median Cauchy distributions. Solid lines are non-parametric approaches, dashed lines are parametric solutions based on computing $\hat{\gamma} = \frac{q_{3/4} - q_{1/4}}{2}$.

Chapter 5

Algorithms for BBD

5.1 Simplifying Assumptions

In this chapter we will present algorithms for solving the BBD problem.[1] To fully solve the BBD problem, an algorithm will make use of data that has been collected and the remaining budget, to specify which test should be performed next and the patient on whom we shall perform the test. For example, collect a microarray from Mrs. Smith, or collect the cardiotoxicity qPCR array from Mr. Jones, *etc*. However, writing a formal policy for an algorithm to follow is a difficult task. Any such policy will contain parameters to control its behaviour. If these parameters are set correctly then the algorithm will behave effectively but if they are set incorrectly the algorithm may behave very poorly. Here we will introduce some additional assumptions and restrictions on the BBD algorithms we will develop. By limiting our algorithms we reduce the number of parameters that they have, and thus make it easier to optimize them to perform well given the limited data that we envision BBD algorithms will be able to collect.[2]

Independence We assume that the expression values of any pair of genes for a specific patient are independent when conditioned on the patient's phenotype,

[1] Much of the material in this chapter corresponds to [52, 53, 54]. In particular, our TNAS-FDR, BBD1, and BBD-Greedy are clear improvements of our previous RR, RR+RR, and mLUCB algorithms (from [54]) respectively. [52, 53] also had other algorithms, but they were very specific to the models we used at the time, and as those models have become outdated so too have the algorithms.

[2] Subsequently, in Chapter 6 we will see that even with these restrictions, it may still be challenging to tune the parameters for some BBD algorithms.

i.e., we assume a naive bayes model. In other words, knowing the value of expression value of gene g_{i_1} for a patient, gives us no information about the unknown expression value of gene g_{i_2} if we know which group (cases or controls) the patient is from.

While we know that this assumption is definitely NOT true, we believe that it is a necessary assumption for algorithms to make. The issue is that we do not know the correlation structure between the genes *a priori*, and thus in order for an algorithm to exploit any dependencies it must first learn them from the data.

As our goal is to determine the relevance of each gene by itself, this assumption does not bias us towards making false statements. However, if we had a reliable correlation structure to exploit we could potentially reach the same conclusions with less data.

Balanced designs and indifference to patients For convenience, we restrict our algorithms such that when collecting data, they can only specify which test to perform. Once a test has been selected, we run that test on a random patient from the case group and a random patient from the control group. This restriction drastically reduces the complexity of our BBD algorithms. Without it the algorithm could select any test $t \in \mathcal{T}$, and any patient in the study. However, we may have issues with picking patients in practice. For example, we may able to recruit a couple hundred cancer patients to participate in a study, but our budget limits us to collecting 50 microarrays. Without some type of gene expression values for a patient a BBD algorithm has no information from which it can assess the utility of collecting more data for that patient. In practice, we expect that biological rationale (outside of the scope of the BBD problem) can be used to select informative patients once the algorithm has decided which test should be performed.

For ease of notation in our algorithms, we will assume that algorithms track and update sufficient statistics for all data collected, and estimates derived from them. Specifically, they track all parameters to compute the aggregate estimate of the effect size,

$$\begin{aligned}
\hat{\mu}_{i,s}^{(t)} &= \frac{1}{n^{(t)}} \sum_{m=1}^{n^{(t)}} \psi_{i,s,m}^{(t)} \\
\hat{\sigma}_{i,s}^{(t)2} &= \frac{1}{n^{(t)} - 1} \sum_{m=1}^{n^{(t)}} \left(\psi_{i,s,m}^{(t)} - \hat{\mu}_{i,s}^{(t)} \right)^2 \\
\hat{\Delta}_i^{(t)} &= \frac{\hat{\mu}_{i,1}^{(t)} - \hat{\mu}_{i,0}^{(t)}}{\sqrt{\left(\hat{\sigma}_{i,1}^{(t)2} + \hat{\sigma}_{i,0}^{(t)2} \right) /2}} \\
\hat{\Delta}_i &= \frac{\sum_{t \in \mathcal{T}} n^{(t)} \hat{\Delta}_i^{(t)}}{\sum_{t \in \mathcal{T}} n^{(t)}} \quad ,
\end{aligned} \tag{2.4 revised}$$

here we use $n^{(t)}$ to denote the number of observations of test, t, we have collected (per class). If one of the algorithms performs a t-test, it uses $\hat{\Delta}_i$ to test if $\Delta_i = 0$, measured by the p-value p_i.

We will also make use of ordered statistics notation [23]. Thus, we use p_i to denote the p-value of gene g_i, and $p_{(i)}$ to denote the i'th largest p-value. In other words, if we sort the p-values of all genes in ascending order, $[p_{(1)}, p_{(2)}, \ldots, p_{(N)}]$, then $p_{(i)}$ corresponds to the i'th element in the sort. Similarly, we will use the term $|\hat{\Delta}|_{(i)}$ to refer the i'th element of the sorted absolute values of the estimated effect sizes, $[|\hat{\Delta}|_{(1)}, |\hat{\Delta}|_{(2)}, \ldots, |\hat{\Delta}|_{(N)}]$. This should not be confused with $|\hat{\Delta}_i|$, which we use to denote the absolute value of the estimated effect size for gene g_i.

5.2 Traditional Naive Association Studies

Our first algorithms represents the Traditional Naive Association Study (TNAS) approach used by many researchers.[3] This approach spends the entire budget collecting microarrays and then uses False Discover Rate (FDR) controlled t-tests to determine which genes are relevant.[4] Algorithm 2 presents pseudocode for our basic TNAS algorithm. The algorithm accepts as input: the available budget B, the

[3] Here we mean that the approach is naive in the context of the BBD problem because they only use microarrays. We are not calling those researchers naive.

[4] A FDR controlled t-test is multiple hypothesis testing proceedure that (under some assumptions) upper bounds the proportion of false positives in the returned genes, *i.e.*, the precision of the algorithm has a lower bound.

cost of a microarray C_{array}, and the level of FDR control α. Line 3 implements the Benjamini-Hochberg method to control the FDR [12].

Algorithm 2 TNAS(B, C_{array}, α)

```
1: $n = \lfloor \frac{B}{2C_{array}} \rfloor$
2: Collect $n$ microarrays from both the cases and controls.
3: $p^* = \max_s \{p_{(s)} : \forall r \leq s : p_{(r)} \leq \frac{r}{N}\alpha\}$
4: return $\hat{R} = \{g_i : p_i \leq p^*\}$
```

A drawback of this algorithm is that there is no clear mapping between the value of α, and the evaluation score. We extend TNAS to TNAS-FDR, which examines the collected data to ascertain the difficulty of the problem, and then selects an appropriate level of FDR control to maximize the expected evaluation. Thus, TNAS-FDR brings the traditional approach into the BBD framework.

In Algorithm 3:line 3, TNAS-FDR calls the FDE algorithm to produce an estimate of the distribution of the true effect sizes, $\hat{f}_{\Delta}$. Knowing this distribution, the algorithm can then hypothesize the expected evaluation of TNAS with various values of the FDR control α, and then select the value, α^*, with the best evaluation in expectation. If our estimated density function is good, then the optimal value α^* for $\hat{f}_{\Delta}$ will also give good results on our actual problem. We suggest using the golden search method to efficiently find this α^*; see Appendix 9.1.

In a similar manner we construct our TNAS-Top$K$ algorithm, Algorithm 4, that instead of accepting genes based on FDR controlled t-tests, accepts the top K features as relevant. Again, we exploit the plate model to tune an appropriate value for K.

Algorithm 3 TNAS-FDR(B, C_{array})

```
1: $n = \lfloor \frac{B}{2C_{array}} \rfloor$
2: Collect $n$ microarrays from both the cases and controls.
3: $\hat{f}_{\Delta} = \text{FDE}\left(\{\hat{\Delta}_i\}, \Phi_{\varepsilon}(t) = e^{-\frac{t^2}{n}}\right)$
4: $\alpha^* = \arg\max_{\alpha} \mathrm{E}_{\Delta}\left[\text{evaluation}\left(R, \text{TNAS}\left(B,\ C_{array},\ \alpha\right)\right)\middle| \hat{f}_{\Delta}\right]$
5: $p^* = \max_s\ p_{(s)} : \forall r \leq s : p_{(r)} \leq \frac{r}{N}\alpha^*$
6: return $\hat{R} = \{g_i : p_i \leq p^*\}$
```

Algorithm 4 TNAS-Top$K$(B, C_{array})

```
$n = \lfloor \frac{B}{2C_{array}} \rfloor$
Collect $n$ microarrays from both the cases and controls.
$\hat{f}_\Delta = \text{FDE}\left(\{\hat{\Delta}_i\}, \Phi_\varepsilon(t) = e^{-\frac{t^2}{n}}\right)$
$K^* = \arg\max_K \mathrm{E}_\Delta \left[\text{evaluation}\left(R, \hat{R} = \left\{g_i : |\hat{\Delta}_i| \geq |\hat{\Delta}|_{(N-K+1)}\right\}\right) \middle| \hat{f}_\Delta\right]$
return $\hat{R} = \left\{g_i : |\hat{\Delta}_i| \geq |\hat{\Delta}|_{(N-K^*+1)}\right\}$
```

5.3 Greedy Algorithm

While it is important to understand TNAS, TNAS-FDR, and TNAS-Top$K$, none of these algorithms model the full BBD problem, as they do not make use of qPCR. Here we consider the full BBD problem where the algorithm has access to a set of possible tests $\mathcal{T}$, where a test may be either a microarray or an "off-the-shelf" qPCR array.[5] Here we will consider an algorithm that makes full use of the available tests. In order to have some information on which to base its decisions, the algorithm will begin by collecting some microarrays, then it will seek to optimize a heuristic function, $h\left(\{(\hat{\Delta}_i, n_i)\}, \mathcal{S}\right)$, where $(\hat{\Delta}_i, n_i)$ is a tuple representing the current estimated effect size of gene g_i and the number of observations it is based on, and $\mathcal{S} \subseteq \mathcal{T}$ is the set of tests the algorithm would like to perform. By constructing the heuristic to be a close approximation of the expected evaluation, maximizing the heuristic will lead the algorithm to perform well on the BBD problem at hand. This idea is similar to the optimization steps in TNAS-FDR and TNAS-Top$K$, *e.g.*, Algorithm 3: line 4. The difference is that, in those algorithms, we were only concerned with how to identify the genes to return, given the current information about the genes; here, we must also determine what information to collect – making the heuristic more complicated.

We now seek to construct an appropriate heuristic function, and an algorithm for maximizing it. Unfortunately, we cannot use the same approach used by the TNAS algorithms, because here our algorithm decides what specific data to collect. Previously, we were able to use $\hat{f}_\Delta$ and our plate model to generate synthetic data

[5] We use the term "off-the-shelf" to refer to arrays that use a pre-defined set of genes; we are not allowing the algorithm to design custom qPCR arrays. Custom qPCR arrays will be considered subsequently in our BBD1 algorithm.

for very similar microarray studies to tune our parameters. But we cannot map the genes of a specific test used here, to a set of fictitious ones used in parameter tuning.

Here we can use our observed data to compute the probability that a given gene, g_i, is relevant,

$$\begin{aligned} p_i(\hat{\Delta}_i, n_i) &= P\left(g_i \in\ R\right) \\ &= \begin{cases} 1 - \frac{1}{2}\left(F_{|\Delta-\hat{\Delta}|}\left(|\hat{\Delta}_i| + \Delta^*\right) - F_{|\Delta-\hat{\Delta}|}\left(|\hat{\Delta}_i| - \Delta^*\right)\right) & |\hat{\Delta}_i| > \Delta^* \\ 1 - \frac{1}{2}\left(F_{|\Delta-\hat{\Delta}|}\left(|\hat{\Delta}_i| + \Delta^*\right) + F_{|\Delta-\hat{\Delta}|}\left(|\hat{\Delta}_i| - \Delta^*\right)\right) & |\hat{\Delta}_i| \leq \Delta^* \end{cases} \\ F_{|\Delta-\hat{\Delta}|}\left(x\right) &\approx \int_{-x}^{x} \frac{1}{\sqrt{4\pi/n_i}} e^{-\frac{n_i y^2}{4}} dy \quad , \end{aligned}$$

where $\hat{\Delta}_i$ and n_i are the values used in Equation 2.4 to summarize our knowledge on gene g_i. To compute this probability, we have used the CDF of the absolute deviation from the mean of an estimated effect size, $F_{|\Delta-\hat{\Delta}|}$. We approximate this CDF using an appropriate normal distribution, see Appendix 8.2. To evaluate the utility of performing the tests in $\mathcal{S}$, we can inflate n_i to match the number of samples seen after the data is collected,

$$\begin{aligned} h\left(\{(\hat{\Delta}_i, n_i)\}, \mathcal{S}\right) &= \max_{\hat{R}} \mathrm{E}_{\Delta}\left[\text{evaluation}(R, \hat{R}) \middle| \left\{p_i(\hat{\Delta}_i, n_i')\right\}\right] \\ n_i' &= n_i + \sum_{t\in\mathcal{S}} \mathbb{1}\left(g_i \in \mathcal{G}_t\right) \quad . \end{aligned} \tag{5.1}$$

To compute the expected evaluation we simply enumerate all possible outcomes, and weight them by the given probabilities. We use the method of [46] to compute the expected evaluation given the probabilities $\{p_i\}$, which requires O(N^4) time.

Now to optimize our heuristic function, we note that it has two key properties. Firstly, extra tests will always improve the expected evaluation, and secondly the impact of new tests will become less as more data is collected, *i.e.*, we observe diminishing returns. More formally, given two sets, $\mathcal{S}_1 \subseteq \mathcal{S}_2$, then,

$$\begin{gathered} h\left(\mathcal{S}_1\right) \leq h\left(\mathcal{S}_2\right) \\ h\left(\{\mathcal{S}_1 \cup t\}\right) - h\left(\mathcal{S}_1\right) \geq h\left(\{\mathcal{S}_2 \cup t\}\right) - h\left(\mathcal{S}_2\right) \quad , \end{gathered}$$

where we have dropped the dependence on $\{(\hat{\Delta}_i, n_i\}$ for notational simplicity. Thus, our heuristic is a monotone submodular set function with respect to $\mathcal{S}$ [33].

It is well-known that greedy algorithms can provide good solutions to submodular maximization problems [56, 65]. Furthermore, since our problem is stochastic, finding the optimal set of tests, $\mathcal{S}^*$, to maximize our initial heuristic may not be a good idea, because as we begin collecting the data for those tests, we can update our heuristic with the new data, and we may then prefer to spend the rest of the budget collecting, $\mathcal{S}' \neq \mathcal{S}^*$. Thus, our problem has the same properties as the adaptive submodular maximization problem [39]. It has been shown that the greedy algorithm is a good solution for such problems.

Algorithm 5 presents the obvious greedy algorithm, BBD-Greedy. Line 5 selects the test (among those that it can afford) that is most cost effective at raising the heuristic. Although we initially said that this algorithm cannot make use of custom qPCR arrays, this selection criteria can be easily extended to include the custom arrays that are most likely useful – *i.e.*, those that cover genes that are borderline relevant. This modification can be done with a simple update to $\mathcal{T}$, but we chose not to show the modification, as it is not informative in understanding the behaviour of this algorithm. Furthermore, we will directly address the problem of using custom qPCR in the subsequent section.

Algorithm 5 BBD-Greedy($\mathcal{T}$)

```
 1: Collect 2 microarrays from both the cases and controls.
 2: $B' = B - 4C_{array}$
 3: while $B' \geq 2\min_{t\in\mathcal{T}} C_t$ do
 4:     $h_{current} = h\left(\{(\hat{\Delta}_i, n_i\}, \{\}\right)$
 5:     $t^* = \arg\max_{t\in\mathcal{T}:2C_t\leq B'} \left\{\frac{h\left(\{(\hat{\Delta}_i,n_i\},t\right)-h_{current}}{C_t}\right\}$
 6:     perform test $t^*$ on 1 example from the cases and 1 from the controls.
 7:     $B' = B' - 2C_{t^*}$
 8: end while
 9: $\hat{R} = \arg\max_{\hat{R}\subseteq\mathcal{G}} \mathrm{E}_{\Delta}\left[\text{evaluation}(R, \hat{R})\middle|\left\{p_i(\hat{\Delta}_i, n_i)\right\}\right]$
10: return $\hat{R}$
```

5.4 Custom qPCR Algorithms

We now consider algorithms that utilize a combination of microarrays and custom qPCR arrays. The envisioned behaviour of these algorithms is to intelligently collect a few microarrays to assess the relevant preliminary info of the problem. Then once it is cost effective to do so, the algorithm will potentially label the best genes as relevant, and select the genes that are on the borderline of relevance for follow-up with custom qPCR arrays. We consider the case where only a single custom qPCR array will be created but we may use it to perform as many tests as we can afford given the budget, *i.e.*, we can specify an arbitrary set of N_{PCR} genes, and then test those genes for a cost of C_{PCR}, as many times as our budget permits. We believe that this is the type of the behaviour desired by the biologists currently looking for a principled method for checking studies [2, 73].

Here we will present two algorithms that, instead of using the data collected to select the tests that are most cost effective at raising the expected evaluation, follow general policies to determine when (if ever) to stop collecting microarrays, and if so which genes should be selected for qPCR based on their ranked statistics. The idea here is that we can use our plate model to tune good parameters for those policies given the estimate of $\hat{f}_\Delta$ from our FDE algorithm.

Algorithm 6 presents pseudocode for our BBD1 algorithm. The algorithm accepts as input: the available budget B, the cost of a microarray C_{array}, the cost of custom qPCR array C_{PCR}, and the number of genes it covers N_{PCR}. At first the algorithm collects the bare minimum number of microarrays needed for it to tune its parameters. As this estimate is likely to be noisy, the algorithm then continues to refine these estimates as more microarray data is collected (see the while loop at Line 5). Once it believes it has collected the appropriate number of microarrays, BBD1 then labels the top K genes as relevant, and follows up on the next N_{PCR} genes with the qPCR. Finally, it decides which of these followed-up genes are relevant by applying a threshold test on the aggregated effect estimates $\hat{\Delta}_i$.

To tune its internal parameters BBD1 makes use of the sub-routine BBD1-Core, which computes the expected evaluation score it will receive for a given parameterization. Appendix 9 will present useful search algorithms to efficiently tune these parameters.

Algorithm 6 BBD1(B, C_{array}, C_{PCR}, N_{PCR})

```
$n = 2$
Collect $n$ microarrays from both cases and controls
$\hat{f}_\Delta = \text{FDE}\left(\{\hat{\Delta}_i\}, \Phi_\varepsilon(t) = e^{-\frac{t^2}{n}}\right)$
$n^*, K^*, \zeta^* = \arg\max_{n,K,\zeta} \mathrm{E}_\Delta\left[\text{evaluation}\left(R, \text{BBD1-Core}\left(n, K, \zeta\right)\right) \middle| \hat{f}_\Delta\right]$
while $n \neq n^*$ do
    $n = n + 1$
    Collect a microarray from both cases and controls
    $\hat{f}_\Delta = \text{FDE}\left(\{\hat{\Delta}_i\}, \Phi_\varepsilon(t) = e^{-\frac{t^2}{n}}\right)$
    $n^*, K^*, \zeta^* = \arg\max_{n,K,\zeta} \mathrm{E}_\Delta\left[\text{evaluation}\left(R, \text{BBD1-Core}\left(n, K, \zeta\right)\right) \middle| \hat{f}_\Delta\right]$
end while
$\hat{R} = \left\{g_i : |\hat{\Delta}_i| \leq |\hat{\Delta}|_{(K^*)}\right\}$
$\mathcal{S} = \left\{g_i : |\hat{\Delta}|_{(K^*)} < |\hat{\Delta}_i| \leq |\hat{\Delta}|_{(K^*+N_{PCR})}\right\}$
Create a custom qPCR array for the genes in $\mathcal{S}$
Spend remainder of budget collecting data on this custom qPCR array
$\hat{R} = \hat{R} \cup \left(\left\{g_i : |\hat{\Delta}_i| \geq \zeta^*\right\} \cap \mathcal{S}\right)$
return $\hat{R}$
```

Algorithm 7 BBD1-Core(n, K, ζ)

```
Collect $n$ microarrays from both cases and controls
$\hat{R} = \left\{g_i : |\hat{\Delta}_i| \leq |\hat{\Delta}|_{(K^*)}\right\}$
$\mathcal{S} = \left\{g_i : |\hat{\Delta}|_{(K^*)} < |\hat{\Delta}_i| \leq |\hat{\Delta}|_{(K^*+N_{PCR})}\right\}$
Create a custom qPCR array for the genes in $\mathcal{S}$
Spend remainder of budget collecting data on this custom qPCR array
$\hat{R} = \hat{R} \cup \left(\left\{g_i : |\hat{\Delta}_i| \geq \zeta\right\} \cap \mathcal{S}\right)$
return $\hat{R}$
```

We also present the BBD2 algorithm, Algorithm 8, which uses FDR controlled t-tests. Our intention with this algorithm is to highlight the subtle difference between using a threshold decision versus an adaptive policy like FDR control. In Section 6.3 we will show that it is easier for us to tune the threshold.

Algorithm 8 BBD2(B, C_{array}, C_{PCR}, N_{PCR})

```
$n = 2$
Collect $n$ microarrays from both cases and controls
$\hat{f}_\Delta = \text{FDE}\left(\{\hat{\Delta}_i\}, \Phi_\varepsilon(t) = e^{-\frac{t^2}{n}}\right)$
$n^*, \alpha_1^*, \alpha_2^* = \arg\max_{n,\alpha_1,\alpha_2} \text{E}_\Delta\left[\text{evaluation}\left(R, \text{BBD2-Core}\left(n, \alpha_1, \alpha_2\right)\right)\middle|\hat{f}_\Delta\right]$
while $n \neq n^*$ do
    $n = n + 1$
    Collect a microarray from both cases and controls
    $\hat{f}_\Delta = \text{FDE}\left(\{\hat{\Delta}_i\}, \Phi_\varepsilon(t) = e^{-\frac{t^2}{n}}\right)$
    $n^*, \alpha_1^*, \alpha_2^* = \arg\max_{n,\alpha_1,\alpha_2} \text{E}_\Delta\left[\text{evaluation}\left(R, \text{BBD2-Core}\left(n, \alpha_1, \alpha_2\right)\right)\middle|\hat{f}_\Delta\right]$
end while
$p^* = \max_s\{p_{(s)} : \forall r \leq s : p_{(r)} \leq \frac{r}{N}\alpha_1^*\}$
$\hat{R} = \{g_i : p_i \leq p^*\}$
$\mathcal{S} = \left\{g_i : |\hat{\Delta}|_{(|\hat{R}|)} < |\hat{\Delta}_i| \leq |\hat{\Delta}|_{(|\hat{R}|+N_{PCR})}\right\}$
Create a custom qPCR array for the genes in $\mathcal{S}$
Spend remainder of budget collecting data on this custom qPCR array
$p^* = \max_s\{p_{(r)} : \forall r \leq s : p_{(r)} \leq \frac{r}{N}\alpha_2^*\}$
$\hat{R} = \{g_i : p_i \leq p^*\}$
return $\hat{R}$
```

Algorithm 9 BBD2-Core(n, α_1, α_2)

```
Collect $n$ microarrays from both cases and controls
$p^* = \max_s\{p_{(s)} : \forall r \leq s : p_{(r)} \leq \frac{r}{N}\alpha_1\}$
$\hat{R} = \{g_i : p_i \leq p^*\}$
$\mathcal{S} = \left\{g_i : |\hat{\Delta}|_{(|\hat{R}|)} < |\hat{\Delta}_i| \leq |\hat{\Delta}|_{(|\hat{R}|+N_{PCR})}\right\}$
Create a custom qPCR array for the genes in $\mathcal{S}$
Spend remainder of budget collecting data on this custom qPCR array
$p^* = \max_s\{p_{(s)} : \forall r \leq s : p_{(r)} \leq \frac{r}{N}\alpha_2\}$
$\hat{R} = \{g_i : p_i \leq p^*\}$
return $\hat{R}$
```

Chapter 6

BBD Experiments

6.1 Comparison of TNAS Algorithms

Here we present an experiment to compare TNAS, TNAS-FDR, and TNAS-Top$K$. We set the FDR control to $\alpha = 0.01$ for the TNAS algorithm. Also, as a comparison we include a fourth algorithm, TNAS-Omniscient, that cheats by looking at the true Δ_i for each gene, and then selects the top K such that the evaluation is maximized, *i.e.*, it gets the maximum evaluation possible for both TNAS-FDR and TNAS-Top$K$.

We begin by using a sub-sampling experiment on the real microarray datasets used in Chapter 4. For each dataset, we take a random sub-sample of n men versus n women, and then present the data to the algorithms. All algorithms receive the same sub-sample of the data, and thus the observed performance difference is based entirely on how they construct their estimate of the relevant genes, $\hat{R}$. To set the ground truth for each gene we compute the estimated effect sizes, $\hat{\Delta}_i$ using all the data and consider those to be the true values. We set $\Delta^* = 1$ and $\beta = 1/10$. Recall that we use the generalized F score as our evaluation,

$$\text{evaluation}(R, \hat{R}) = \left(1 + \beta^2\right) \frac{\text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}} \quad . \qquad \text{(2.6 revisited)}$$

Figure 6.1 presents the results. We can see that TNAS and TNAS-FDR have similar performance. This is unsurprising, as the only difference is that TNAS-FDR will use the estimated distribution, $\hat{f}_\Delta$, from FDE to tune its level of FDR control, whereas TNAS uses a fixed value – if we can set a reasonable choice of α *a priori* then we cannot greatly improve it by using $\hat{f}_\triangle$ to tune α^*.[1] In comparison TNAS-Top$K$ performs worse than these algorithms. Recall that FDE will over-estimate the tails of the true distribution when computing $\hat{f}_\triangle$, and thus cause the algorithm to believe there are more relevant genes. This tricks TNAS-Top$K$ into returning too many genes – *e.g.*, for small values of n it has much higher recall than the other algorithms but suffers due to the lost precision. Interestingly, as n increases, TNAS-Top$K$ becomes less aggressive and returns fewer genes. We believe that with larger n, FDE returns better estimates, $\hat{f}_\triangle$, which allow TNAS-Top$K$ to see the value in trading-off recall for precision.

To further compare these algorithms, we ran simulation studies on synthetic datasets wherein we know the ground truth of everything, and all our assumptions hold. We consider the case where we have $N = 50\ 000$ genes observed in an experiment of 10 cases versus 10 controls. For each gene, g_i, we draw a Δ_i from $f_\triangle$. Then we draw 10 values from a normal $\mathcal{N}(\Delta_i, 1)$ distribution (for the cases) and 10 values from a $\mathcal{N}(0, 1)$ distribution (for the controls), and then use these values to estimate $\hat{\Delta}_i$, and compute corresponding p-values. For this experiment we consider three cases: 1) $f_\triangle$ is a normal distribution, 2) $f_\triangle$ is a Laplace distribution, and 3) $f_\triangle$ is a uniform distribution. Figure 6.2 shows the results. As we have fixed the number of microarrays in this experiment, we sweep the variance of $f_\triangle$, to increase the number of relevant genes.

Now we see a much stronger contrast in the algorithms' performances. TNAS now suffers heavily due its use of a constant level of FDR control $\alpha = 0.01$. TNAS-Top$K$ displays the same behaviour as it did with the real data. When the problem is hard, it seeks a good score by aggressively labelling many genes relevant, but as the problem gets easier (*i.e.*, for large values of σ^2 and b) it becomes less aggressive and starts trading-off recall for precision. Thus, we can conclude that if analyzing real data one should use TNAS-FDR.

From the synthetic datasets, we see that the difficulty of the problem is related

[1] Had we chosen a different value of α for TNAS, or chosen a different value of β for the evaluation, there would have been more contrast between the algorithms. In our subsequent experiments, keeping α and β constant hut changing the datasets, we will show a more pronounced difference between TNAS and TNAS-FDR.

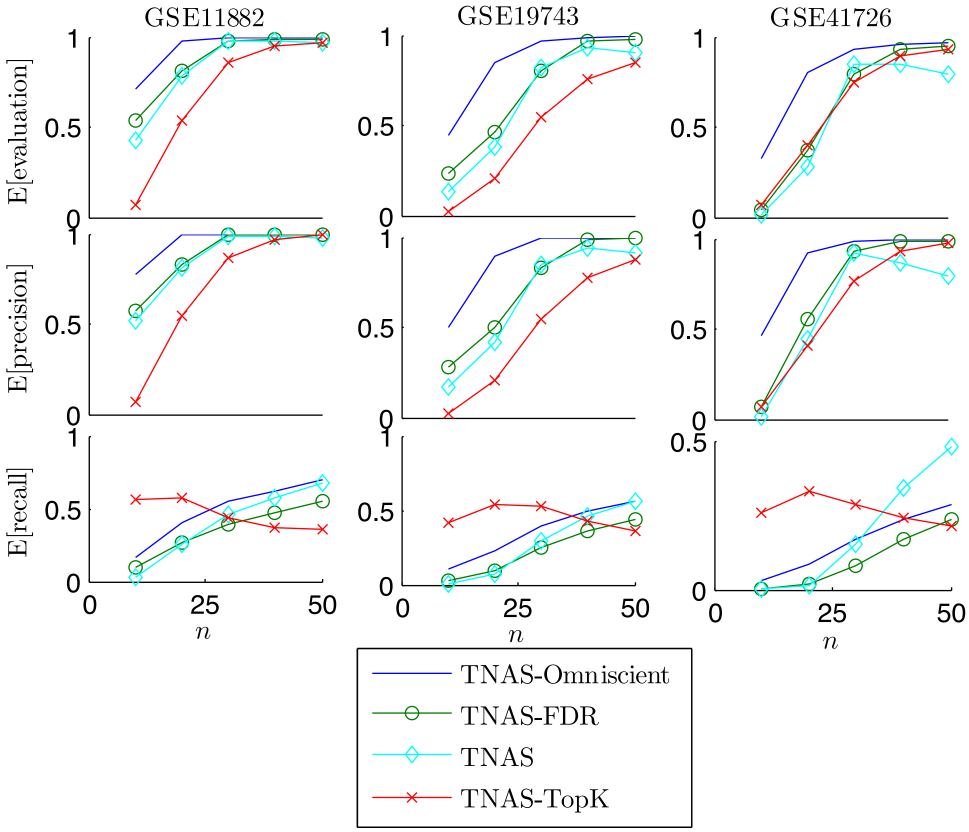

Figure 6.1: A comparison of our microarray only BBD algorithms, on sub-samples of real microarray data from GEO datasets. n is the number of microarrays used in the sub-sampling.

to the tails of f_Δ. Of our three synthetic models, the Laplace distribution has the heaviest tails, and we can see that it is the easiest to solve. Here we use the performance gap between TNAS-FDR and TNAS-Omniscient as a measure of easiness – *i.e.*, the problem must be easy if our algorithms can perform as well as the oracle algorithm. On the Laplace distributions, TNAS-FDR is almost indistinguishable from TNAS-Omniscient. Thus, we conclude that BBD problems on Laplace distributions are relatively easy. We claim that this relates to the tails of f_Δ because heavy tails mean that there will be more genes with very large values of $|\Delta_i|$. As our evaluation favours precision over recall, algorithms can do very well by only labelling as relevant the extremely obvious genes. The normal distribution has a slightly lighter tail, and thus there are less of these easy genes to pick off – note there is a gap between TNAS-FDR and TNAS-Omniscient. The uniform distribution has no tails at all, and thus no gene can have a very large $|\Delta_i|$. This makes it difficult to distinguish the relevant genes from the irrelevant ones – note the large gap between TNAS-Omniscient and TNAS-FDR.

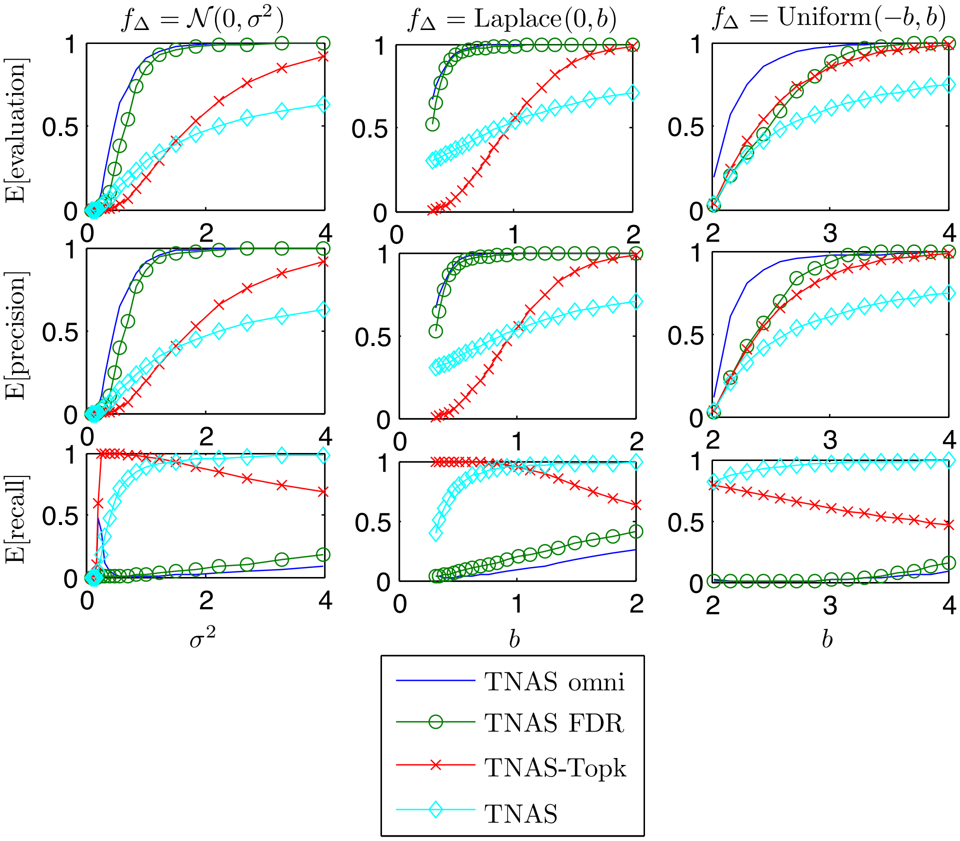

Figure 6.2: A comparison of our microarray only BBD algorithms, on synthetic data. For the fixed case of 10 versus 10 microarrays.

6.2 Comparison of BBD-Greedy To BBD1

We now compare BBD-Greedy to BBD1 using a small experiment on synthetic data. We consider a set with 20 genes, and we can either do a microarray at cost $C_{array} = 10$ to observe all of them, or a custom qPCR on 3 genes for a cost $C_{PCR} = 1$. If BBD-Greedy decides to collect qPCR, we force it to then spend its entire remaining budget testing the same 3 genes – thus both algorithms have the same choices available to them. While this experiment is not necessarily representative of real problems, we will argue that the results can be extrapolated to cases of interest.

For the experiment, we set f_Δ to be a normal distribution, $\mathcal{N}(0, \sigma^2)$, and we will sweep σ^2 and keep the budget fixed at $B = 200$. Thus, an algorithm may perform a microarray only study, with 10 per class, or a 9 versus 9 microarray study with an additional 10 versus 10 qPCR study, *etc*. Figure 6.3 shows the results.

We see that in all metrics, BBD1 outperforms BBD-Greedy. This is because BBD-Greedy considers the genes independently when evaluating its expected evaluation in Equation (5.1), whereas BBD1 uses $\hat{f}_\Delta$ to get an empirical estimate of its performance using our plate model – *i.e.*, it exploits the fact that it can get a crude estimate of true $|\Delta|_{(i)}$ even though it does not know how to map those values to the genes.

If we scale up the problem to more realistic instances, FDE will produce better estimates of f_Δ, which means BBD1 will be able to tune its parameters more effectively, and thus perform better. Conversely, as the number of genes increases BBD-Greedy will further suffer from considering the genes individually when assessing the utility of tests, and thus perform worse. Therefore, we do *NOT* recommend the use of BBD-Greedy in practice.

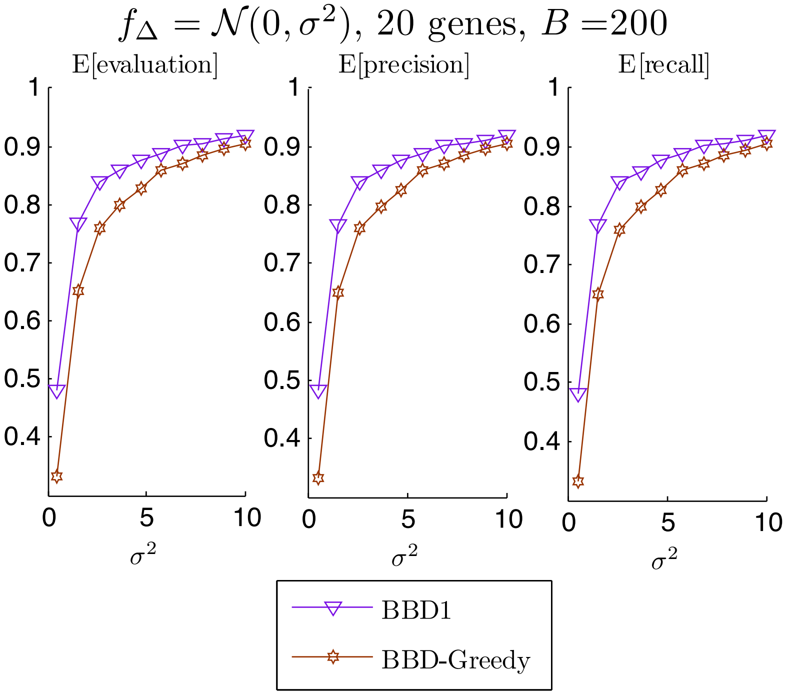

Figure 6.3: A comparison of BBD1 to BBD-Greedy over a class of small synthetic datasets. Microarrays cover all genes for cost $C_{array} = 10$, and qPCR covers $N_{PCR} = 3$ genes for cost $C_{PCR} = 1$.

6.3 Comparison of Custom qPCR Algorithms

Now we compare our custom qPCR algorithms, BBD1, and BBD2. As a comparison point, we include TNAS-FDR, to represent a microarray only study. Again, we use our real datasets to build a realistic experimental setup. When an algorithm decides to collect a microarray, we randomly select a microarray from a patient in the study. If an algorithm decides to collect qPCR data, we randomly draw two patients from the study and then average the gene expression values of the appropriate subset of genes on their microarrays, to produce a more accurate measurement of the gene expression values.[2]

For the experiment, we set the cost of a microarray to $C_{array} = 10$ and the cost of qPCR to $C_{PCR} = 1$, which covers $N_{PCR} = 100$ genes of our choosing. We vary different levels of the experimentation budget, B. For example, if the budget is $B = 200$ then the algorithm may perform a microarray only study, with 10 per class, or a 9 versus 9 microarray study with an additional 10 versus 10 qPCR study.

Figure 6.4 shows the results. We can see that the use of qPCR data does produce an improvement over the microarray only study – BBD1 and BBD2 generally have higher expected evaluation than TNAS-FDR. Interestingly these algorithms obtain their score quite differently: BBD1 does well by finding the same relevant genes as TNAS-FDR, but with a higher precision. BBD2 does well by labelling more genes as relevant and scoring very high recall, at the expense of some precision. We believe that BBD1 is displaying the behaviour sought by biologists looking for a principled way to conduct a checking study.

Now we compare algorithms on synthetic data to see what happens to our algorithms over a larger range of problems. We re-use the synthetic data experiment from Section 6.1. We consider the case where we have $N = 50\ 000$ genes, and an experimental budget $B = 200$; here again microarrays cost $C_{array} = 10$, and custom qPCRs cost $C_{PCR} = 1$, and cover $N_{PCR} = 100$ genes. For each gene, g_i, we draw a Δ_i from f_Δ, and we use our plate model from Section 2.1 to draw random

[2] We know that there is a fundamental difference between microarray and qPCR data – see Section 1.3. We do this because we know that qPCR data is more accurate than microarray and we do not have real qPCR data for all genes.

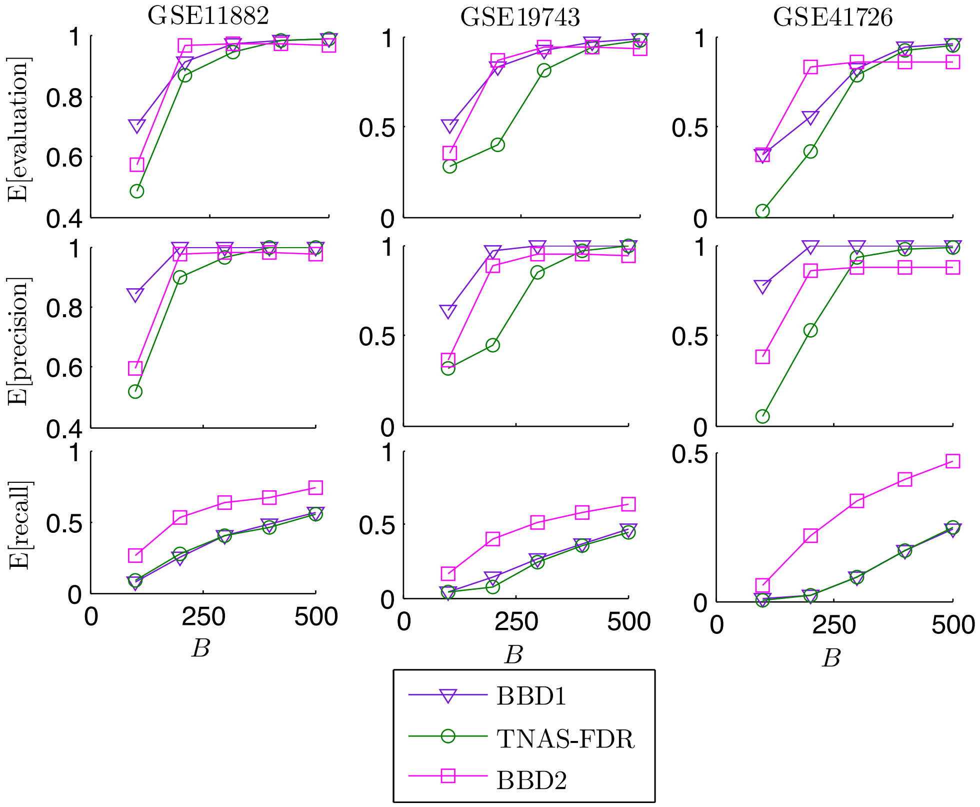

Figure 6.4: A comparison of our full BBD algorithms on realistic data derived from GEO datasets. We fix $C_{array} = 10$, $C_{PCR} = 1$, $N_{PCR} = 100$, and we sweep the budget, B.

variables for the expression values of the tests that the algorithms choose to collect. For this experiment we consider three cases: 1) f_Δ is a normal distribution, 2) f_Δ is a Laplace distribution, and 3) f_Δ is a uniform distribution.

Figure 6.5 shows the results. Shockingly, BBD2 is now clearly the worst of the three algorithms. Theoretically, it should be able to default into TNAS-FDR if it believes that collecting microarrays only is the best strategy. Since this is synthetic data, wherein we know the ground truth and all the modelling assumptions are true, these results show that it is more difficult to tune the parameters for BBD2 than BBD1, with the same limited amount of data. While both algorithms have the same numbers of free parameters to be tuned with the data, BBD2's parameters describe slightly more complex behaviours – thus making them harder to tune with the same data. For example, when selecting which genes to follow-up in custom qPCR, BBD1 simply puts the top K genes into $\hat{R}$ and moves on to the next genes, whereas BBD2 uses an FDR controlled t-test to determine which genes have low p-values and then focuses on those with p-values that were too large to pass that test. It seems reasonable that this policy represents a higher level of complexity. Also note that, because TNAS-FDR spends all the budget collecting microarrays, it has a more accurate estimate of f_Δ than either BBD1 or BBD2, which allows it to properly tune itself.

We posit that it will be difficult to construct an algorithm that performs notably better than BBD1. We believe this to be true because in order to beat BBD1 an algorithm will likely require a more sophisticated policy and that policy will have parameters to tune. However, as we have just shown that we could not effectively tune the parameters of BBD2 (which is only moderately more complex than BBD1), we believe that it will not be possible to tune the parameters of this algorithm in practice.

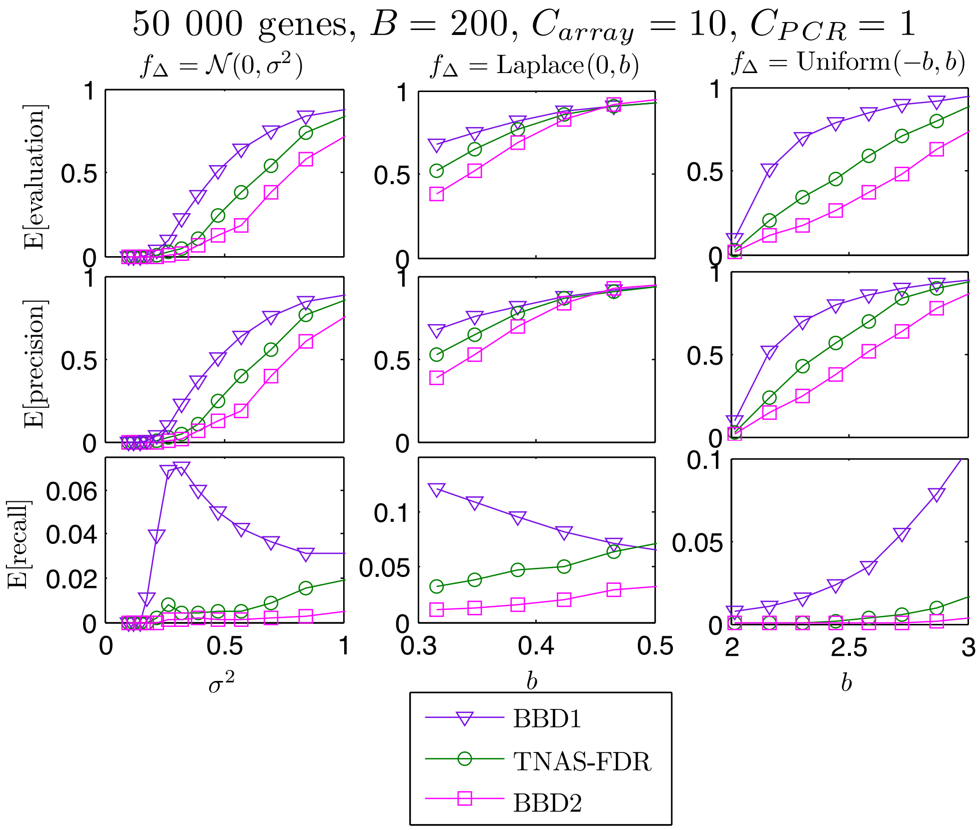

Figure 6.5: A comparison of our full BBD algorithms on synthetic data. We fix $C_{array} = 10$, $C_{PCR} = 1$, $N_{PCR} = 100$, and we sweep the appropriate free parameter of f_Δ.

6.4 Examining the Effect of Different Costs

To show that the performance advantage of BBD1 is not due to us unfairly constructing our experiments towards it, either by giving it an unfair cost advantage or by making them overly accurate, we now perform an experiment to observe the effect of the different costs, C_{PCR}. To make the comparison fair for this experiment, we use synthetic data, wherein both microarrays and qPCR have the same accuracy, *i.e.*, $\Delta_i^{(microarray)} = \Delta_i^{(PCR)}$. Thus, the observed differences will be due to the difference between C_{array} and C_{PCR}.

Here we fix the true effect size distribution, $f_{\Delta} = \mathcal{N}(0, 0.75)$, and sweep the cost of our custom qPCR arrays, C_{PCR}, maintaining $C_{array} = 10$ and $B = 200$. We chose this particular f_{Δ} because it was a point at which BBD1 was doing moderately well in our previous experiment, and thus could be potentially affected by the choice of C_{PCR}.

For interesting comparison points, we include TNAS-FDR and TNAS-Omniscient. Figure 6.6 shows the results. We begin examining this plot from the extreme right, where the costs of qPCR and microarrays have been set to be equal. Here we can see that the performance is very similar, but BBD1 has a slight advantage in recall. This is because it benefits from its decision to accept the top K genes as relevant, and then focus on the borderline genes. As we decrease the cost of qPCR BBD1 can afford more tests on the same budget and so it begins to see an advantage in precision, but the recall is largely unaffected. This is because no matter how cheap we make the qPCR arrays, they can only test $N_{PCR} = 100$ genes in our setup, and we may be unlucky in choosing those genes. However, we can be very precise about the genes we have picked.

Thus, regardless of the cost of qPCR BBD1 will outperform TNAS-FDR on our synthetic datasets. Interestingly, if the cost is low enough, it may also outperform TNAS-Omniscient. While this result is on synthetic data, we do believe it is likely to translate to real datasets, as we have seen in our experiments the assumptions of normal distributions seems pretty reasonable, and there is a good correspondence between BBD1 and TNAS-FDR across all our experiments. Unfortunately, without real ground truth values for our datasets, we cannot verify that this is actually happening. Hopefully, as the BBD framework gets adopted and researchers begin publishing both microarray and qPCR data with their studies we can verify this in the future.

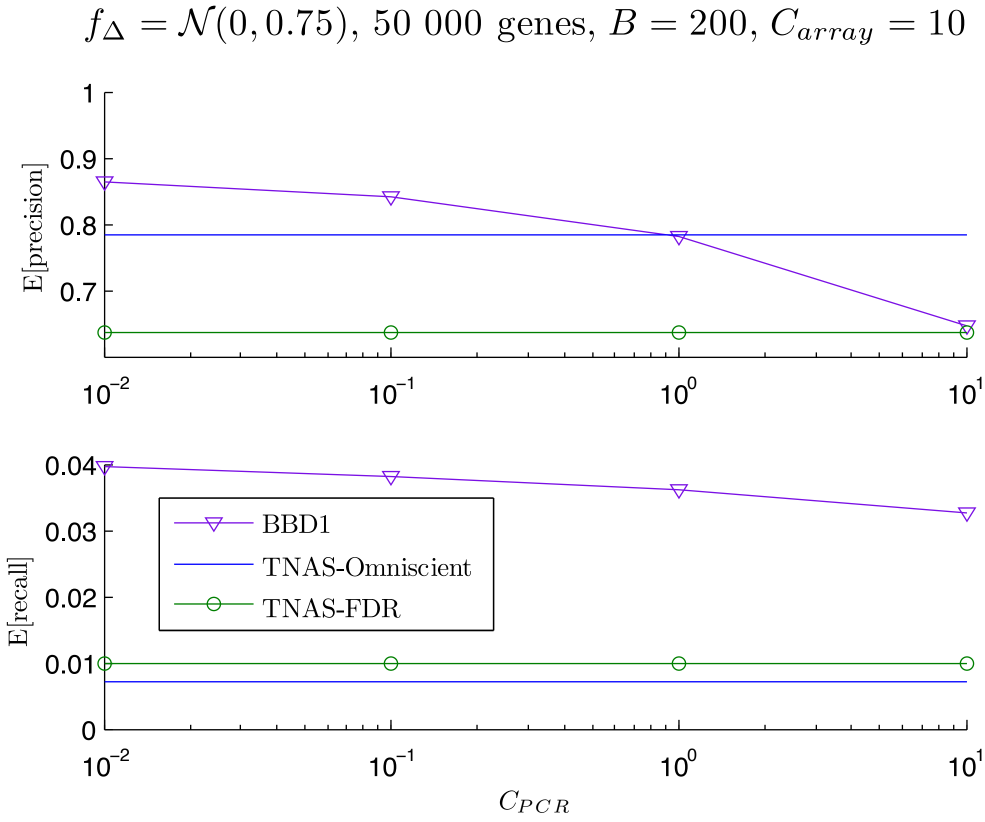

Figure 6.6: The effect of the cost of custom qPCR on the performance of BBD1. Results for TNAS-FDR and TNAS-Omniscient are also provided to compare against our best microarray only algorithm, and the best possible microarray only algorithm.

Chapter 7

Conclusions

In this thesis we presented the BBD problem as an alternative to using simple association studies to find genes that are biomarkers. To motivate our BBD framework, we showed that it addressed several concepts that are ambiguous with the traditional approaches taken within association studies.

1. In many association studies, p-values are used to determine which genes are biomarkers; if a gene has a sufficiently low p-value then it is a biomarker. However, we believe that there are no genes for which $\Delta_i = 0$, and thus given a sufficiently large amount of data, every gene can have a non-zero p-value. We argued that it should be preferable to assesses genes based on their standardized effect score, Δ_i, and called those with genes with $|\Delta_i| \geq \Delta^*$ "relevant". The benefit is that when we call a gene relevant we refer to an intrinsic, immutable, and interpretable property of the gene with respect to the phenotype. Thus, the goal of researchers should be to identify genes that are relevant.

2. The BBD framework provides an evaluation function that assesses the quality of the estimated set of relevant genes, $\hat{R}$, based on the set of truly relevant genes, R; with higher numbers being preferable. Having a clearly defined evaluation function for BBD is critical because it enables us to develop algorithms with the goal of maximizing that function, and thus provide the behaviour desired by biologists looking for relevant genes. We also note that this evaluation function can help us better understand the issue of irrepro-

ducibility in microarray studies. For example, if two studies on the same phenotype produced the sets of genes, $\hat{R}_1$ and $\hat{R}_2$, then due to statistical variations we would expect $|\hat{R}_1 \cap \hat{R}_2| \approx 0$ [35], but if they followed the same experimental and analytical procedures then we would expect that the quality of their sets be similar, *i.e.*, evaluation$(R, \hat{R}_1) \approx$ evaluation$(R, \hat{R}_2)$.

3. Currently most researchers agree that qPCR should be used to follow up microarray association studies, to validate the discovered biomarkers. Unfortunately, there is yet to be a consensus within the community of how this checking should be standardized. The BBD framework solves this by stating that microarray and qPCR technologies should be used jointly to produce the best estimate of the relevant genes, $\hat{R}$, that is possible given the available experimentation budget. In other words, microarray and qPCR should be combined with the goal of maximizing our evaluation function, subject to the budget constraints.

We showed, by number of datasets submitted to GEO, that there is a growing trend towards using microarrays for association studies; with approximately 10 000 datasets produced in 2013. Furthermore we also showed that the microarrays used per dataset is typically on the order of 10–15 arrays, and thus interpretation of the results will require follow up in qPCR. By posing the analysis as a BBD problem researchers can ensure that they are collecting their data in a cost efficient manner.

We also presented the problem of estimating the distribution of univariate statistics in microarray data and presented the FDE algorithm to solve it. Our FDE algorithm relied on the use of a plate model for gene expression values, and an assumption of normality. We showed, on real and synthetic data, that FDE estimates converge to the true distribution faster than a naive method that relies only on the Glivenko-Cantelli theorem to produce an empirical estimate from the observed values of $\hat{\Delta}_i$. In order for FDE to have worked well on real microarray data, our underlying plate model and assumption of normal distributions must have been well founded. Furthermore, we showed that good solutions to the BBD problem will incorporate our plate model and solve this density estimation problem as a subroutine. Our BBD1, BBD2, and TNAS-FDR algorithms used the plate model to tune their parameters and thus out-performed our TNAS and BBD-Greedy algorithms, which did not use any plate model.

We showed that our BBD1 algorithm provides the most robust solution, amongst the algorithms presented here, to the BBD problem. In all of our experiments, in both real and synthetic data, it had very good performance – in most experiments it had the highest evaluation score of all algorithms. We believe that BBD1 displays the behaviour in combining microarray and qPCR data that researchers performing association and checking studies desire.

We claim that it is extremely difficult to construct an algorithm that uses microarrays and custom qPCR data that can significantly outperform BBD1. The issue is that such an algorithm would require a policy that is more elaborate than BBD1's to describe its behaviour, but if the policy is too complex the algorithm will be unable to tune its parameters to behave effectively. We showed that our BBD2 algorithm used a policy that was only a slightly more complicated policy than BBD1's, and it was unable to tune its parameters in synthetic data experiments where all modelling assumptions were true. In all of these experiments, BBD2 performed worse than TNAS-FDR, which corresponds to the special case where BBD2 collects only microarray data, *i.e.*, BBD2 failed to realize that it could have scored a higher evaluation by collecting only microarrays. All the while, in the same experiments, BBD1 signficantly outperformed both TNAS-FDR, and BBD2 by appropriately combining microarrays with custom qPCR.

7.1 Recommended Future Works

In this thesis we presented the BBD problem and provided the TNAS-FDR, and BBD1 algorithms as good algorithms for solving it; TNAS-FDR should be used when retro-analyzing microarray data, and BBD1 should be used when we can collect both microarrays and custom qPCR. However, these algorithms may be viewed as good initial first steps. Here we will outline several interesting next steps that could be taken towards analyzing and understanding the BBD problem.

1. All of our successful BBD algorithms made use of the FDE algorithm as a subroutine. Thus, by improving FDE we can improve the performance of our BBD algorithms. Furthermore, we showed that BBD2 could not properly tune its parameters given the estimated distribution, $\hat{f}_\Delta$, from FDE, and thus the construction of more complex BBD algorithms will mandate an associated improvement to the FDE algorithm.

2. Our BBD1 algorithm showed that the use of a custom qPCR array can improve upon the performance of an algorithm that only uses microarrays. However, we did not consider the possibility of using more than one custom qPCR array. It would be interesting to see if an algorithm could do better using multiple custom qPCR arrays.

3. We presented the BBD problem from the perspective of having a limited budget. However, it may be enlightening to also consider BBD from the perspective of being mandated to discover a specific amount of relevant genes. For example, if we are required to find 100 relevant genes, what would be the most cost effective manner to do so? And how much would it cost? Algorithms for this task would likely make the claim that, with high probability, they can reliably find the required relevant genes. It would be interesting to show how the high probability requirement relates to the budget. Among other things, answering these questions would allow researchers to assess the feasibility of their research goals, and write more accurate grant proposals.

4. Rather than searching for the set of relevant genes, researchers interested in biomarkers for diagnostics may wish to search for a set of genes that can be used used to build a good classifier. In other words, we may wish to solve the problem of collecting data to build a classifier for the phenotype, while being efficient with the budget. We note that this sounds similar to the problems that have been previously addressed in the field of active learning. However, an interesting distinction here is that the classifier may be intended to work using a specific technology, such as qPCR but the learning can utilize different technologies, such as microarrays, as a cost effective means to do

feature selection.

5. While our presentation of the BBD problem was restricted to just gene expression data, it would be interesting to approach the BBD problem from a systems biology perspective and utilize multiple ’omics technologies. This approach may also involve exploiting datasets from related studies on GEO, and datamining databases such as KEGG, GO, *etc*.

Bibliography

[1] C.J. Albers, R.C. Jansen, J. Kok, O.P. Kuipers, and S.AFT van Hijum. Simage: simulation of dna-microarray gene expression data. *BMC Bioinformatics*, 7(1):205, 2006.

[2] D.B. Allison, X. Cui, G.P. Page, and M. Sabripour. Microarray data analysis: from disarray to consolidation and consensus. *Nat. Rev. Genet.*, 7(1):55–65, January 2006.

[3] N. Alon, N. Cesa-Bianchi, C. Gentile, and Y. Mansour. From bandits to experts: A tale of domination and independence. *NIPS*, 2013.

[4] M. Ashburner, C.A. Ball, J.A. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T Eppig, M.A. Harris, D.P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G.M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. *Nat. Genet.*, 25(1):25–29, 2000.

[5] R. Askey. Some characteristic functions of unimodal distributions. *J. Math. Anal. Appl.*, 50(3):465–469, 1975.

[6] J.-Y. Audibert and S. Bubeck. Best arm identification in multi-armed bandits. *COLT*, 2010.

[7] P. Bachman and D. Precup. Greedy Confidence Pursuit : A Pragmatic Approach to Multi-bandit Optimization. In *Mach. Learn. Knowl. Discov. Databases*, pages 241–256. Springer Berlin Heidelberg, 2013.

[8] A. Badanidiyuru, R. Kleinberg, and A. Slivkins. Bandits with knapsacks. *FOCS*, 2013.

[9] C.A. Ball, A. Brazma, H. Causton, S. Chervitz, R. Edgar, P. Hingamp, J.C. Matese, H. Parkinson, J. Quackenbush, M. Ringwald, S.-A. Sansone, G. Sherlock, P. Spellman, C. Stoeckert, Y. Tateno, R. Taylor, J White, and N. Winegarden. Submission of microarray data to public repositories. *PLoS Biol.*, 2(9):e317, 2004.

[10] K. Basso, A.A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera, and A. Califano. Reverse engineering of regulatory networks in human b cells. *Nat. Genet.*, 37(4):382–390, 2005.

[11] M. Bastani, L. Vos, N. Asgarian, J. Deschenes, K. Graham, J. Mackey, and R. Greiner. A machine learned classifier that uses gene expression data to accurately predict estrogen receptor status. *PLoS One*, 8(12):e82144, 2013.

[12] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc.*, 57(1):289–300, 1995.

[13] B.M. Bolstad, F. Collin, K.M. Simpson, R.A. Irizarry, and T.P. Speed. Experimental design and low-level analysis of microarray data. *Int. Rev. Neurobiol.*, 60:25–58, 2004.

[14] A-L. Boulesteix and M. Slawski. Stability and aggregation of ranked gene lists. *Brief. Bioinform.*, 10(5):556–68, 2009.

[15] A. Brazma, P. Hingamp, J. Quackenbush, G. Sherlock, P. Spellman, C. Stoeckert, J. Aach, W. Ansorge, C.A. Ball, H.C. Causton, T. Gaasterland, P. Glenisson, F.C.P. Holstege, I.F. Kim, V. Markowitz, J.C. Matese, H. Parkinson, A. Robinson, U. Sarkans, S. Schulze-Kremer, J. Stewart, R. Taylor, J. Vilo, and M. Vingron. Minimum information about a microarray experiment (miame) toward standards for microarray data. *Nat. Genet.*, 29(4):365–371, 2001.

[16] J.W. Brown, R.V. Churchill, and M. Lapidus. *Complex variables and applications*, volume 7. McGraw-Hill New York, 1996.

[17] S. Bubek, T. Wang, and N. Viswanathan. Multiple Identifications in Multi-Armed Bandits. *ICML*, 2013.

[18] S.A Bustin, V. Benes, J.A. Garson, J. Hellemans, J. Huggett, M. Kubista, R. Mueller, T. Nolan, M.W. Pfaffl, G.L. Shipley, J. Vandesompele, and C.T. Wittwer. The miqe guidelines: minimum information for publication of quantitative real-time pcr experiments. *Clinical Chemistry*, 55(4):611–622, 2009.

[19] N.A. Campbell and J.B. Reece. *Biology*. Benjamin Cummings, 8th edition, 2008.

[20] G.J. Cane. Linear Estimation of Parameters of the Cauchy Distribution Based on Sample Quantiles. *J. Am. Stat. Assoc.*, 69(345):243–245, 1974.

[21] R.F. Chuaqui, R.F. Bonner, C.J.M. Best, J.W. Gillespie, M.J. Flaig, S.M. Hewitt, J.L. Phillips, D.B. Krizman, M.A. Tangrea, M. Ahram, W. M. Linehan, V. Knezevic, and M.R. Emmert-Buck. Post-analysis follow-up and validation of microarray experiments. *Nat. Genet.*, 32:509–14, 2002.

[22] X. Cui and G.A. Churchill. Statistical tests for differential expression in cDNA microarray experiments. *Genome Biol.*, 4(4):210, 2003.

[23] H.A. David and H.N. Nagaraja. *Order statistics*. Wiley Online Library, 1970.

[24] K. Deng, R. Greiner, and S. Murphy. Budgeted learning for developing personalized treatment. *ICMLA*, 2014.

[25] K. Deng, J. Pineau, and S. Murphy. Active learning for personalizing treatment. *ADPRL*, 2011.

[26] B.J. Deroo and K.S Korach. Estrogen receptors and human disease. *J. Clin. Invest.*, 116(3):561–570, 2006.

[27] F. Dieterle, B. Riefke, G. Schlotterbeck, A. Ross, H. Senn, and A. Amberg. Nmr and ms methods for metabonomics. In *Drug Safety Evaluation*, pages 385–415. Springer, 2011.

[28] C. Ding and C.R. Cantor. Quantitative analysis of nucleic acids-the last few years of progress. *J. Biochem. Mol. Biol.*, 37(1):1–10, 2004.

[29] W. Ding, T. Qin, X.-D. Zhang, and T.-Y. Liu. Multi-armed bandit with budget constraint and variable costs. *AAAI*, 2013.

[30] K.K. Dobbin, Y. Zhao, and R.M. Simon. How large a training set is needed to develop a classifier for microarray data? *Clin. Cancer Res.*, 14(1):108–114, 2008.

[31] A. Dupuy and R.M. Simon. Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. *J. Natl. Cancer Inst.*, 99(2):147–157, 2007.

[32] R. Edgar, M. Domrachev, and A. E. Lash. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res.*, 30(1):207–10, 2002.

[33] J. Edmonds. Submodular functions, matroids, and certain polyhedra. *Edited by G. Goos, J. Hartmanis, and J. van Leeuwen*, page 11, 1970.

[34] B. Efron, R. Tibshirani, J. D. Storey, and V. Tusher. Empirical Bayes Analysis of a Microarray Experiment. *J. Am. Stat. Assoc.*, 96(456):1151–1160, 2001.

[35] L. Ein-Dor, O. Zuk, and E. Domany. Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *Proc. Natl. Acad. Sci. U. S. A.*, 103(15):5923–8, 2006.

[36] P.D. Ellis. *The essential guide to effect sizes: Statistical power, meta-analysis, and the interpretation of research results*. Cambridge University Press, 2010.

[37] R.A. Fisher. Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. *Biometrika*, pages 507–521, 1915.

[38] R.A. Fisher. On the 'probable error' of a coefficient of correlation deduced from a small sample. *Metron*, 1:3–32, 1921.

[39] D. Golovin and A. Krause. Adaptive submodularity: A new approach to active learning and stochastic optimization. *COLT*, 2010.

[40] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Mach. Learn.*, 46:389–422, January 2002.

[41] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*, volume 2. Springer, 2009.

[42] D.W. Huang, B.T. Sherman, and R.A. Lempicki. Systematic and integrative analysis of large gene lists using david bioinformatics resources. *Nat. Protoc.*, 4(1):44–57, 2008.

[43] D.W. Huang, B.T. Sherman, and R.A. Lempicki. Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. *Nucleic Acids Res.*, 37(1):1–13, 2009.

[44] P.J. Hurd and C.J. Nelson. Advantages of next-generation sequencing versus the microarray in epigenetic research. *Brief. Funct. Genomics*, page elp013, 2009.

[45] J. P.A. Ioannidis, D. B. Allison, C. A. Ball, I. Coulibaly, X. Cui, A. C. Culhane, M. Falchi, C. Furlanello, L. Game, G. Jurman, J. Mangion, T. Mehta, M. Nitzberg, G. P. Page, E. Petretto, and V. van Noort. Repeatability of published microarray gene expression analyses. *Nat. Genet.*, 41(2):149–55, 2009.

[46] M. Jansche. A Maximum Expected Utility Framework for Binary Sequence Labeling. In *ACL*, 2007.

[47] W.E. Johnson, C. Li, and A. Rabinovic. Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, 8(1):118–127, 2007.

[48] S.-H. Jung. Sample size for fdr-control in microarray data analysis. *Bioinformatics*, 21(14):3097–3104, 2005.

[49] S. Kalyanakrishnan, A. Tewari, P. Auer, and P. Stone. PAC Subset Selection in Stochastic Multi-armed Bandits. *ICML*, 2012.

[50] M. Kanehisa, S. Goto, Y. Sato, M. Kawashima, M. Furumichi, and M. Tanabe. Data, information, knowledge and principle: back to metabolism in kegg. *Nucleic Acids Res.*, 42(D1):D199–D205, 2014.

[51] R.M. Karp. *Reducibility among combinatorial problems*. Springer, 1972.

[52] S. Khan and R. Greiner. Finding discriminatory genes: a methodology for validating microarray studies. *ICDM workshop on Data Mining for Biomedical Applications (BioDM)*, 2013.

[53] S. Khan and R. Greiner. The budgeted biomarker discovery problem: A variant of association studies. *AAAI workshop on Modern Artificial Intelligence for Health Analytics (MAIHA)*, 2014.

[54] S. Khan and R. Greiner. Budgeted transcript discovery: A framework for joint exploration and validation studies. *BIBM*, 2014.

[55] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[56] A. Krause and D. Golovin. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 3:19, 2012.

[57] F. Lanfranco, A. Kamischke, M. Zitzmann, and E. Nieschlag. Klinefelter's syndrome. *Lancet*, 364(9430):273–283, 2004.

[58] J.T. Leek, R.B. Scharpf, H.C. Bravo, D. Simcha, B. Langmead, W.E. Johnson, D. Geman, K. Baggerly, and R.A. Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nat. Rev. Genet.*, 11(10):733–739, 2010.

[59] L. Li, B. Póczos, C. Szepesvári, and R. Greiner. Budgeted Distribution Learning of Belief Net Parameters. *ICML*, 2010.

[60] W. Li and Y. Yang. Zipf's Law in Importance of Genes for Cancer Classification Using Microarray Data. *J. Theor. Biol.*, 29(4):539–551, 2002.

[61] B. Lippe. Turner syndrome. *Endocrinol. Metab. Clin. North Am.*, 20(1):121–152, 1991.

[62] A.A. Margolin, I. Nemenman, K. Basso, C. Wiggins, R.D. Stolovitzky, G.and Dalla-Favera, and A. Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7(Suppl 1):S7, 2006.

[63] J.S. Morey, J.C. Ryan, and F.M. Van Dolah. Microarray validation: factors influencing correlation between oligonucleotide microarrays and real-time PCR. *Biol. Proced. Online*, 8(1):175–93, 2006.

[64] P. Müller, G. Parmigiani, C. Robert, and J. Rousseau. Optimal sample size for multiple testing: the case of gene expression microarrays. *J. Am. Statist. Assoc.*, 99(468):990–1001, 2004.

[65] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An Analysis of Approximations For Maximizing Submodular Set Functions. *Math. Program.*, 14(1):265–294, 1978.

[66] M. Nykter, T. Aho, M. Ahdesmäki, P. Ruusuvuori, A. Lehmussola, and O. Yli-Harja. Simulation of microarray data with realistic characteristics. *BMC Bioinformatics*, 7(1):349, 2006.

[67] A.V. Oppenheim, R.W. Schafer, and J.R. Buck. *Discrete-time signal processing: Second Edition*. Prentice-hall, 1989.

[68] A. Papoulis and S.U. Pillai. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 2002.

[69] M.S. Pepe, R. Etzioni, Z. Feng, J.D. Potter, M.L. Thompson, M. Thornquist, M. Winget, and Y. Yasui. Phases of biomarker development for early detection of cancer. *J. Natl. Cancer Inst.*, 93(14):1054–1061, 2001.

[70] V. Popovici, W. Chen, B.G. Gallas, C. Hatzis, W. Shi, F.W. Samuelson, Y. Nikolsky, M. Tsyganova, A. Ishkin, T. Nikolskaya, K.R. Hess, V. Valero, D. Booser, M. Delorenzi1, G.N. Hortobagyi, L. Shi, W.F. Symmans, and L. Pusztai. Effect of training-sample size and classification difficulty on the accuracy of genomic predictors. *Breast Cancer Res.*, 12(1):R5, 2010.

[71] W. H. Press, S. A. Teukolsky, W.T. Vetterling, and B. P. Flannery. *Numerical Recipes 3rd edition: The art of scientific computing*. Cambridge University Press, 2007.

[72] J.G. Proakis. *Digital Communication Systems: 4th Edition*. Wiley Online Library, 2000.

[73] J.C. Rockett and G.M. Hellmann. Confirming microarray data–is it really necessary? *Genomics*, 83(4):541–9, 2004.

[74] B. Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11, 2010.

[75] L. Shi, W. Tong, H. Fang, U. Scherf, J. Han, R.K. Puri, F.W. Frueh, F.M. Goodsaid, L. Guo, Z. Su, T. Han, J.C. Fuscoe, Z.A. Xu, T.A. Patterson, H. Hong, Q. Xie, R.G. Perkins, J.J. Chen, and D.A. Casciano. Cross-platform comparability of microarray technology: intra-platform consistency and appropriate data analysis procedures are essential. *BMC Bioinformatics*, 6(Suppl 2):S12, 2005.

[76] B.W. Silverman. *Density estimation for statistics and data analysis*. CRC press, 1986.

[77] G.K. Smyth. Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Stat. Appl. Genet. Mol. Biol.*, 3(1):1–25, 2004.

[78] C. Stretch, S. Khan, N. Asgarian, R. Eisner, S. Vaisipour, S. Damaraju, K. Graham, O.F. Bathe, H. Steed, R. Greiner, and V.E. Baracos. Effects of sample size on differential gene expression, rank order and prediction accuracy of a gene signature. *PLoS One*, 8(6):e65380, 2013.

[79] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl. Acad. Sci. U. S. A.*, 102(43):15545–50, 2005.

[80] R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*. MIT Press, 1998.

[81] Duncan C. Thomas, Graham Casey, David V. Conti, Robert W. Haile, Juan Pablo Lewinger, and Daniel O. Stram. Methodological Issues in Multistage Genome-Wide Association Studies. *Stat. Sci.*, 24(4):414–429, November 2009.

[82] R Tibshirani. Regression Shrinkage and Selection via the lasso. *J. R. Stat. Soc. Ser. B*, 58(1):267—-288, 1996.

[83] R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc. Natl. Acad. Sci. U. S. A.*, 99(10):6567–72, 2002.

[84] S. Tong and D. Koller. Active learning for parameter estimation in bayesian networks. *NIPS*, 2000.

[85] S. Tong and D. Koller. Active Learning for Structure in Bayesian Networks. *IJCAI*, 2001.

[86] L. Tran-Thanh, A.C. Chapman, A. Rogers, and N.R. Jennings. Knapsack based optimal policies for budget-limited multi-armed bandits. *AAAI*, 2012.

[87] V.G. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *Proc. Natl. Acad. Sci. U.S.A.*, 98(9):5116–5121, 2001.

[88] A. W. Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge University Press, 2000.

[89] J.A. Wagner, S.A. Williams, and C.J. Webster. Biomarkers and surrogate end points for fit-for-purpose development and regulatory evaluation of new drugs. *Clin. Pharmacol. Ther.*, 81(1):104–107, 2007.

[90] A. Wald. Sequential Tests of Statistical Hypotheses. *Ann. Math. Stat.*, 16(2):117–186, 1945.

[91] Y. Wang, C. Barbacioru, F. Hyland, W. Xiao, K.L. Hunkapiller, J. Blake, F. Chan, C. Gonzalez, L. Zhang, and R.R. Samaha. Large scale real-time pcr validation on gene expression measurements from two commercial long-oligonucleotide microarrays. *BMC Genomics*, 7(1):59, 2006.

[92] C. Wei, J. Li, and R.E. Bumgarner. Sample size for detecting differentially expressed genes in microarray experiments. *BMC Genomics*, 5:87, 2004.

[93] N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT Press, 1949.

[94] D. Witten and R. Tibshirani. A comparison of fold-change and the t-statistic for microarray data analysis. Technical Report 650, 2007.

[95] H. Yang, C.A. Harrington, K. Vartanian, C.D. Coldren, R. Hall, and G.A. Churchill. Randomization in laboratory procedure is key to obtaining reproducible microarray results. *PLoS One*, 3(11):e3724, 2008.

Chapter 8

Probability Distributions Used

This appendix is intended to be a convenient reference for the distributions used within the thesis. All of this material is readily found in [68].

8.1 Basic Properties of the Distributions

Here we list the basic properties of the distributions we have used to describe random variables in this thesis. Specifically, we list the PDF, CDF, characteristic function, mean, and variance.

Note that, while we did not use the Chi-squared distribution, we include it here as it will be used as a stepping stone in our analysis of the distribution of $\hat{\Delta}$ in Section 8.2.

Some of the functions make use of the gamma function,

$$\Gamma(x) = \begin{cases} (x-1)! & x \in \mathbb{N} \\ \int_0^\infty y^{x-1} e^{-y} dy & \text{otherwise} \end{cases} .$$

General properties for the normal distribution, $x \sim \mathcal{N}\left(\mu, \sigma^2\right)$

$f_x\left(x\right)$	=	$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\mu-x)^2}{2\sigma^2}}$
$F_x\left(x\right)$	=	no closed form expression exists[1][2]
$\Phi_x\left(t\right)$	=	$e^{j\mu t - \frac{1}{2}\sigma^2 t^2}$
$\mathrm{E}[x]$	=	μ
$\mathrm{Var}(x)$	=	σ^2

[1] It is common to use $\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt$, but we do not use this notation, as by our conventions Φ denotes a characteristic function.

[2] While this function is well defined, it is difficult to compactly write in the table. It is a function composed of integrals that cannot be analytically solved.

General properties for the Laplace distribution, $x \sim$ Laplace (μ, b)

$$\begin{aligned}
f_x(x) &= \tfrac{1}{2b} e^{-|x-\mu|/b} \\
F_x(x) &= \begin{cases} \tfrac{1}{2} e^{(x-\mu)/b} & x < \mu \\ 1 - \tfrac{1}{2} e^{-(x-\mu)/b} & x \geq \mu \end{cases} \\
\Phi_x(t) &= \tfrac{1}{1+b^2 t^2} e^{j\mu t} \\
\mathrm{E}[x] &= \mu \\
\mathrm{Var}(x) &= 2b^2
\end{aligned}$$

General properties for the Cauchy distribution, $x \sim$ Cauchy (x_0, γ)

$$\begin{aligned}
f_x(x) &= \tfrac{1}{\pi\gamma} \left(1 + \left(\tfrac{x-x_0}{\gamma}\right)^2\right)^{-1} \\
F_x(x) &= \tfrac{1}{\pi} \arctan\left(\tfrac{x-x_0}{\gamma}\right) + \tfrac{1}{2} \\
\Phi_x(t) &= e^{jx_0 t - \gamma|t|} \\
\mathrm{E}[x] &= \text{undefined}^{3} \\
\mathrm{Var}(x) &= \text{undefined}^{4}
\end{aligned}$$

General properties for the uniform distribution, $x \sim \mathcal{U}(a, b)$

$$\begin{aligned}
f_x(x) &= \begin{cases} \tfrac{1}{b-a} & a \leq x \leq b \\ 0 & \text{otherwise} \end{cases} \\
F_x(x) &= \begin{cases} 0 & x < a \\ \tfrac{x-a}{b-a} & a \leq x < b \\ 1 & b \leq x \end{cases} \\
\Phi_x(t) &= \tfrac{e^{jtb} - e^{jtb}}{jt(b-a)} \\
\mathrm{E}[x] &= \tfrac{1}{2}(a+b) \\
\mathrm{Var}(x) &= \tfrac{1}{12}(b-a)^2
\end{aligned}$$

General properties for the chi-squared distribution, $x \sim \chi^2(\nu)$

$$\begin{aligned}
f_x(x) &= \tfrac{1}{2^{\nu/2}\Gamma(\nu/2)} x^{\nu/2-1} e^{-x/2} \\
F_x(x) &= \text{no closed form expression exists}^{2} \\
\Phi_x(t) &= (1 - j2t)^{\nu/2} \\
\mathrm{E}[x] &= \nu \\
\mathrm{Var}(x) &= 2\nu
\end{aligned}$$

[3] By undefined we mean that the integral $\int_{-\infty}^{\infty} x f_x(x)\, dx$ cannot be solved.

[4] Because $\mathrm{E}[x]$ is undefined, AND $\mathrm{E}[x^2] = \infty$ then the $\mathrm{Var}(x) = \mathrm{E}[x^2] - \mathrm{E}[x]^2$ is also undefined.

		General properties for the non-central t-distribution, $x \sim T(\nu, \mu)$
$f_x(x)$	=	no closed form expression exists[2]
$F_x(x)$	=	no closed form expression exists[2]
$\Phi_x(t)$	=	no closed form expression exists[2]
$\mathrm{E}[x]$	=	$\mu\sqrt{\frac{\nu}{2}}\frac{\Gamma((\nu-1)/2)}{\Gamma(\nu/2)} \quad \nu > 1$
$\mathrm{Var}(x)$	=	$\frac{\nu(1+\mu^2)}{\nu-2} - (\mathrm{E}[x])^2 \quad \nu > 2$

8.2 The Distribution of $\hat{\Delta}$

We now derive the distribution of $\hat{\Delta}$. We begin by formally stating how the non-central t-distribution arises, as a function of two random variables. If $x \sim \mathcal{N}(\mu, 1)$ and $y \sim \chi^2(\nu)$ are independent random variables then the ratio,

$$t = \frac{x}{\sqrt{y/\nu}} \quad , \tag{8.1}$$

is said to follow a non-central t-distribution with ν degrees of freedom and non-centrality parameter μ, ***i.e.***, $t \sim T(\nu, \mu)$.

Theorem 5. *If the expression values of gene g_i, observed by test t, are normally distributed with common variance ($\sigma_i^2 = \sigma_{i,s}^2$), $\psi_{i,s} \sim \mathcal{N}(\mu_{i,s}, \sigma_i^2)$, and we have observed $n^{(t)}$ values per class, then the scaled estimated effect size follows a non-central t-distribution,*

$$\hat{\Delta}_i\sqrt{n/2} \sim T\left(2n-2, \Delta\sqrt{n/2}\right) \quad , \tag{8.2}$$

where we have suppressed the dependence on t for notational convenience.

Proof of Theorem 5. For convenience of notation, we suppress the superscript notation denoting the test t.

We construct the random variables $x = \frac{\hat{\mu}_{i,1}-\hat{\mu}_{i,0}}{\sigma_i}$, and $y = (n-1)\frac{\hat{\sigma}_{i,1}^2+\hat{\sigma}_{i,0}^2}{\sigma_i^2}$. From the normal assumption, x and y are independent random variables[1] with distributions,

$$x \sim \mathcal{N}\left(\frac{\mu_{i,1}-\mu_{i,0}}{\sigma_i}, \frac{2}{n}\right) = \mathcal{N}\left(\Delta_i, \frac{2}{n}\right)$$
$$y \sim \chi^2(2n-2)$$

[1] Technically by the normal assumption $\hat{\mu}_{i,s}$ and $\hat{\sigma}_{i,s}^2$ are independt random variables [68, Section 8.4], and thus x and y are independent by extension.

The result follows from our definition of $\hat{\Delta}$ in Equation (2.4) with a few lines of algebra.

$$\begin{aligned}\hat{\Delta}_i\sqrt{n/2} &= \frac{\hat{\mu}_{i,1}-\hat{\mu}_{i,0}}{\sqrt{\left(\hat{\sigma}^2_{i,1}+\hat{\sigma}^2_{i,0}\right)/2}}\sqrt{\frac{n}{2}} \\ &= \frac{\hat{\mu}_{i,1}-\hat{\mu}_{i,0}}{\sqrt{\left(\hat{\sigma}^2_{i,1}+\hat{\sigma}^2_{i,0}\right)/2\sigma^2_i}}\sqrt{\frac{n}{2\sigma^2_i}} \\ &= \frac{x}{\sqrt{y/(2n-2)}}\sqrt{\frac{n}{2}} \qquad (8.3)\end{aligned}$$

Equation (8.3) is the ratio of a normal random variables with distribution $\mathcal{N}\left(\Delta\sqrt{n/2},1\right)$ and the appropriately scaled root of a chi-squared random variable with $\nu = 2n-2$ degrees of freedom, as required by the definition in Equation (8.1). □

Lastly we claim that we can approximate the non-central t-distribution with a normal distribution,

$$T\left(2n-2,\Delta\sqrt{n/2}\right) \approx \mathcal{N}\left(\Delta\sqrt{n/2},1\right) \quad . \qquad (8.4)$$

This claim is pretty reasonable as it a slight variation on the well accepted limit, $\lim_{\nu\to\infty} T\left(\nu,0\right) = \mathcal{N}\left(0,1\right)$. Rather than showing this mathematical we just compare their plots. Figure 8.1 shows both distributions from Equation (8.4) for the case where $\Delta = 1$ and $n = \{5, 10, 15, 20\}$. Even for these relatively small values of n we can see that the normal approximation is quite good.

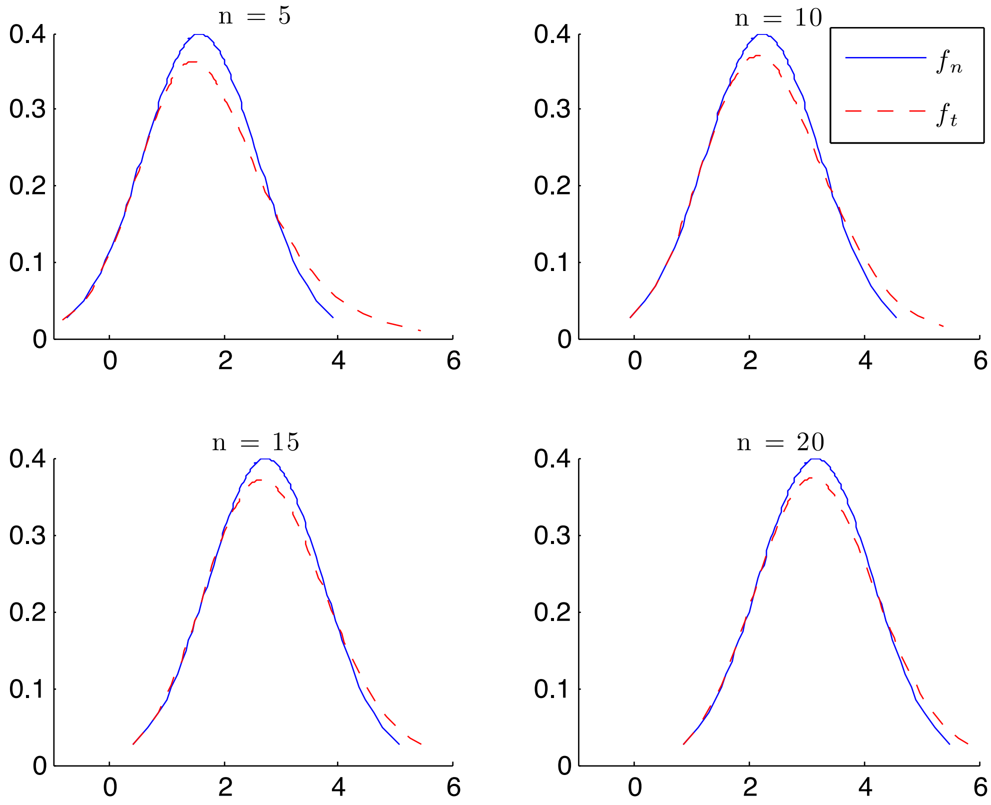

Figure 8.1: Comparison of the non-central t-distribution (red) and the normal distribution (blue).

Chapter 9

Tuning Algorithms for BBD

Now that we have shown that we can estimate f_Δ in Chapter 4, we can use our plate model to tune parameters for the algorithms we have constructed. Knowing f_Δ, we can reason about what values of Δ we should see given N draws from the distribution. Then we can analyze how the algorithm will behave when confronting this situation. Thus, conceptually we can analytically tune the parameters to optimize the algorithms performance, *i.e.*, maximize the given evaluation function.

Unfortunately, even for very simple algorithms the analysis is infeasible. Consider our TNAS algorithm which uses a fixed level of FDR control, α, to label the genes as relevant. In order to analyze such an algorithm we may need to compute the probability that the algorithm labels exactly K genes as relevant,

$$\begin{aligned}
&P\,(\text{TNAS labels } K \text{ genes relevant}) \\
&= P\left(p_{(K)} \leq \frac{K}{N}\alpha\right) P\left(p_{(K+1)} > \frac{K+1}{N}\alpha\right) \\
&= F_{p_{(K)}}\left(\frac{K}{N}\alpha\right)\left(1 - F_{p_{(K+1)}}\left(\frac{K+1}{N}\alpha\right)\right) \quad . \qquad (9.1)
\end{aligned}$$

Equation (9.1) is deceptively simple as it would seem that we only need to evaluate the two CDFs, $F_{p_{(K)}}$ and $F_{p_{(K+1)}}$. Unfortunately we do not have these CDFs, but we can compute very good approximations for them. If we know f_Δ, then we know all the Δ_i values we expect to see across the genes. For each Δ_i, Equation (8.2) gives us the distribution of $\hat{\Delta}_i$, which we can transform to get the distribution of the corresponding p-vaue, f_{p_i}. Then we can compute the distribution of ordered statics on independent, non-identically distributed random variables

[23][Chapter 2.8]. Unfortunately, this process takes $\mathrm{O}\big(N^2\binom{N}{K}\big)$ work, per evaluation of Equation (9.1). Alternatively we can use our plate model to simulate TNAS in $\mathrm{O}(N + N\log(N))$.[1]Thus, rather than computing the probability analytically, it is more efficient compute it empirically by averaging over several runs of the algorithm on synthetic datasets.

To tune an algorithm, such that it will perform effectively, we must consider several parameterizations and for parameterization we must perform some simulations to get an empirical estimate of its expected evaluation. In this appendix we will present some heuristic search techniques that can be used to reduce the computational effort spent in parameter tuning.

9.1 Golden Search

Here we consider the general problem of finding the value x to maximize some unimodal function $h\left(x\right)$, assuming that we have a bounded interval on the possible values of x. Specifically we wish to solve the problem,

$$x^* = \arg\max_{x\in[a,b]} h\left(x\right) \quad . \tag{9.2}$$

This problem will arise when tuning parameters for our BBD algorithms. For example, to tune TNAS-FDR we would set,

$$\begin{aligned} h\left(x\right) &= \mathrm{E}_\Delta\left[\text{evaluation}\left(R, \text{TNAS}(B, C_{array}, x)\right)\middle|\hat{f}_\Delta\right] \\ a &= 0 \\ b &= 1 \quad . \end{aligned}$$

The golden search algorithm is a divide and conquer method that works by iteratively breaking the interval $[a, b]$ into sub-intervals, which it can quickly check to see which contains the maximum. The algorithm maintains an ordered triplet of values (x_1, x_2, x_3) for which it know the values $h\left(x_1\right)$, $h\left(x_2\right)$, and $h\left(x_3\right)$. By considering a new point $x_4 \in [x_1, x_3]$, and evaluating $h\left(x_4\right)$, the algorithm can

[1] We can dramatically reduce the work done by TNAS in practice, by noting that the computational bottleneck is from sorting the p-values. As most genes are not relevant, we need only sort the bottom few. Thus, we can reduce the work to $\mathrm{O}(N + K\log(K))$, where K is the number of genes passing FDR control.

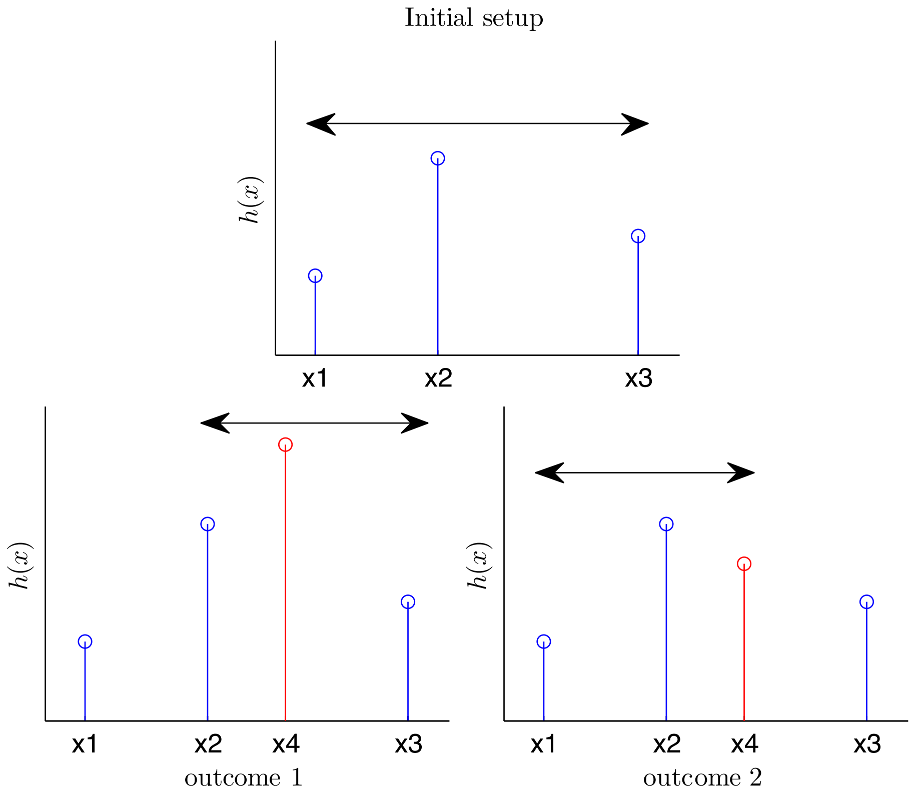

Figure 9.1: One iteration of the golden search algorithm. Double arrows represent the intervals wherein $x^* = \arg\max h(x)$ may lie.

determine which interval the maximum lies in by comparing all the $h(x_i)$ values. Figure 9.1 illustrates the two possible outcomes of the comparison, for the case where $x_4 > x_2$. [2] Either, $h(x_4) \geq h(x_2)$ and we can conclude that $x^* \in [x_2, x_3]$ because we know $h(x)$ is unimodal, Figure 9.1 outcome 1. Or, $h(x_4) < h(x_2)$ and we can conclude that $x^* \in [x_1, x_4]$, Figure 9.1 outcome 2.

Now we consider how the algorithm selects the point x_4. Regardless of the value of $h(x_4)$, we would like to the algorithm to make the same amount of progress per iteration, *i.e.*, we desire the intervals $[x_1, x_4]$ and $[x_2, x_3]$ to have the same length. Algorithm 10 presents pseudocode for the golden search algorithm, named so as it uses the well known golden ratio to select the point x_4. The algorithm accepts as inputs: the function to maximize $h(\cdot)$, and upper and lower bounds to begin the

[2] The case case where $x_4 < x_2$ behaves similarly.

parameter search, x_1 and x_3 respectively. This algorithm is guaranteed to find the optimal setting x^* for any unimodal objective function $h(x)$.

Algorithm 10 uses a check for unimodality to terminate our search at line 8. Theoretically this should never happen as we began by stating that $h\left(x\right)$ is unimodal. However, for our purposes we will not have access to the true $h\left(x\right)$, and will rely on an empirical estimate generated from simulating in our plate model – when the interval is sufficiently small statistical variations on $h\left(x\right)$ will cause the algorithm to terminate.

Algorithm 10 Golden Search($h\left(\cdot\right)$, a, b)

```
 1: $r = \frac{1+\sqrt{5}}{2}$
 2: $x_1 = a$ and $x_3 = b$
 3: $x_2 = x_1 + \frac{x_3 - x_1}{1+r}$
 4: evaluate $h\left(x_1\right)$, $h\left(x_2\right)$, $h\left(x_3\right)$)
 5: while $true$ do
 6:     $x_4 = x_1 + \left(x_3 - x_2\right)$
 7:     evaluate $h\left(x_4\right)$
 8:     if $h\left(x\right)$ is bi-modal then
 9:         break
10:     end if
11:     switch $true$
12:         case $\left(x_4 > x_2\right)$ and $\left(h(x_4) > h\left(x_2\right)\right)$
13:             $x_1 = x_2$
14:             $x_2 = x_4$
15:         case $\left(x_4 > x_2\right)$ and $\left(h\left(x_4\right) < h\left(x_2\right)\right)$
16:             $x_3 = x_4$
17:         case $\left(x_2 > x_4\right)$ and $\left(h\left(x_4\right) < h\left(x_2\right)\right)$
18:             $x_1 = x_4$
19:         case $\left(x_2 > x_4\right)$ and $\left(h\left(x_4\right) > h\left(x_2\right)\right)$
20:             $x_3 = x_2$
21:             $x_2 = x_4$
22:     end switch
23: end while
24: return $x^* = x_2$
```

We note that there are other interval search algorithms that could be used [71], but golden search is particularly appropriate for two reasons:

1. It has slightly faster convergence towards the value x^*, in terms of the number of evaluations of $h\left(x_4\right)$ to the reduction of the interval $[x_1, x_3]$.

2. For every evaluation of $h\left(x\right)$ we can immediately reduce the interval $x^* \in [a, b]$, whereas methods like ternary search require two evaluations in order to reduce the interval.

In the following branch and bound search we will benefit from this, as faster updates to the intervals will improve our bounds and help us to avoid necessary evaluations of $h\left(x\right)$.

9.2 Branch and Bound Search

The golden search method works well for tuning functions with a single parameter, and thus is well suited for our TNAS-FDR and TNAS-Top$K$ algorithms. But, it clearly will not work for BBD1. Here we present a branch and bound technique that we can use for tuning more complicated BBD algorithms. We will use the example of parameter tuning for BBD1 as running example, but the approach applies equally well to BBD2, and is likely to generalize to other BBD algorithms.

BBD1 must solve the problem,

$$n^*, K^*, \zeta^* = \arg\max_{n,K,\zeta} \mathrm{E}_\triangle \left[\text{evaluation}\left(R, \text{BBD1-Core}\left(n, K, \zeta\right)\right) \middle| \hat{f}_\triangle \right] \quad .$$

At first this may seem like it we must solve a single optimization problem. But if we observe how the BBD1-Core algorithm operates, the choice of ζ depends on the values of n and K, the choice of K depends on n, and n can be set arbitrarily. Thus, we can decompose the problem to,

$$\begin{aligned} n^*, K^*, \zeta^* &= \arg_{n,K,\zeta} \left[\max_n \left\{ \max_K \left(\max_\zeta h(n, K, \zeta) \right) \right\} \right] \qquad (9.3) \\ h(n, K, \zeta) &= \mathrm{E}_\triangle \left[\text{evaluation}\left(R, \text{BBD1-Core}\left(n, K, \zeta\right)\right) \middle| \hat{f}_\triangle \right] \end{aligned}$$

With this decomposition we can now use Algorithm 10 to solve each of the component maximizations, as they involve sweeping a single parameter.

We can get further improvements by noting that using interval searches on parameters allows us to use branch and bound search to solve our optimization problem. For example, suppose are considering a parameterization with n, K, and we know $\zeta \in [\zeta_1, \zeta_2]$. Furthermore, because we use simulations in our plate model to get an empirical estimate of the expected evaluation for a given parameterization we can also compute the precision and recall. We note that by increasing ζ it is harder for genes to be added to $\hat{R}$, and thus we increase the precision and decrease the recall (in expectation). Thus we construct an upper bound on the evaluation score by using the best possible values of the precision and recall.

$$\begin{aligned}
\text{precision}_1 =& \mathrm{E}_\Delta \left[\text{precision of BBD1-Core}\left(n, K, \zeta_1\right) \middle| \hat{f}_\Delta \right] \\
\text{recall}_2 =& \mathrm{E}_\Delta \left[\text{recall of BBD1-Core}\left(n, K, \zeta_2\right) \middle| \hat{f}_\Delta \right] \\
& \max_{\zeta \in [\zeta_1, \zeta 2]} \mathrm{E}_\Delta \left[\text{evaluation}\left(R, \text{BBD1-Core}\left(n, K, \zeta\right)\right) \middle| \hat{f}_\Delta \right] \\
& \leq \left(1 + \beta^2\right) \frac{\text{precision}_1 \times \text{recall}_2}{\beta^2 \times \text{precision}_1 + \text{recall}_2}
\end{aligned}$$

When considering two different parameterizations, one with n and K_1, and one with n and K_2, we may not need to find the optimal ζ_1 if we have observed some some value of ζ_2 where the second parameterization has a higher evaluation than the bound on the first, then we can abandon the parameter search on ζ_1, as the second parameterization must be better. Thus, we focus our effort into finding the values of ζ_2^* for the given values of n_2 and K_2. Naturally, we can apply this same idea to comparing parameterizations using different values of n.

Lastly, we note that BBD1 is often checking to see if it should collect more microarrays. If the decision is to collect more microarrays, then the actual values of n^*, K^*, and ζ^*, are irrelevant, because the algorithm will update $\hat{f}_\Delta$ and then resolve Equation (9.3). Thus, we can get a further savings in practice by terminating the search, whenever it has been determined that more microarrays will be collected.

The important things to take note of to generalize these ideas to tune BBD algorithms is:

1. There is likely a natural hierarchy to the parameters which we can exploit for the optimization.

2. By using interval searches on a single parameter, we can use the precision and recall at the end points of the interval to bound the function over the interval.

3. By quickly identifying intervals that the optimal parameters reside in, we can focus on promising areas of the parameter space and solve the problem faster.

Chapter 10

Proofs

In this section we provide proofs for the theorems presented in Chapter 4.

Proof of Theorem 2: Characteristic functions of symmetric distributions. If $f_x\left(x\right) = f_x\left(-x\right)$ then with some manipulations:

$$\begin{aligned}
\Phi_x\left(t\right) &= \int_{-\infty}^{\infty} f_x\left(x\right) e^{jxt} dx \\
&= \int_{-\infty}^{0} f_x\left(x\right) e^{jxt} dx + \int_{0}^{\infty} f_x\left(x\right) e^{jxt} dx \\
&= \int_{0}^{\infty} f_x\left(-x\right) e^{-jxt} dx + \int_{0}^{\infty} f_x\left(x\right) e^{jxt} dx \\
&= \int_{0}^{\infty} f_x\left(x\right) \left(e^{-jxt} + e^{jxt}\right) dx \\
&= 2\int_{0}^{\infty} f_x\left(x\right) \cos\left(xt\right) dx \\
&= \Phi_x\left(-t\right)
\end{aligned}$$

□

Proof of Theorem 3 requires the following Lemmas.

Lemma 6 (Convolution Theorem [67, Equation 2.196] **).** *If $h(x)$ and $g(x)$ are arbitrary real valued functions, and both $\int_{-\infty}^{\infty} h(x)e^{jxt}dx$ and $\int_{-\infty}^{\infty} g(x)e^{jxt}dx$ are defined, then,*

$$\int_{-\infty}^{\infty} \left(h(x) * g(x)\right)(x) \times e^{jxt} dx = \left(\int_{-\infty}^{\infty} h(x)e^{jxt}dx\right)\left(\int_{-\infty}^{\infty} g(x)e^{jxt}dx\right) \tag{10.1}$$

Lemma 7 (Sifting [67, Equation 4.4])**.** *If $\delta(t)$ is the Dirac delta function, and $h(t)$ is an arbitrary real valued function, then,*

$$\int_{-\infty}^{\infty} h(t)\delta(t) = h(0) \tag{10.2}$$

Lemma 8 (Impulse Train [67, Equation 4.5])**.** *If $\delta(t)$ is the Dirac delta function, and $T \in \mathbb{R}$ is a constant, then,*

$$T\sum_{k\in\mathbb{Z}} \int_{-\infty}^{\infty} e^{jxt}\delta(x-kT)dx = \sum_{k\in\mathbb{Z}} \delta\left(t-\frac{k}{T}\right) \tag{10.3}$$

Lemma 9 (Inverse Transform [68, Equation 5.76])**.** *If $h(x)$ is an arbitrary function then,*

$$h(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty} h(x')e^{jx't}dx'\right)e^{-jxt}dt \tag{10.4}$$

Proof of Theorem 3: Aliasing. From our approximation in Equation 4.10, the limit as $T \to \infty$ is,

$$\begin{aligned}\lim_{T\to\infty} \hat{f}_{\Delta}(x;\tau,T) &= \lim_{T\to\infty}\frac{\tau}{2\pi}\sum_{k\,:\,|k\tau|\le T} e^{-jxk\tau}\Phi_{\hat{\Delta}}(k\tau)/\Phi_{\varepsilon}(k\tau)\\ &= \frac{\tau}{2\pi}\sum_{k\in\mathbb{Z}} e^{-jxk\tau}\Phi_{\Delta}(k\tau)\end{aligned}$$

By Lemma 7,

$$\begin{aligned}&= \frac{\tau}{2\pi}\sum_{k\in\mathbb{Z}}\int_{-\infty}^{\infty} e^{-jxt}\Phi_{\Delta}(t)\,\delta(t-k\tau)dt\\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty}\Phi_{\Delta}(t)\left(\tau\sum_{k\in\mathbb{Z}}\delta(t-k\tau)\right)e^{-jxt}dt\end{aligned}$$

By Lemma 6 (using Equation (4.6) and Lemma 8 for the components $h(x)$ and $g(x)$),

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\left(f_{\Delta}(x) * \sum_{k\in\mathbb{Z}}\delta\left(x-\frac{k}{\tau}\right)\right)(x)\,e^{jxt}dx\right]e^{-jxt}dt$$

By Lemma 9,

$$\begin{aligned}&= \left(f_{\Delta}(x) * \sum_{k\in\mathbb{Z}}\delta\left(x-\frac{k}{\tau}\right)\right)(x)\\ &= \int_{-\infty}^{\infty} f_{\Delta}(x-y)\left(\sum_{k\in\mathbb{Z}}\delta\left(y-\frac{k}{\tau}\right)\right)dy\\ &= \sum_{k\in\mathbb{Z}}\int_{-\infty}^{\infty} f_{\Delta}(x-y)\,\delta\left(y-\frac{k}{\tau}\right)dy\end{aligned}$$

By Lemma 7

$$= \sum_{k \in \mathbb{Z}} f_\Delta \left(x - \frac{k}{\tau} \right)$$

□

Proof of Theorem 4: Windowing. From our approximation in Equation 4.10, the limit as $\tau \to 0$ is,

$$\begin{aligned} \lim_{\tau \to 0} \hat{f}_\Delta \left(x; \tau, T\right) &= \lim_{\tau \to 0} \frac{\tau}{2\pi} \sum_{k \,:\, |k\tau| \leq T} e^{-jxk\tau} \Phi_{\hat{\Delta}} \left(k\tau\right) / \Phi_\varepsilon \left(k\tau\right) \\ &= \frac{1}{2\pi} \int_{-T}^{T} e^{-jxt} \Phi_\Delta \left(t\right) dt \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \Phi_\Delta \left(kt\right) \mathbb{1} \left(|t| \leq T\right) e^{-jxkt} dt \end{aligned}$$

By Lemma 6,

$$= \frac{1}{2\pi} \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} \left(f_\Delta \left(x'\right) * \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathbb{1} \left(|t| \leq T\right) e^{-jxt} dt \right) \left(x\right) e^{-jxt} dx \right) e^{-jxt} dt$$

By Lemma 9,

$$= \left(f_\Delta \left(x'\right) * \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathbb{1} \left(|t| \leq T\right) e^{-jxt} dt \right) \left(x\right)$$

By calculus [16],

$$\left(f_\Delta \left(x'\right) * \frac{\sin(Tx)}{\pi x} \right) \left(x\right)$$

□