University of Alberta

Detecting, correcting, and preventing the batch effects in multi-site data, with a focus on gene expression Microarrays

by

Saman Vaisipour

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

Spring 2014
Edmonton, Alberta

Abstract

Gene expression microarrays are widely used to better understand the complex biological mechanisms inside cells. One of the main obstacles of applying statistical learning algorithms to microarray data is the large gap between the number of features (p) and the number of available instances (n), *i.e.,* the *"large p, small n"* challenge. This thesis explores two ways to deal with this challenge.

One approach is to increase n by combining similarly appropriate microarray data sets together. This is appealing as there are now many publicly available microarray studies. The main problem of this approach is the batch effect, *i.e.,* the influence of non-biological factors on expression intensities that can confound the biological signal. As a result, combining gene expression studies without correcting for batch effects may lead to misleading findings.

This thesis proposes a novel batch correction algorithm, called batch effect correction using canonical correlation analysis (BECCA), that assumes the batch effect is due to additive independent confounding factors and so utilizes canonical correlation analysis to separate technical bias from the measured biological signal. We compare BECCA to various existing batch correction algorithms using several real-world gene expression studies and find that BECCA has similar performance. The key advantage of utilizing BECCA, compared to other similar performing algorithms, is its flexibility, as BECCA allows the user to adjust how much common signal to preserve across the batches and how much batch related signal to remove from each one by changing the values of BECCA parameters.

The second approach to batch correction considers the wisdom of reducing p by selecting a subset of genes. Our experiments suggest that some genes in microarray data sets contain very little biological signal, *i.e.,* including only these genes in the calculations makes all specimens highly correlated, regardless of their tissue of origin or disease state. It is, therefore, desirable to identify and remove these misleading genes before conducing downstream analysis or batch correction. For this purpose,

we propose an efficient algorithm to extend the single-study variance-based gene selection method to a multi-study gene selection algorithm. Our empirical results show this feature selection algorithm outperforms other algorithms in reducing the destructive influence of batch effects.

Acknowledgements

I like to think of a PhD program as an adventurous exploration. You start from a certain point but you never know where you will end up and what challenges you will face. The ultimate goal of a PhD (besides writing a thesis) is to learn how to explore independently without getting lost.

The most important factor in how one finishes this journey is the journey's guide. The guide should not walk too fast to leave you behind, or too slow to leave you confused. S/he should also be patient and wise to understand what the student is going through. Without a great guide finishing the journey, if not impossible, would be extremely difficult.

On my journey Russell Greiner was a great guide. He taught me numerous lessons with his actions. I'd like to think that he made me a better person. I would like to express my sincere gratitude to him for everything.

I would like to thank Andy Sims, whom I got to know 3 years ago when I read one of his articles and asked him a couple of questions. Since then, he has been extremely helpful with guiding me and raising many invaluable points in our discussions. His dedication and commitment is extraordinary, I hope one day I will meet him and have a chance to thank him in person.

I would also like to thank my committee members, Dale Schuurmans and David Wishart. Their helpful comments were vital for finishing this work. My gratitude also goes to Paul Boutro's lab members at OICR.

Getting through my dissertation required more than academic support, and I have many, many people to thank, for listening to and, at times, having to tolerate me over the past years. I express my deepest gratitude to Arash for his friendship. Without his rational advises throughout my life, I would get lost at many different stages of my life. I hope I will have him beside me in the future. I am also grateful to many other people, mostly in the CS department: Amir-Ali, Amir-Masoud, Amin, Babak, Bret, Davoud, Fatemeh, Fariba, Farzaneh, Hootan, Hossein, Jake, Joan, Kiana, Laurie, Mays, Meysam, Mohsen, Nasimeh, Parisa, Reihaneh, Reza N, Reza S, Saimak, Shadi, Shahab, Shahin, (C) Shahni, Sheehan, Thatchin, Yasin, and Yavar. These people turned the cold nights of Edmonton into the warmest memories of my life.

I want to make a very special mention of those friends at home who stood by me when I left, when I came back and all the time in between: Armin, Farough, and Sina. Even though they were thousands of kilometers away, I always knew I have them by my side.

Most importantly, none of this could have happened without my family. My grandmother always offered her encouragement through her simple honest prayers. My parents, Mehri and Ayoub, are the best parents I could wish for. They raised my siblings and me to be caring and kind. They spent everything they had into our education. This dissertation stands as a testament to their unconditional love and encouragement. Without them, I would not be writing these lines. I also like to thank Pakhshan and Sirwan. Having such a great sister and brother makes me feel safe. If it was not for their sacrifices, I would not be able to come back and continue my PhD, for that, I am forever grateful.

Lastly, I would also like to thank Mariana and appreciate her presence in my life. Even in my native language, I cannot express how significant her role is in my life. She was such a great companion on this journey. I cannot wait for our future journeys.

Table of Contents

Chapter 1

Introduction

Measuring gene expression levels (transcription of genes) using microarrays is now a common practice for finding new insights about different diseases and conditions. Examples of the wide range of microarray applications includes: finding new biomarkers by exploring the relationship between the expression of genes and the state of a disease, predicting the clinical outcomes such as the response of cancer patients to particular adjuvant treatments or the chance of rejecting a transplanted organ, and discovering processes taking place inside certain types of cells. Statistical learning algorithms have been widely utilized to assist scientists with exploring gene expression data.

One of the main challenges of elucidating meaningful results from microarray experiments is that the number of observations, also know as sample size (n), is typically significantly smaller than the number of measured features (p). The value of n is often less than 100 while p is typically more than 10,000 [1]. In this thesis, we consider two different approaches for this problem:

1. Increase n by combining several studies together.

2. Reduce p by applying a feature selection algorithm.

Combining available studies (that are appropriately "similar") together to increase n is an appealing solution to decrease the gap between n and p, especially when there are many publicly available data sets. However, this task can be challenging if the different studies are done by different labs using different technologies as these technical differences confound the biological signal of interest and influence the measured expression intensities in a way that they might not be comparable across studies. The influence of technical differences on the expression values is

known as the *"batch effect" (BE)* [2]. Before one can combine such datasets, it is necessary to apply some correction procedure to these studies in order to remove the batch effect [3] and make their expression values numerically comparable. In this thesis we explore ways to detect, and remove, batch effects. We first study available batch effect removal methods proposed for correcting gene expression studies.

We introduce a new batch removal method based on canonical correlation analysis (CCA) called BECCA. CCA is a well known statistical algorithm that is used to find the directions of high correlation between two sets of features that have been observed for a same set of instances [4–6]. Similar to principal component analysis (PCA), which is used to reduce the dimensionality of one data set at a time, CCA typically is used to reduce the number of features between a pair of data sets by projecting them onto directions that makes their data maximally correlated [7]. The main requirement of our novel approach, BECCA, is that the data sets that are being combined together contain a common correlated signal and that the signal is influenced by batch effects independently in each data set. BECCA finds the directions of high correlation between two data sets and projects them onto these directions. The residuals are presumably batch effects, which can then be removed from each data set independently. We empirically compare BECCA's performance to other available BE correction methods and find results indicating that BECCA corrects the batch effect as effectively as other state of the art algorithms. One main benefits of utilizing BECCA is its flexibility; users can decide how much common signal to preserve across the batches and how much batch-related signal to remove from each one by changing the values of BECCA's parameters, *i.e.,* to control the "level of batch correction".

Another approach to deal with this "large p small n" challenge is to perform feature selection, to reduce the number of genes that are used in downstream analysis. The benefits of using gene selection algorithms to identify differentially expressed genes is well-known [8]. An effective gene selection method is able to find a subset of genes that are least influenced by the confounding factors and contain most of the biological signal of interest. The findings in this thesis suggest that utilizing such feature selection methods will improve the performance of learning algorithms not only by reducing the gap between p and n but also by removing genes whose expression intensity is more severely influenced by non-biological confounding factors, note that utilizing these gens for learning will degrade the performance. We studied

the relationship between the batch effect and the feature set used when combining gene expression studies.

Our experiments also indicate that there are some genes that implicitly claim that any pair of expression profiles are highly correlated regardless of their tissue of origin or disease state. In other words, if we only take into account these genes to calculate the correlation coefficients between profiles, then all the scores will be almost 1, irrespective of the biological properties of profiled specimens. We believe neglecting to filter these genes out of the data set can badly harm the effectiveness of batch effect correction algorithms, especially the "matrix factorization-based" correction algorithms [2], as well as the downstream analysis.

There are gene selection algorithms that can be applied successfully to single study cases, including the variance-based gene selection that ranks genes based on their variance assuming that high variance is caused by the biological variations of instances. We propose an efficient algorithm to extend the single study variance-based gene selection algorithm to the multi-study case. Our proposed algorithm receives the variance-based ranking of several studies and generates one ranking for genes by combining them. We compare this feature selection algorithm with other available feature selection algorithms and evaluate them based on the reduced confounding influence of batch effects. Our experiments indicates this algorithm outperforms other feature selection algorithms. One might think of the feature selection step as a pre-batch correction stage in order to remove misleading genes whose expression intensities are highly influenced by confounding technical factors.

The literature shows that some correction algorithms distort the data, leading to a poorer result for the downstream analysis than the original batch-affected data [2, 9]. Therefore, evaluating the performance of batch correction algorithms to ensure that these "corrections" do not further distort the data is as important as correcting the batch effect itself [2]. The performance evaluation is not a trivial task as batch correction algorithms have two competing goals, namely removing influence of technical factors on data and imposing minimum modification of true biological differences between samples. Many of the qualitative evaluation methods assess batch effect correction algorithms only according to one of these goals and fail to consider the other one. In this thesis we study available evaluation methods and introduce two new evaluation methods by modifying available algorithms to ensure they evaluate batch correction algorithms for both expected goals. This

thesis introduces a method that can measure the severity of batch effects influences by utilizing a biological labeling associated with the instances. By comparing the intensity of BE before and after applying a particular BE correction algorithm, we can evaluate an algorithm's performance under different circumstances.

As many authors have pointed out, the effectiveness of batch correction process depends on how the technical factors confound the biological signal. Gagnon-Bartsch and Speed [9] mentioned that, if the unwanted technical variation is not orthogonal to the biological signal of interest, then a batch correction algorithm might not be able to avoid also removing useful biological information. This mainly depends on the experimental design and the processing order of samples, which determines how the technical factors influence the expression values. If the instances (samples) are grouped according to their biological properties, *i.e.,* batches are formed according to the biological properties, then the technical factors will confound the biological signal and it will become impossible to correct the batch effect without losing useful biological signal. In this thesis we provide general guidelines for experimental design to reduce the confounding influence of batch effects and increase the likelihood that correction algorithms will be able to remove the batch effects without distorting the biological signal.

The thesis is organized as follows: Chapter 2 discusses the wide range of learning algorithms applied to gene expression studies and the "large p small n" challenge that is faced by all algorithms. Chapter 3 looks at the batch effect and explains what prevents us from simply combining gene expression studies in order to increase n. This chapter also proposes some basic guidelines to conduct experiments that are known to reduce the influence of confounding factors. Chapter 4 explores the available gene selection algorithms for reducing p, which is an alternative approach to addressing the large p small n challenge, and it extends an available single study gene selection algorithm to a multi-study methodology. Chapter 5 explores different ways to test for the existence of batch effects and also evaluates the performance of batch effect removal methods. Chapter 6 introduces our novel batch correction method, BECCA, which utilizes canonical correlation analysis for performing batch correction. Chapter 7 offers a more empirical analysis of ideas and methods in the previous chapters. More specifically, Chapter 7 utilizes the findings of Chapters 4, 5, and 6 to conduct batch effect correction on gene expression data sets and compares the performance of BECCA to other available correction algorithms. We also

included a full mathematical description of CCA in Appendix A.

1.1 Contributions

In summary, this thesis mainly deals with different solutions for the "large p small n" challenge in gene expression microarrays. Our solutions are either based on reducing p, by utilizing effective gene selection algorithms, or based on increasing n, by combining several studies together and correcting for batch effects. The following list provides an overview of the main contributions of this thesis:

Contribution 1

We introduce a new batch correction algorithm based on canonical correlation analysis called BECCA. Assuming (1) additive batch effects confound each study and also (2) these effects are independent from the biological signal and (3) from each other, BECCA can separate the biological related signal from the signal caused by technical factors. BECCA removes the variation caused by technical factors by projecting each study onto the null space of the technical factors' main directions of variation (for more details see Chapter 6).

Contribution 2

Our empirical results demonstrate that, given the right parameter values, BECCA can remove batch effects as effectively as the current leading and widely used batch correction algorithm. This suggests the assumptions of BECCA are reasonable in dealing with real-world gene expression data sets (for more details see Chapter 7).

Contribution 3

Our analytic methods identify a set of biologically uninformative genes – using just this subset of genes, the mean correlation between the profiles of any two studies is around 0.95, regardless of their tissue of origin or disease state. Filtering these genes out will improve the performance of batch correction algorithms as well as the downstream analysis conducted on the data (for more details see Section 4.2.3).

Contribution 4

We extend the effective single-study variance-based gene selection method to a multi-study gene selection algorithm by proposing a computationally efficient

implementation that scales linearly with the number of datasets. Empirical results indicate this feature selection algorithm can reduce the destructive role of batch effects by identifying the genes that are most affected by technical differences (for more details see Section 4.2.2).

Contribution 5

We extended an existing method for evaluating the performance of batch effect correction algorithms that operates based on the number of differentially expressed genes, both within and across-batches. This evaluation method is able to detect the influence and severity of batch effects and does so in a very general way based on minimal assumptions (for more details see Chapter 5).

Chapter 2

Background and Related Work

The molecular processes underlying health and disease are extremely complex. Understanding how genes and proteins are responsible for complex diseases such as cancer in individual patients and at the population level is a huge challenge. According to available statistics, cancer is one of the leading causes of death worldwide [10]. Moreover, the rate of cancer is also increasing due to the influence of aging and population growth [10]. There are many different inter-related factors that make predicting, prognosis, and identifying the most appropriate treatment of cancer extremely difficult. Not only can cancer affect different organs in the body, but cancers in each organ are also very heterogeneous. For example breast cancer, the most common malignant tumor in women, has two main histological subtypes, and each one divides further into five subtypes based on its molecular properties[11–13]. This diversity might be the main reason for observing different outcomes after applying the same treatment to patients with apparently similar conditions. The same reason applies to the difficult task of prediction of metastasis[1] and recurrence[2] in cancer patients [14].

To deal with this complexity, scientists have started using systems biology and high-throughput molecular techniques to understand cellular cancer mechanisms at the level of DNA, RNA, and proteins [15]. As explained by the central dogma of molecular biology, many relevant properties of cells depends on a two step mechanism: First, *transcription* of genetic information encoded in DNA into messenger RNA (mRNA); and second, *translation* of mRNA into proteins, the major functional and structural elements in cells. In recent years, high throughput system biology

[1] Migration of cancer from one organ to another organ in the host's body.

[2] Return of cancer in the same position or the same organ, after being removed by surgery and treated by chemotherapy and/or radiotherapy.

methods have made it possible to characterize cells at the level of DNA, RNA, and proteins. Respectively this can be done by finding single-nucleotide polymorphisms (SNPs) and copy-number variations (CNVs) in DNA, measuring transcribed mRNA and microRNA (miRNA) levels, and the concentration of proteins and their post-translational states. These measurements provide a *snapshot* of what is happening inside cells in a comprehensive maner. Using these techniques has helped scientists to learn a lot of important information about mechanisms and processes for different types of cancer, which consequently has helped them to find new and more effective treatments.

As mentioned above, one of these techniques is transcriptomics, which measures gene expression levels (the amount of transcribed mRNA for each gene). Transcriptomics has been used extensively in cancer research [16]. Using this technology, researchers can measure the expression of several thousand genes simultaneously.

Each modern microarray chip is an array consisting of hundreds of thousands of hybridization probes, each a short fragment of DNA, usually between 20 and 60 bases long [17, 18]. These probes are designed to bind to a specific mRNA or DNA fragment that has the complementary sequence. The whole array of probes are designed to essentially cover all of the transcribed genome. Gene expression studies usually involve processing microarrays for a group of individuals. Different studies involve different groups of individuals: they might be specimens of cancerous tissues taken from cancer patients or tissues from mice that have been exposed to a specific drug, or cell lines cultured *in vitro* that have been exposed to a particular chemical.

In the context of cancer and microarray studies, different researchers may have different goals, with each such goal requiring its own specific algorithms and techniques. The goal of a study might be finding differences between patients that suffer from early *relapse* from those that do not. If physicians are able to predict accurately which patients will experience early relapse, then they can justify the use of more aggressive treatments to increase the chance of survival of only those high risk patients. Another goal is to predict which adjuvant treatment will be effective for a cancer patient. For each cancer type, several different chemicals can be used for chemotherapy, each of which is helpful for some, but not all, patients. Predicting the drug that works best for each patient is beneficial from several aspects. First, it improves the survival rate of patients, by both eliminating cancer using the most effective chemicals and not harming the general health of patients by exposing fewer

of them to more doses of dangerous chemicals. Second, it also reduces the imposed cost to the health system. Third, microarrays can be used to predict cancer subtypes, which consequently can help identify the appropriate treatments. Currently pathologists use histology to determine such subtypes, which can be costly, slow, and imperfect, due to human errors. We anticipate the techniques developed for targeting one of these goals in one kind of cancer are applicable to other types of cancer, especially among less common or less studied cancer types.

One major challenge for microarrays is their dimensionality: finding useful information from a relatively small number (n) of instances, each of which has a large number of features (p), *e.g.,* n is often less than 100 patients and p is often greater than 10,000. We will need to use novel techniques to deal with this "large p small n" challenge [8, 19, 20]. We anticipate these techniques will extend beyond microarray analysis, and apply to many related tasks, including many other biological datasets, such as SNPs, CNVs, proteomics, metabolomics, and others.

Section 2.1 we briefly summarize many previous results related to microarrays, focusing on the ways that other researchers have extracted meaningful data from them. We examine and compare attempts to deal with the "large p small n" challenge in Section 2.2, and analyze their abilities and limitations. Section 2.3 briefly describes potential solutions for this challenge and Section 2.3.3 analyzes the associated with those solutions.

2.1 Microarray Analysis

Microarrays have been used extensively in the context of cancer research. A simple search in the two largest public microarray repositories, *Gene Expression Omnibus* (GEO) [21] and *Array Express* [22], reveals that there are respectively 4151 and 4163 studies with the keyword "cancer" mentioned in their titles or description [3]. The availability of such an extensive set of microarray samples, all related to cancer, supports the claim that microarrays play an important role in cancer research. In this chapter we present a brief overview of many of the tasks and algorithms commonly used in microarray research. We categorize these tasks into two main subgroups, *association studies* (Section 2.1.1) and *prediction studies* (Section 2.1.2).

[3] Using the GEOmetadb package [23], we were able to extract the number of arrays in each 4151 cancer studies of GEO repositories. The average number of samples in cancer gene expression studies was 45.5471.

In order to measure the abundance of mRNA using microarrays, a lab technician first extracts mRNA from the intended cells and then exposes it to the microarray chip. Given enough time before washing the extracted mRNA off the array, each strand of mRNA will bind to the associated hybridization probe on the array. The higher the abundance of mRNA in the cell, the more of this mRNA will bind to the associated probe. A special scanner reads these intensities from the array and records them as an image; intensities on each probe shows the amount of mRNA bound to that probe. Specific image processing algorithms read intensities from these image files and save them into numerical files containing long list of numbers, one number for each probe. This format is called the *raw data*. In many platforms, a single gene has more than one probe measuring the transcription level of that gene [24], so here we need to aggregate the values of these probes in a smart way to find the best estimate for the gene expression values, respecting post-translational effects; see Section 2.3.3. This aggregation step is a part of the *preprocessing* stage. The output of this preprocessing step is again a list with p values in it, which is usually one order of magnitude smaller than the raw data. Companies that make microarrays have their own standard protocols for each step in this process. Figure 2.2 in Section 2.3 depicts these data types and algorithms that produce them.

This process is repeated for all n individuals of a study, which result in a $p \times n$ matrix of numbers. This chapter focuses on such $p \times n$ matrices of n instances (samples or patients) with p features (the number of probesets or genes) [25]. To simplify descriptions, we will assume that p is the number of genes.

Researchers have looked at this $p \times n$ matrix from two different perspectives. One perspective is *patient-based*, which analyzes each patient as a vector of p genes expression values. The second perspective, *gene-based*, looks at each gene as a n dimensional vector that represents one gene's expression levels across n patients (samples). The *gene-based* perspective finds relationships between genes by analyzing the relation between their vectors. For example, the normalized dot product of these vectors shows the similarity between expression patterns of two genes across different patients. As another example, the direction that these p individual vectors from $\mathbb{R}^n$ mostly spread out is the direction of the principle component of the data [26]. This perspective is useful for finding novel relations between genes; we, however, will focus on the *patient based* perspective.

The *patient-based* perspective allows us to find similarities and relations among

Table 2.1: Categorizing of microarray tasks based on the two mentioned perspectives.

	Patient-Based	**Gene-Based**
Supervised	▷ Phenotype Association ▷ Phenotype Prediction ▷ Survival Analysis	▷ Expanding Pathways ▷ Finding Novel gene-gene interactions
Unspervised	▷ Discovering Patient sub-populations	▷ Discovering consistent gene sets ▷ Learning Causal relationships

a population of patients by looking at their gene expression profiles. For example, by running an unsupervised learning algorithm (clustering) on expression profiles of patients, we can find sub-populations of patients with similar transcription levels. Later we can examine whether these sub-populations have similar clinical phenotypes. Supervised learning algorithms are also used here; by using the label assigned to each patient in the training set (*e.g.,* high risk versus low risk cancer patients) and applying a learner we can learn a relation between genes and the assigned labels. The output of such a learning procedure is a classifier that can predict the label for future patients based on their gene expression profile. Table 2.1 summarizes several standard tasks, categorized into 4 subgroups according to their perspective and learning method.

For supervised tasks that are"patient-based", the label is the patient's phenotype; when gene-based, the label could be a gene's membership in some pathway for "Expanding Pathways", or could be the bit for whether a pair of genes are known to interact, for the "Finding novel gene-gene interaction" task [27]. In this study we only look at **patient-based** tasks, specifically association studies and prediction studies are analyzed in Sections 2.1.1 and 2.1.2 respectively. Lee and Wang [28] present further information about survival analysis in this setting. One of the challenges faced by patient-based algorithms is "large p small n", since typically in a microarray dataset $n \ll p$, which is the opposite of the usual learning setting $n \gg p$ [19]. Section 2.2 explores this challenge in more detail.

2.1.1 Phenotype Association Studies

Table 2.1 notes that phenotype association studies are supervised patient-based learning tasks. The main goal of these studies is to find a gene set that is closely associated with the condition under study (*i.e.,* the phenotype). The resulting set of associated genes is sometimes called the *expression signature* of that phenotype.

For example, if the microarray study contains samples of two populations, *i.e.,* high risk versus low risk breast cancer patients, then the output of an association study will be a set of k genes, $k \ll p$, that are each highly correlated with this high risk versus low risk dichotomy, *i.e.,* each gene in this set is *differentially expressed (DE)* across these two conditions. One of the main issues with these association studies is the way the research community assesses such findings. Usually the researchers who conduct the association studies will then evaluate the expression signature against the current biological knowledge, *i.e.,* they compare their gene set with the set of gene that are already known to be related to the condition according to biological experiments. The fact that their new gene signature shares a non-trivial subset with these known set of genes is enough to suggest their newly obtained gene signature is "correct". This allows them to claim that the other genes in the signature are also probably associated with that condition, even if this is not yet supported by biological evidence. Notice this is a biological validation, and not a computer science one – *i.e.,* it is based on prior biological knowledge. It is also difficult to quantify: how good is it if 20 out of the 30 genes are known? How much better is knowing 25 out of 30?

There are two types of association studies, which differ based on whether they use prior knowledge of gene sets. One type of association study, *gene set analysis (GSA)* [29–31], receives an *a priori* defined gene set as input and decides whether this set of genes (individually or collectively) are expressed significantly differently between two conditions. The other division does not use any prior knowledge about a gene set; instead these algorithms assign a score to each gene that attempts to measure how "correlated" this gene is to the phenotype, and then rank genes according to their scores. We call them *gene ranking* methods; they are also known as *cutoff-based* methods since they apply a cutoff threshold to ranked genes in order to choose the top signature genes [29].

One can also categorize association studies based on the type of the statistic they use, which can be *associative* or *predictive* [31]. Associative statistics try to find differentially expressed genes by comparing the average expression of each gene over all instances in each of the phenotypes. Genes that have large differences between the averages of the two phenotypes are considered to be more "associated" with these phenotypes. On the other hand, predictive statistics look at the power of each gene for predicting the phenotype of a new instance. According to predictive

Table 2.2: Categorizing of phenotype association studies.

<table>
<tr><th></th><th>Gene Ranking
(Without Prior)</th><th colspan="2">Gene Set Analysis – GSA
(With Prior)</th></tr>
<tr><td rowspan="3">Associative
Statistics</td><td rowspan="3">correlation,
p-value, q-value [33],
signal to noise ratio,
fold change, SAM [34],
GSRF [35], GSFLD [35]</td><td>Sample
Randomization</td><td>SAFE [29], SAM-GS [36]</td></tr>
<tr><td>Gene
Randomization</td><td>PAGE [29], T-profiler [31]
GAGE [31], Random-Set [31]</td></tr>
<tr><td>Hybrid</td><td>GSEA [37]</td></tr>
<tr><td>Predictive
Statistics</td><td>PAM [38], k-TSP [39]</td><td colspan="2">GLAPA [30]</td></tr>
</table>

statistics, if a gene value is a good indicator for accurately predicting the phenotype of new instances then it is consider to be "associated" with the phenotype. In more technical terms, training a classifier using such a gene set will result in a small generalization error. Table 2.2 characterizes the tasks for association studies, based on these two criteria[4].

To assess the significance of a gene set under the null hypothesis that there is no difference in expression between two phenotypes, GSA methods usually use permutation tests [40]. The standard methods include *"sample randomization"*, *gene randomization*, or *hybrid*, which is a combination of both. The small table inside the *Association Statistics/With Prior* cell of Table 2.2 shows these three methods of *significance assessment*. Sample randomization approaches compare test statistics of the given gene set with the true label, versus the same gene sets statistics relative to a random permutation of labels assigned to instances [29, 31]. In contrast, gene randomization approach uses random permutation of the *genes* or a parametric distribution over genes. Luo et al. [31] explain that sample randomization keeps the correlation structure between genes in the gene set while gene randomization does not. However, in order to have more accurate estimation, sample randomization needs more instances of each phenotype.

Goeman and Bühlmann [41] call these two approaches *self-contained null hypothesis* versus *competitive null hypothesis*, as the former method requires only the genes in the given set to evaluate their significance while the latter methods "competes" these genes with other sets of genes to find their significance. Goeman and Bühlmann [41] analyzed these two approaches along with their assumptions pre-

[4]There are online tools like FatiScan [32] that let users utilize each of these methods through a simple graphical interface.

cisely and interpreted the resulting p-values of each one. They concluded that sample randomization approach is valid and its p-value is interpretable according to the traditional statistical definition of p-value. On the other hand, in the gene randomization approach, conclusions based on p-values are not trustworthy since the assumption of *i.i.d.* genes is not valid in the context of gene expression datasets.

Researchers have compared the effectiveness of different GSA methods in different circumstances by comparing their performances [30, 31]. Researchers also attempted to propose a general framework to cover all variates of GSA methods. Ackermann and Strimmer [29] proposes a modular framework that covers most of the GSA methods by trying all combinations of available algorithms for each module. They extensively compared 261 GSA variants on 9 syntactic datasets and 2 real microarray studies. Their results show that the average of simple uni-variate statistics, like t-statistics, combined with simple transformations, like ranking, and assessed using permutation tests performs reliably well across a wide range of situations. In another study, Adewale et al. [36] extended the works of [42–44] by introducing significance analysis of microarray to gene-sets *(SAM-GS)*, a unified measure based on the ratio of the regression coefficients to their standard error. They calculate this score for each gene in the set and then add up the squared scores to form the whole set's score. They apply this general settings to three types of phenotypes: uncensored independent, uncensored correlated binary phenotypes, and also censored survival analysis. These three types cover a wide range of applications and studies. Dinu et al. [45] extended SAM-GS to perform a significance analysis of microarrays for gene-set reduction *(SAM-GSR)* that not only declares the significance of a gene set but it also finds a subset of "core genes" in the set that are mainly responsible for the significance of the whole set. SAM-GSR gradually reduces a significant gene set and analyzes the association of remaining genes with the phenotype. The reduction stops at a certain threshold, which is chosen arbitrarily by the user without any claimed statistical property.

Considering the extensive research done in the context of associative studies and GSA algorithms, it seems the ultimate goal, finding a core gene set associated with a given phenotype, still is far beyond reach. A brief survey of the currently available expression signatures[5] associated with a certain phenotype, shows that there is no or

[5]For the list of breast cancer related signatures, go to http://rock.icr.ac.uk/browse/browse_

very limited agreement between studies conducted by different researchers. Ein-Dor et al. [46] studied this phenomenon toward determining whether *there is a unique set* or not. They looked at one of the first studies that used expression profiling for finding associated genes as well as predicting a phenotype, conducted by van't Veer et al. [47]. The original study used 77 breast cancer instances to find a prognostic signature in the expression profiles based on the top 70 associated genes, then used a disjoint set of 19 instances to evaluate the accuracy of this learned classifier. Ein-Dor et al. [46] repeated this method, but using several different random subsets of 77 patients. Surprisingly they found that top 70 genes selected using different subsets of patients is not unique and is strongly dependent to the subset of patients used for gene selection. However, using any subset of 70 genes selected from top 5000 genes correlated with survival resulted in the classifiers with approximately similar performances. Sims et al. [48] argues this is not surprising – findings are highly "context specific" and depend upon the patient sample used.

Ein-Dor et al. [49] later claimed that "thousands of samples are needed to generate a robust gene list". They claimed the main reasons of observing such an inconsistency is the fact that there are lots of genes correlated with the phenotype and the differences of these correlations are small. They hypothesized that one would get more consistent results by first dividing patients into smaller, biologically-similar subgroups and finding a separate gene signature for each subgroup [50]. Because each random sample contains different proportions of patients belong to these finer subgroups, genes correlations fluctuate strongly with the chosen random subset of patients. Ein-Dor et al. [46] also emphasized the distinction between learning a prognostic tool, *i.e.,* a classifer, versus finding a gene signature. One might learn a fairly reliable classifier using a large enough set of genes. However, the specific genes used for this purpose will not necessarily play an important role in the histology of the disease under study and so might not be the best potentials targets for treatment. In the next section we will talk about methods of learning such prognostic tools in more detail. We refer readers to von Heydebreck et al. [51] for a detailed guide of conducting association studies in R.

`gx_signature.jsp` gathered by ROCK, Breast Cancer Functional Genomics.

2.1.2 Phenotype Prediction Studies

As we mentioned earlier in Table 2.1, prediction studies are supervised patient-based learning methods, *i.e.,* they work with vectors of expression profiles of patients. Since many studies apply both *unsupervised* and *supervised* patient-based methods to the microarray data in order to get more insights, we present both of them in this section. In the rest of this section, we will use "prediction study" to refer to either of these two learning schemes, interchangeably.

Some prediction studies start their analysis with a *pre-filtering* step in which researchers apply a simple inter quantile range (IQR) filter [51] to the extensive space of features in order to remove genes that have almost constant expression values across all patients. This IQR step reduces the number of feature from p to p' where $p' \leq p$. Other researchers use statistical methods like principle component analysis (PCA) to reduce the number of features, but this is less common in this context as it is more difficult to interpret a feature set produced by PCA [52].

One type of prediction study is supervised learning, which learns a relation between expression profiles of patients and some clinically interesting label, using some *training data* in the form of a pair of expression profiles and labels: (X, y), where X is the $p \times n$ matrix of expression values and y is a a $1 \times n$ vector of labels. The output of a supervised learning algorithm is a *classifier*, which is an algorithm that predicts labels of future unlabeled patients based on their expression profiles. We evaluate these learned classifiers by how accurately they can predict labels of new instances. The best way to assess the accuracy of a classifier is to use a dataset that is disjoint from the training set, called a *validation set*. This approach is especially problematic in the case of microarrays, since we already have very few instances for training. Another common way for assessing accuracy is to use *cross validation* (CV) scheme [19]: here, we estimate the performance of the classifier obtained by applying a learner L to a dataset by dividing it into k disjoint subsets, known as *folds*. A classifier is trained using $k-1$ folds, whose accuracy is assessed using the last unused fold. This process repeats k times and the average of accuracy for all folds is reported as the estimate of overall accuracy. Conducting $k = 5$ or $k = 10$ folds cross validation is very usual; *leave one out* (LOO-CV) is a cross validation where the number of folds is equal to the number of instances ($k = n$).

Unsupervised patient-based methods, on the other hand, try to assign patients

to one of the *clusters* of patients, based on the similarity of their expression profiles and without looking at their labels (y vector). K-Means and hierarchical clustering are two most widely used algorithms for clustering [53]. *Multidimensional scaling (MDS)* is also used to find an intuitive visual representation of similarity among expression profiles of patients [54]. In this context, it is very common to run a feature selection algorithm before performing the unsupervised learning step to find some *"intrinsic"* genes [11, 47, 50, 55], *i.e.,* a subset of genes $p'' \leq p'$ chosen according to their correlation with some phenotypic property. Thus, depending on the set of features used by unsupervised method, they can find different types of clusters. Unsupervised methods that only use p'' intrinsic genes, versus those that use all pre-filtered p' genes, to find similarity among patients, will tend to find clusters that are more "biased" toward the phenotype that was used for finding the intrinsic gene set.

After finding clusters of patients, using either all genes or only intrinsic genes, researchers try to establish clinical relevance of their discovered clusters by looking at phenotypic properties of patients in each cluster. One of the common statistical ways to validate discovered clusters is to compare survival time of patients in each cluster; by first estimating survival distribution of that subset of patients using a Kaplan-Meier survival curve [28]; then comparing these survival distributions to determine if they are significantly different or not. For discrete phenotypic data such as pos/neg recurrence, poor/good outcome, pos/neg lymph node invasion, low/medium/high cancer grade, ..., many researchers use a similar statistical validation procedure (*i.e.,* using a χ^2 test) to compare the distribution of patients in these subsets and in the discovered clusters [55]. Metrics like *corrected rand index* or *adjusted mutual information* are also used for the same purpose, *i.e.,* to compare the clusters with the partitions defined by discrete phenotypes [54].

Many prediction studies apply a feature selection method; this leads to some subtleties here, especially when conducing cross validation. Dupuy and Simon [53] looked at many published prediction studies on microarrays and highlighted the common mistakes that many researchers make when assessing their classifiers. To avoid mistakes when conducting these two steps, feature selection and classifier training, Dupuy and Simon [53] explicitly showed the correct way of conducing CV on microarray datasets using a flowchart, similar to the one shown in Figure 2.1.

Choosing the best combination of feature selection and classification algorithm

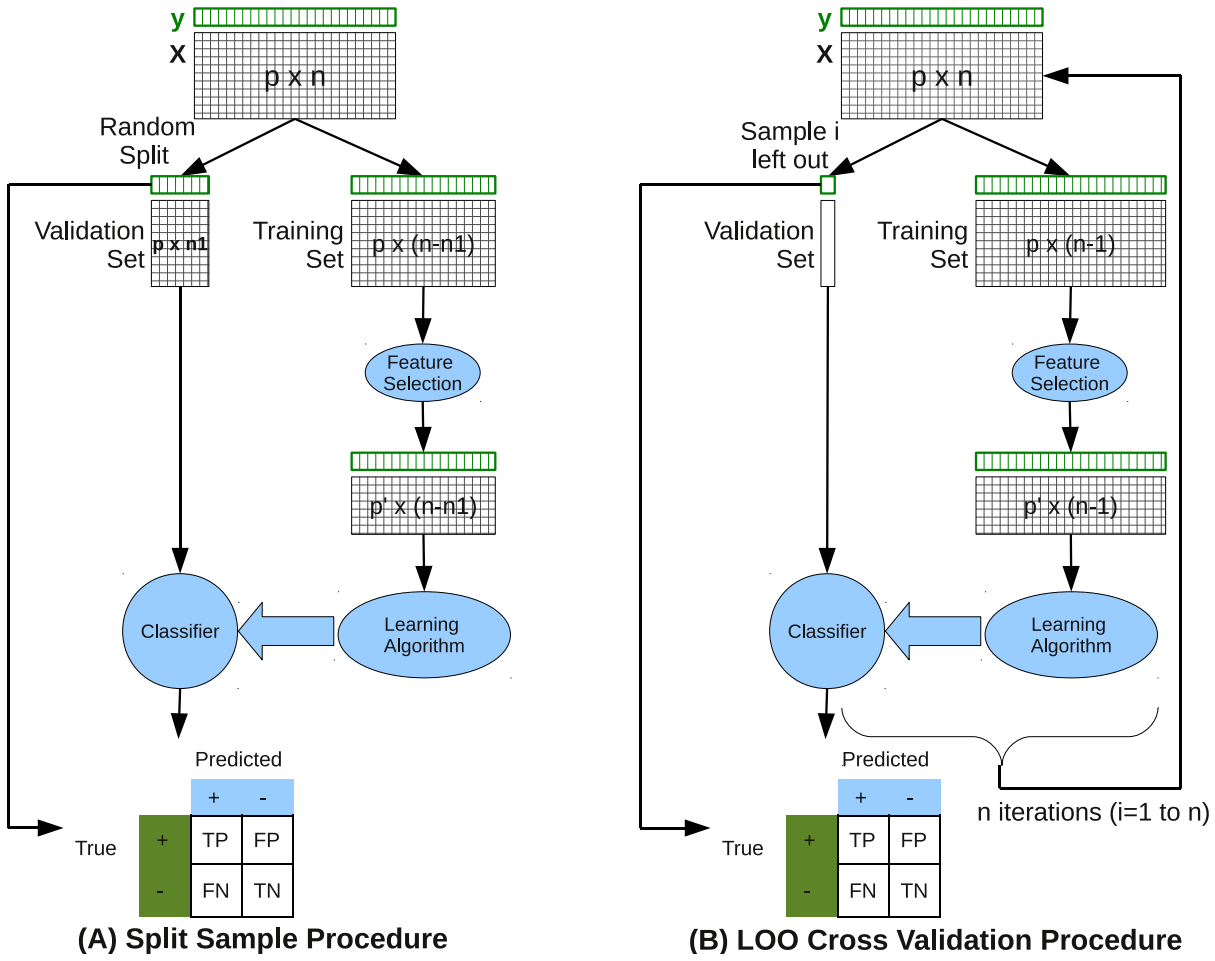


Figure 2.1: How to perform (A) Split sample (B) LOO-CV properly to get valid accuracy estimation, this is an estimate of the accuracy of the classifier produced by running a particular learning algorithm, on the ENTIRE n training instances. Adopted from [53].

is a hard task too. Popovici et al. [56] ran an extensive set of experiments to compare performance of 5 different feature selection methods combined with 8 different classification algorithms on 3 prediction tasks: an easy one, a medium one, and a hard one. They also looked at the effect of training sample size on the performance of classifier. They reported that the choice of a feature selection and classification algorithm only has a modest affect on the performance of the final learned predictor. However, they showed that the number of training samples mainly determines the quality of predictor. This can be one of the explanations why so many researchers recently are trying to reduce the huge gap between n and p by combining different studies together. We will analyze these methods in the next section.

2.2 Large p small n Challenge

The "large p small n" challenge is a well studied phenomena in the context of statistical learning methods. Hastie et al. [19] dedicates a whole chapter to "high-dimensional" problems and using a simple example demonstrate *"less fitting is better"* principal that applies to $p \gg n$ scenarios. They generate three datasets, each containing $n = 100$ samples, but with different number of features: $p = 20$, $p = 100$, and $p = 1000$, and then they fit a simple ridge regression to each dataset. They observe that ridge regression can easily exploit the correlation between features when $p < n$ but not for $p \gg n$, because in this case there is not enough information in the small number of samples, relative to the number of features, to effectively estimate the high-dimensional covariance matrix. They concluded *"analysis of high-dimensional data requires either modification of procedures designed for the $p < n$ scenario, or entirely new procedures"*. They then describe different techniques to deal with high-dimensional problems, including many that add *regularization terms* to the learning algorithms to reduce the number of used features in the final learned model.

Having a small sample set not only makes a lot of learning algorithms inapplicable in the domain of microarrays but also raises the question of whether the training sample is a good representation of the whole population. This is a very important concern, especially for supervised learning methods that are expected to classify all future instances correctly, regardless of the population that they belong to. For example consider a classifier for predicting breast cancer metastasis, trained by specimens of a hospital somewhere in North America, but then used by another

hospital somewhere else in the world, such as an Asian country. Clearly the training sample is not necessarily a good representative of the test population. In summary there are two concerns imposed by relatively small training samples: First, is a particular training set sufficiently large? Meaning, how will the learned model perform on an independent test set with very similar subjects. And second, is training set representative of the whole population? Meaning, how well the learned model will performed when tried on a more diverse set of subjects; probably different ages, ethnicities, and environmental factors?

Joint analysis of several available gene expression studies to increase n and consequently solve both of these challenges seems like a very natural solution as there are public repositories like GEO that have more than half of a million instances. While individual studies may not provide sufficient statistical power to infer conclusions about the relationship of genes and phenotype of interest, collectively analyzing gene expression data sets studying the same biological phenomenon may provide substantially more evidence to infer relationships [57]. We call techniques that combine gene expression studies, explicitly or implicitly, *joint analysis of gene expression studies*. Ideally, conducting joint analysis would involve these simple steps:

Algorithm 1 Naive algorithm for conducting joint analysis on K data sets that all contain samples of a particular phenotype of interest.

Require: $(X_1, y_1), \cdots, (X_k, y_k), \cdots, (X_K, y_K)$
where each X_k is a $[p_k \times n_k]$ matrix of gene expressions
Set of genes in each data set is denoted as G_k where $||G_k|| = p_k$
Vector of phenotype labels for n_k patients is denoted by y_k, which is a $[1 \times n_k]$
Find genes in common across all K studies
$G \leftarrow \{G_1 \cap G_2 \cap \cdots \cap G_K\}$ where $||G|| = p'$
for $k = 1, 2, \cdots, K$ **do**
Reduce each X_k to X'_k by making it to include only p' shared genes
$X'_k \leftarrow X_k[G, :]$
end for
Concatenate all reduced X'_k matrices to make a large $\left[p' \times \sum_{k=1}^{K} n_k\right]$ matrix
$X \leftarrow [X'_1, X'_2, \cdots, X'_K]$
Concatenate all y_k vectors to make a $\left[1 \times \sum_{k=1}^{K} n_k\right]$ vector of phenotype labels
$y \leftarrow [y_1, y_2, \cdots, y_K]$
return X and y

In reality, however, there are several problems with this approach caused by systematic differences among studies, such as: using different protocols in labs for specimen preparation including hybridization, washing, and staining; variability of

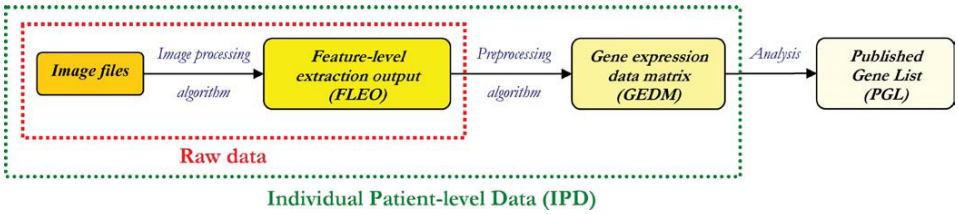


Figure 2.2: Levels of data abstraction in a gene expression study, adopted from [1].

probes, probesets, and genes among different platforms and difficulty of finding a way to map them together; varying scales and precisions used by different platforms to measure gene expression values; and availability of several ways for preprocessing and background correction of raw data. We call these systematic differences *batch effects* [2, 3, 58]. Chapter 3 provides more details about batch effects and how this complicates integrative analysis methods. For the rest of this chapter, we review the joint analysis techniques without considering the batch effect and we discuss how these methods tackle the challenging task of combining several studies. We should note, however, many of these techniques are designed particularly to avoid the batch effect problem.

2.3 Joint analysis of multiple gene expression studies

As the name suggests, joint analysis techniques combine (merge, or integrate) several gene expression studies together at different levels of data abstraction. Ramasamy et al. [1] showed four different types of data generated in gene expression studies, see Figure 2.2. In these studies, the data processing phase starts with scanning hybridized chips, producing an *image file*, which is later read by an image processing algorithm to produce a *feature-level extraction output (FLEO)* file. This file, known as *raw data*, contains raw intensity values for each probe on the chip. Preprocessing algorithms read these raw data files and apply some scaling and background correction to these values, accordingly make a *gene expression data matrix (GEDM)*, which is a vector with p values. This process is repeated for all n patients in the study and the final product will be a $p \times n$ matrix of expression values. This matrix will typically undergo further analysis to extract some meaningful biological knowledge from it, such as *published gene list*. Other than image files, the other 3 types of data are targets for conducting joint analysis.

Based on the type of data targeted for combining, joint analysis technologies are

divided into two main groups, *meta-analysis* and *integrative analysis* [59]. Meta-analysis methods analyze each gene expression study separately, and then combine the resulting statistics together to form more robust and general conclusions. Meta-analysis seeks a significant result that is observed in big enough number of individual studies and generalizes it to the particular condition that the studies were analyzing [2]. These results might be in the form of primary statistics, such as t-statistics or p-values, or secondary statistics, such as list of significant genes derived from individual studies [60]. As it is hard to derive rigorous statistical inference using small sample sizes, meta-analysis faces limitations when individual studies that are being pooled together are all small.

On the other hand, integrative analysis methods are cross-normalization algorithms that are applied to raw data or gene expression data matrix of individual studies, in the hope to make these data statistically comparable. After normalization, these matrices are combined together to produce a single unified larger dataset to which subsequent standard learning algorithms can be applied. The main advantage of the integrative analysis approach over the meta-analysis approach is its more robust results due to a larger sample size. There is no consensus among the researchers about *meta-analysis* and *integrative analysis* definitions. Campain and Yang [61] calls them *relative meta-analysis* and *absolute meta-analysis* respectively. Integrative analysis methods are also known as *normalization methods* [62] and *cross-platform classification* [63]. Figure 2.3 which shows the distinction between meta-analysis and integrative analysis in an intuitive way.

Both methods somehow follow the general naive Algorithm 1 of joint analysis. They both start with two, or more, gene expression studies, A and B. After preprocessing the raw data, each study has an expression matrix, X_1 and X_2 of size $p_1 \times n_1$ and $p_2 \times n_2$ respectively. Then we need to find common set of genes between them and reduce expression matrices to shard genes only; they will be $p' \times n_1$ and $p' \times n_2$ respectively. The main difference between meta-analysis methods and integrative methods happens in this stage. Meta-analysis methods extract a $p' \times 1$ vector of summary statistics from each study individually and then combine these vectors, while in integrative analysis we *normalize* expression matrices to make two new matrices with the same size, here called X_1' and X_2', and combine them together to make a bigger $p' \times (n_1 + n_2)$ gene expression matrix.

Several statistical methodologies have been used for conducting joint analysis of

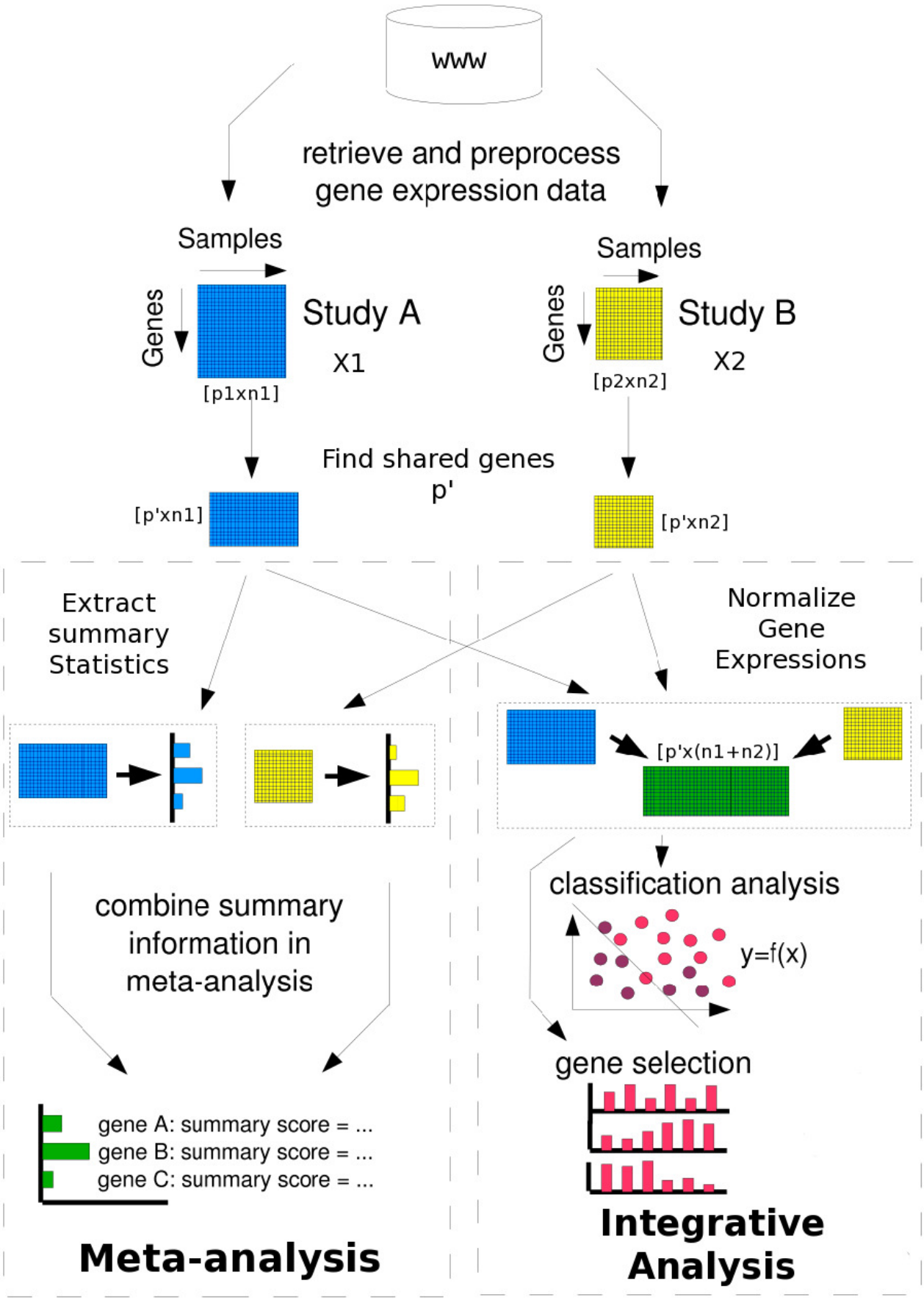


Figure 2.3: Meta-analysis methods, left panel, versus integrative analysis methods, right panel, adapted from [63].

Table 2.3: Categorizing of joint analysis of multiple gene expression studies.

<table>
<tr><td colspan="2" rowspan="2">Meta-analysis</td><td>Vote counting
Meta-signature [1]</td><td>Combining ranks
RankProd [1], METARADISC [1]</td></tr>
<tr><td>Combining p-values
Fisher's inverse χ^2 [1]</td><td>Combining effect sizes
GeneMeta [64], mDEDS [61]</td></tr>
<tr><td rowspan="2">BE correction

(Integrative Analysis)</td><td>Discrete</td><td colspan="2">k-TSP [39], MRS [63], QD [63], POE [65], NORDI [66]</td></tr>
<tr><td>Continuous</td><td colspan="2">Pairwise DWD [67], LTR [68], RUV-2 [9], XPN [60]
Multiple ComBat [69], SVA [70], BFRM [71], Weibul based normalization [54], housekeeping gene based normalization [54], FA [72], AGC [54]</td></tr>
</table>

gene expression studies; Table 2.3 categorizes them based on different criteria. In the following sections we review these categories in more detail. In Section 2.3.1 we present available meta-analysis techniques and compare them together and then in Section 2.3.2 we analyze different integrative analysis methods. Finally in Section 2.3.3 we will describe the challenges of conducting a joint analysis and also the measures used for assessing effectiveness of these methods.

2.3.1 Meta Analysis Techniques

Ramasamy et al. [1] provides a comprehensive review of meta-analysis methods and categorizes them into four groups. Note that all of these methods use primary or secondary results of gene expression studies and combine them together.

- **Vote counting.** This technique counts how many times a gene has been declared differentially expressed across studies. It might use statistical methods, such as permutation tests, to find the null distribution of votes. Rhodes et al. [73] proposed meta-signature method based on this technique and they identified common transcriptional profiles of neoplastic cancers using it.

- **Combining ranks.** Unlike the vote counting, this technique accounts for the rank of differentially expressed genes across studies. Different statistical algorithms such as Markov chains, order-statistics, and Monte Carlo permutation test are used for estimating the null distribution. METRADISC [1] and RankProd [1] from Table 2.3 each fall into this category.

- **Combining p-values.** As name suggests, this technique combines p-values assigned to individual genes across studies to make a more robust score.

Fisher's inverse chi square falls into this category [74].

- **Combining effect sizes.** Techniques in this category calculate effect size for each gene, which is the difference of mean expression value of gene in two groups of patients standardized by pooled standard deviation of the gene. Choi et al. [64] used this technique and introduced GeneMeta method.

There are more advanced methods that use several DE measures to insure that selected genes are robustly significant both across different measures and different studies, such as meta differential expression via distance synthesis (mDEDS) [61].

After introducing these 4 categories, Ramasamy et al. [1] gives a simple guideline for choosing the most suitable technique for different situations. Their guidelines suggest incorporating the level of data, shown in Figure 2.2, and set of genes available for each study. The other issue is concern with frequently-studied and rarely-studied genes, especially when combining an old platform, which measures expression of a limited set of genes, with a newer one. Clearly, in this case, vote counting is not a suitable choice since all genes do not have the same chance to be DE in different studies. Since some techniques need more studies to calculate the significance of a DE gene, the other issue arises when one combines a small number of studies. Another factor for choosing a meta-analysis is computational complexity; faster methods are desired, especially when we combine several dozens of datasets and computation time becomes a limiting factor. They concluded that techniques based on the *combining effect size* are the most effective meta-analysis for two-class gene expression studies. They mentioned a few reasons for this conclusion, including the facility for combining data from one- versus two-channel microarrays.

2.3.2 Integrative Analysis Techniques

Table 2.3 shows that one major type of joint analysis is *integrative analysis*, in which several gene expression studies are combined at expression value levels after transforming them to be numerically comparable. In general, integrative analysis involves three steps. First, one needs to find the common genes or probesets between studies by cross-referencing genes to a common annotation, such as *Entrez Gene ID* [75]. This is not a trivial task because of discrepancy of probes in different platforms; see Section 2.3.3. After retaining only p' shared genes in each study, the second step is to apply some data transformation and data normalization technique

to the expression values of each study to make them numerically comparable with each other. The main difference between integrative analysis methods mentioned in Table 2.3 is the statistical methodology used for *normalizing* data. The last step is to concatenate these normalized gene expression matrices together to make a new gene expression matrix that contains instances of all integrated studies. After the integration analysis is finished, we can apply any of the association and prediction studies mentioned in Sections 2.1.1 and 2.1.2 to this new larger matrix of gene expression.

If we ignore the biological properties of gene expression values and look at them as a $p' \times n$ matrix (*i.e.,* n instances in a p' dimensional space), then we can investigate normalization methods under a broader range of algorithms known as *transfer learning* [76]. Table 2.3 categorizes Integrative analysis methods at the highest level into two groups: *discrete* and *continuous*. Most of the discrete methods are based on *equal frequency binning* [63], *i.e.,* all values in each gene expression profile are sorted into b bins, $3 \leq b \leq p'$, over all profiles in all studies. These algorithms then assign the same value to all genes that fall into the same bin across all studies. These values might be cardinal numbers or continuous numbers derived from mean gene expression values. All ranking and quantile normalization methods, such as median rank score (MRS) [63], quantile discretization (QD) [63], and k top-scoring pairs (k-TSP) [39, 77], follow this simple methodology. The other type of discrete methods divide genes into three groups *under-expressed*, *over-expressed*, and *unexpressed* and then replace each gene expression value by -1, $+1$, and 0 according to its assigned group. Probability of expression (POE) [65] and normal discretization (NORDI) [66] each implements this idea. They each fit a normal distribution to genes expression values and then divide them into three groups using some statistical measures, such as a z-score. POE fits a normal distribution to each gene across all instances in a study, while NORDI fits a normal distribution to each expression profile, after eliminating outlier genes. There is no consensus among researchers on which of these two ways is the better way to assign genes to under expressed and over expressed groups. Warnat et al. [63] compared two discrete methods, MRS and QD, and found that, in most of the cases, either of them improve the results significantly compared to using no normalization. They also found that, except in one of the experiments, where QD performed significantly better than MRS, they performed equally.

The second big subset of integrative studies, in Table 2.3, are called *continuous* methods. We further divide continuous methods into two subsets, *pairwise* and *multiple*. As their names suggest, pairwise methods normalize studies in pairs. In order to integrate several studies using a pairwise method there are two options: one is to choose one study as *reference* and normalize all other *non-reference* studies based on it. The other option proposed by Taminau et al. [78], is simply pairwise merging of data sets and the intermediate results recursively until only one merged dataset remains. As listed in the Table 2.3, *multiple methods* are not limited to pairs of data sets and they can normalize multiple studies simultaneously.

All of the continuous integrative analysis methods work based on "shifting" the gene expression measurements of all studies "toward" one another. For example, the *distance weighted discrimination (DWD)* method [67] finds a hyperplane in p' dimensional space that separates data points of the first study from the second one and then shifts both of these "clouds" of points toward this hyperplane. The Combat method [69] uses empirical Bayes (EB) techniques to remove *batch effects* from gene expression values. They assume that the expression of each gene in a study is affected by a known designed batch factor. They estimate these batch effects, using one additive and one multiplicative term, for each gene in each study and then remove them from the expression values of genes. Later, Bayesian factor regression modeling (BFRM) [71] and surrogate variable analysis (SVA) [70] in two independent, but very similar, ways extend the ComBat framework by incorporating the effect of *unmodeled latent factors* on the expression values of each gene, in addition to the effects of known designed factors. They both use an additive model to find the effect of the latent factors on the residuals of expression values after counting for the effect of known factors.

Another method called remove unwanted variation 2-step (RUV-2) [9] utilizes the *negative control genes*, genes that are known or believed to be unrelated to the biological signal of study and thus their intensity variation across patients is due to *unwanted variation*, *i.e.,* technical factors, and noise. They also assume that the unwanted variation that affects the negative control genes, similarly affects all other genes. To correct the batch effect, RUV first finds the top few directions of unwanted variations of control genes, second it removes the variation of all genes in these directions. The first step is performed simply by applying a SVD analysis to the expression of negative control genes and finding the top k directions. The

second step projects all data into the orthogonal complement of subspace spanned by the k unwanted directions.

Many researchers (including Autio et al. [62]) use a Weibull distribution to normalize gene expression values, especially studies that are made using Affymetrix platforms that result in logarithmic probeset values. Housekeeping genes[6] have also been used for normalizing studies [62]. M2DB [79] and ArrayMining [80], which are two web applications that help users to integrate available gene expression studies, offer some of the techniques mentioned in Table 2.3.

Note that some of these methods, however, might not be applicable in all integration scenarios because of their assumptions. For example, Boutros [68] used *linear transformation of replicates (LTR)*, which assumes that there are some replicated instances shared between studies. LTR learns a mapping function between studies using these replicated instances and then uses this function to integrate studies. Unfortunately, only some pairs of studies include replicated instances. Another example is *cross-platform normalization (XPN)* [60], which is based on a block-linear model, *i.e.,* it assumes that each study consist of $K \times L$ *bi-clusters*, which involves K groups of genes defined over all studies and L groups of statistically homogeneous samples that are roughly similar between studies. Obviously finding equivalent bi-clusters in all studies that undergo the integration analysis is not guaranteed. Furthermore, most of the time integration analyses are done to extend the diversity of training samples by combining studies that contain very different instances. Another extreme example of methods with limited applicability is *factor analysis (FA)* [72], which "unifies" gene expression values of a same specimen that has been measured using several platforms.

The other characteristic of continuous integrative analysis methods is their scope, *i.e.,* what kind of batch effect (Chapter 3) they are able to remove. While most of the methods we mentioned here, such as EB, XPN, and Weibull based normalization, are suitable for removing cross batch and cross lab effects, despite their misleading names, they are not very suitable for removing cross platform effects, which are usually much more severe than other types of batch effect. Kilpinen et al. [54] proposed the *array generation gene centering (AGC)* for removing cross platform effects. Their method needs a large number of samples for its statistical modeling;

[6]Housekeeping genes are a specific set of genes that are assumed to be expressed identically across all samples [62].

e.g., they used 9783 samples to integrate 5 platforms together. AGC assumes that the mean of the expression values of each gene is equal across all platforms. If this is not the case for one gene in one platform, that gene's expression values in that platform will be shifted toward common mean of the same gene in other platforms. This AGC method needs to have relatively large number of samples from each platform to normalize gene expressions values based on its assumptions. Autio et al. [62] compared the performance of AGC combined with 4 cross-lab normalization methods in a big prediction study containing 1461 instances of 35 anatomical classes of healthy tissues. They showed that the performance of all 4 normalization methods improved when they were used after removing the cross platform effects by AGC.

So far we have looked at all joint analysis methods comprehensively and we saw techniques that have been used for combining gene expression studies. In the next section, we will briefly look at the issue and challenges faced by these methods.

2.3.3 Issues and Challenges

There are lots of issues that need to addressed when conducting a proper joint microarray analysis. Ramasamy et al. [1] list seven issues regarding meta-analysis and challenges faced by researchers who are conducting a meta-analysis. Some of these issues and challenges are not specific to meta-analysis, *i.e.,* they are also encountered by integrative analysis too. One of these challenges is related to the annotation of individual platforms and finding a way to map probes across platforms. Microarrays do not use the full sequence of a gene for their probes; instead they use a *short* highly specific region of a gene for a probe. Shortness of probes raises several difficulties, such as binding of some probes to more than one gene's transcript and overlap of some of the probes with an intron, an alternative splicing region, or other location, due to a single nucleotide polymorphisms (SNP). Moreover, platforms use different regions of a gene for their own probes, which makes finding a mapping between different platforms hard. The relationship between genes and probes is "many-to-many", *i.e.,* there might be a probe that matches to several genes and there might be several probes that match to a single gene. For probes that match to several genes, the solution is clear; they are defective and should be removed from the gene expression matrix. However, for matching of several probes to a same gene there is no clear solution. Choosing one of them at random will clearly lose

information, by not only eliminating some potentially informative features but also by losing the post-translational information such as alternative splicing. Instead of choosing one probe at random, some researchers suggested to pick one with the highest variance over the instances. Averaging is another solution, but will smooth out some information, just like the alternative splicing effect. This is a very hard issue to resolve for gene summarized data.

To find defective probes and remove them so as to decrease the level of noise and misleading signals, researchers have been looking at probe quality, especially for widely used platforms. Nurtdinov et al. [81] divides all Affymetrix probes into 4 classes, identified with four colors, green, yellow, black, and red. The best group, green, meets three conditions: (1) the probe matches with the target gene, (2) it does not match to any other gene, and (3) it does not match to any other non-coding region. Yellow group violates only the third condition, red group violates the second condition, and black, which contains the worst type of probes, violates the first condition. Following this protocol, they showed that the latest Affymetrix platform for human, `HGU_133_Plus2` has less than 60% green probes. Dai et al. [82] also look at the same problem by comparing probe sequences with the latest knowledge of the human genome for a wider range of microarray platforms. They release their updated probe definition via *custom chip description files* (CDF), known as *BrainArray*, which can be used by preprocessing algorithms to filter out defective probes and so produce a higher quality gene expression matrix. In their latest release, version14, only 51% of the probes in `HGU_133_Plus2` platform is retained. Sandberg and Larsson [83] reported that using these custom cdf files, instead of original probes definition provided by manufactures, improved both precision and accuracy of expression intensity measurements(see Chapter 3).

2.4 Summary

This chapter has briefly covered the wide range of meta-analysis/data integration algorithms that are being applied to gene expression studies. These algorithms mainly serve two purposes, finding differentially expressed genes between two or more phenotypic groups and distinguishing between different phenotypic group's instances. All algorithms utilized for this purpose suffer from the "large p small n problem" that naturally exist in gene expression studies.

There are two solutions for reducing the large gap between p and n: one is to

reduce the number of features and the other is to increase the sample size. The feature selection algorithms are explored in Chapter 4 and the current chapter reviewed integrative analysis, which combines multiple gene expression data sets in order to increase n. Combining gene expression data sets is not a trivial task as their expression values are influenced by the batch effect. One way to deal with this problem is to extract some statistics from the gene expression studies and combine these statistics instead of combining the intensity values directly. In these chapter we reviewed some of the methods proposed and utilized for this purpose in the literature. In the next chapter we will be looking at the batch effect and methods that are proposed to correct this confounding factor.

Chapter 3

Batch Effects in Microarray data

Conducting large-scale genomic data analysis is highly desirable for two main reasons. One is to increase the statistical power of the tests and produce more robust the findings by studying larger sample sizes. The other benefit of conducting large scale analysis is to cover the biological diversity of the phenomenon under study and be able to infer relevant discoveries that are general enough to be applied to individuals regardless of their environmental factors. A clear example that needs a large collection of biologically diverse samples is the clinical outcome prediction where thousands of samples are needed to generate robust gene/protein signatures [49, 84].

Because of the costs involved with gene expression analysis and also the time and geographical/logistic limitations, it is not possible for one research center to conduct huge genomic studies in isolation. However, the abundance of public gene expression studies makes the integrative analysis approach a natural solution for large-scale gene expression analysis. Indeed the capability of integrative analysis of multiple gene expression studies has been acknowledged by several studies [2, 85]

As we saw in Chapter 2, conducting integrative analysis is not a trivial task due to the confounding role of technical factors on the measured expression values, known as batch effect. The influences of batch effect on the expression intensities makes these values not directly comparable across data sets. This means if one integrates data sets with similar biological properties, without correcting for differences due to their technical factors, then the batch effect will mask the common biological signal and learning algorithms will fail to effectively identify it. In this chapter we will be looking at batch effect and learn about algorithms that have been proposed to correct it.

Section 3.1 first explains what can be considered as a "batch" in gene expression data sets and Section 3.2 examines possible sources of technical difference. Section 3.3 reviews the basic principles of experimental design and proposes general guidelines for conducting gene expression studies to reduce the chances of technical factors confounding the signal of interest. The last section of this chapter introduces the main ideas for detecting the batch effects and measuring their relative severity.

3.1 What is a batch?

There are many proposed definitions for batch effects. Lazar et al. [2] identified five definitions and highlighted the two consistent ideas in them: (1) emphasizing the distinction between batch effects and biological signals; (2) mentioning different potential sources of batch effects. Based on these observations, Lazar et al. [2] proposed this definition:

> *The batch effect represents the systematic technical differences when instances are processed and measured in different batches. Batch effects are unrelated to any biological variation recorded during the microarray gene expression experiment.*

They added that the term "batch" refers to a collection of instances that are processed at the same site over a short period of time using the same platform and under approximately identical conditions [2]. We extended this definition by having less restriction on what we call a "batch".

In this thesis, the word "batch" refers to a set of instances (samples) that are being grouped together and share some technical details. We study only gene expression microarrays for our experiments [1], however, these measurements might be any other high-throughput technique such as microRNA, proteomics, metabolomics, RNAseq, or perhaps any other data matrix where technical factors have a significant effect on measured values. In the case of gene expression microarrays, the potential technical commonality might occur at different scales [86–88]

- the same RNA extraction agent
- the same technician who handled the arrays
- the same microarray platform and/or chip generation

[1] Instances of gene expression microarrays are know as gene expression profiles, gene expression arrays, or simply just profiles or arrays.

- the same scanner
- being profiled in the same day
- being profiled in the same lab
- being preprocessed in the same study

Different batches may have different settings for each of these technical differences, so each one can be a potential source of a batch effect. According to these items, a batch has a very flexible definition. To better understand the meaning of a batch, we use Figure 3.1, which schematically shows the gene expression matrices of two studies (with sizes of $[p \times 12]$ and $[p \times 8]$ were p is the number of genes) combined together, resulting in a $[p \times 20]$ data matrix. The first study's arrays (one per instance) $A_1, A_2, \cdots, A_{12}$ and the second study's arrays are $A_{13}, A_{14}, \cdots, A_{20}$. Note that these two studies each measure the expression values of the same set of p genes, shown as $G_1, G_2, \cdots, G_p$, so we can simply merge the two matrices together. The element in i^{th} row and j^{th} column of this matrix contains the expression value of gene G_i in array A_j. However, since this is just a schematic example we left all elements of the matrix empty. The top 4 rows contain some information about each gene expression array.

The entire matrix of combined gene expression studies in Figure 3.1 is grouped along its columns into some sub-matrices. In the highest level of this nested grouping, arrays are grouped (based on their study of origin) into two groups, `Study1` and `Study2`, as shown in the first row. The study itself is often the clearest "batch" because it is normally the largest source of variation among all technical factors. Instances belonging to the same study are generally processed using the same standard protocol, equipment, and algorithm for each step of RNA extraction, labeling, hybridization, scanning, and preprocessing. This typically makes the instances of each study very similar to one another and often different from instances of another study, even if both studies are exploring the same biological phenomenon.

Within each study, there might be additional sources of technical differences, such as different hybridization dates, different technicians conducting the profiling, different scanners used for reading the slides, or any other recorded non-biological factor that potentially might have an effect on the gene expression intensities. These factors group arrays, within each study, into different *batches*. We represent these

1	Study1												Study2							
2	Bt_{11}		Bt_{12}		Bt_{13}	Bt_{11}	Bt_{12}	Bt_{13}	Bt_{11}	Bt_{12}	Bt_{13}		Bt_{21}				Bt_{22}			
3	PhnTyp1					PhnTyp2			PhnTyp3				PhnTyp1		PhnTyp2		PhnTyp3			
4	SubTyp1			SubTyp2		ST1	ST2	ST1	SubTyp1			ST2	SubType2			-	-	-	-	-
	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20
G_1																				
G_2																				
G_3																				
G_4																				
G_5																				
G_6																				
G_7																				
$\cdots$																				
G_p																				

Figure 3.1: A schematic example of merging two gene expression studies which results in a $[p \times 20]$ data matrix. Genes are annotated as $G_1, G_2, \cdots, G_p$ and arrays are annotated as $A_1, A_2, \cdots, A_{20}$.
The top 4 rows contain some annotation for each array: technical factors are in Row1-2 and biological factors are in Row3-4.

groups symbolically in the second row by $\texttt{Bt}_{1i}$ $i = 1, 2, 3$ and $\texttt{Bt}_{2j}$ $j = 1, 2$, respectively, for `Study1` and `Study2`. Studies might have more than one technical factor; if so, they would need more than one row to represent their batches. This also means that two studies might have different number of rows to represent their technical factors.

After considering the batch related factors, we have the biological covariates (row3-4). Again a nested relation exists among these factors. For example, at the highest level we usually have the treatment factor (case versus control) and then under these two groups we might have other biological properties such as different cancer subtypes and grades, age, ethnicity, clinical outcome, etc. There is no need for all studies to have the exact same biological annotations, *e.g.,* one study might include the age of patients while another study might not record the age. As an example in Figure 3.1, `Study1` has the full `subtype` annotation in row4 but `Study2` does not.

One important difference between groups defined by batches versus groups defined by biological factors is their correspondence across studies. As batches defined inside one study are irrelevant to batches in other studies, we represent them with

different names in Figure 3.1 ($\texttt{Bt}_{1i}$ versus $\texttt{Bt}_{2j}$). Note that these names are completely arbitrary and they do not correspond to any biological concept. On the other hand, when two studies share some biological annotation in a row, then the annotations are consistent across studies; we show this by using the same biological names across the two studies in Figure 3.1 in row3-4. For example `PhnTyp1` has the same meaning across the two studies as it refers to the same biological phenomenon in both `Study1` and `Study2`. Names used for biological annotation correspond to some biological phenomena that can be repeated and restudied by other researchers. This important contrast between technical factors and biological factors, is the main reason why they are modeled respectively as *random effects* and *fixed effects* in linear mixed models [89].

Note that a particular array might belong to more than one batch, for example A_1 belongs to both `Study1`, which is inherently a batch grouping, and $\texttt{Bt}_{11}$. Also two arrays might share a few batches while they belong to some other batches exclusively. For example A_1 and A_3 both belong to the `Study1` batch while their membership in $\texttt{Bt}_{11}$ and $\texttt{Bt}_{12}$ batches is exclusive from one another.

Because of this nested property of BEs, we might need to correct for them at different levels, depending on what batches are merged together. For example, if we are only using arrays in `Study1` then we only need to correct for differences between $\texttt{Bt}_{1i}$ $i = 1, 2, 3$. However, if we want to merge arrays in `Study1` and `Study2` then we need to correct for three BEs: two internal BEs within each study ($\texttt{Bt}_{1i}$ $i = 1, 2, 3$ and $\texttt{Bt}_{1j}$ $i = 1, 2$), and BE across the studies (`Study1` versus `Study2`). In this chapter we will be mostly looking at studies as batches, for example, two breast cancer studies that were conducted by two research groups in two different labs completely independent from each other. Methods and ideas in this thesis, however, are applicable to batch effects at any other scale.

The arrangement of the technical factors and biological factors with respect to each other and the assignment of arrays to them determines how correctable the BE is. More specifically, we say that (some of) the biological factors are *confounded* with BE when their effects cannot be distinguished from one another. The batches of `Study2` in Figure 3.1 are an example of this phenomenon; one can see that phenotypes are confounded with batches. For example, assume one gene's expression values is significantly difference between `Phntype2` (arrays A_{15}, A_{16}) and `Phntype3` (arrays $A_{17}, \cdots, A_{20}$). In this case we can not determine whether its significant

difference is due to the phenotype effect or due to the batch effect.

In contrast, `Study1` does not suffer from this phenomenon, *i.e.,* batchs and phenotypes are arranged in a way that makes it possible to distinguish the significant differences in expression values due to phenotype versus those due to the batch effect. Many researchers emphasized that applying batch correction algorithms in cases like this, where the unwanted factors are correlated with the factor of interest, will "harm" data rather than "help" it [9]. In Section 3.3 we briefly look at the principles of experimental design and propose simple guidelines to avoid situations like `Study2`.

3.2 Identifying a significant batch effect

To confirm the confounding role of batch effects on gene expression data, scientists have conducted experiments under controlled perturbation of technical factors. These experiments usually involve repeated gene expression profiling of a set of specimens under a set of controlled technical conditions. There are also studies that were conducted to evaluate the *reliability* of gene expression measurement techniques and/or to *validate* the conclusions of a study by re-running it using a new platform [57]. Even though these studies usually are not designed to analyze the batch effect as a main goal, they often re-discover the batch effect when they attempt to *combine* these parallel studies in order to increase the statistical power [57, 90]. These controled expriments usually study the effects of three types of technical perturbations:

1. **Cross platform effect:** comparing the expression intensities of the same set of specimens profiled using different platforms.

2. **Cross lab effect:** comparing the expression intensities of the same set of specimens profiled using same platform but in different labs.

3. **Cross batch effect:** comparing the expression intensities of the same set of specimens profiled using the same platform and in the same lab but possibly by different lab technicians and/or at different times.

McCall and Irizarry [91] analyzed the *cross platform effect* by comparing the precision of the measurements on three platforms, Affymetrix, Agilent, and Illumina, using spike-in experiments where the outcome of measurements is known *a priori*.

They conclude Agilent and Illumina have better overall accuracy while Affymetrix has better control of the precision, where they define *accuracy* as the ability to detect $2\times$ observed intensity when the nominal concentration of a gene is doubled, *precision* as the variability of log-ratios, which are expected to be zero, generated by comparison of genes that have the same nominal concentrations.

Weis [92] provided two specimens, rat liver specimens and a pooled specimen of five organs of rats including their livers, to seven laboratories and asked them to measure gene expressions of these two specimen using all platforms available in the lab; this involved 12 microarray platforms overall. By comparing the measurements from the labs they evaluated the facility of comparing results across labs and across platforms, and so analyzed both *cross platform* and *cross lab effects*. They concluded that the highest similarity between measurements is achieved by using commercial platforms and following all standard protocols proposed by manufactures for performing each step, from hybridization to preprocessing of raw data. They also measured the relative contribution of each factor – platform, lab, replicate, and tissue – in the variability of gene expression measurements using an ANOVA random effect model. Their analysis revealed that more than half of the observed variability is attributable to the platform; lab and replicate factors contributions are less.

Yang et al. [93] ran a similar experiment in a more controlled way, *i.e.,* instead of hybridizing on different platforms, they used the same platform, Affymetrix GeneChip, in all labs. They prepared specimens from two male and two female mice belonging to each of the 5 different stains and sent these $(2+2)\times 5 = 20$ specimens to 5 labs to measure gene expression following the existing standard protocols for all steps of hybridization. The raw data returned from each lab was prepocessed using the standard Affymetrix algorithm, MAS5 [94]; this way they avoided all diversities of data processing and acquisition. Then they compared gene expression values across labs according to several metrics, including median and median absolute deviation (MAD) of intensities, set of differently expressed (DE) genes for both gender and stain phenotypes, number of shared high ranked significant genes, and direction of principle components of genes for each lab.

Even though these expression profiles were measured under controlled situations and using the same platform, Yang et al. [93] still observed large discrepancies in expression profiles of same samples across labs. In particular, they found that the number of DE genes associated with the phenotypes were very different across the

labs; for example the number of DE genes at a significance level of $\alpha = 0.001$ for stain phenotype was 377 for one of the labs but 1390 for another lab. They found that only 17% to 50% of DE genes were shared between two labs. To explain this considerable discrepancy between DE genes, they hypothesized that the calibration of the statistical tests, p-value and q-value [33], might be a contributing factor.

To investigate this hypothesis, they ranked p-values of genes in each study and then found the intersection between these ranked lists. For stain phenotype they found that 60% to 80% of the 70 top ranked genes were declared significant in at least two labs. Using a hierarchical clustering method to find similarities between expression profiles, they found that the samples of each lab formed a cluster together regardless of the strain or sex of the mice. However, after applying a normalization method [95], the clusters were formed according to the sex and strain of mice rather than the originating lab. They concluded that, not only does the *cross lab effect* make systematic differences between samples, but also there is a considerable *batch effect* within the same lab. To reduce this type of effect they proposed that labs should avoid hybridization of samples in batches with similar phenotype and instead make the process more *randomized*. We will look at this matter, and more generally the correct experimental design principals, in the next section.

Note that batch effects might have many different sources other than the three major ones we highlighted here. Scherer [58] studied all potential sources of batch effect for the gene expression micorarrays and Lazar et al. [2] summarized them in one informative plot, which is reproduced in Figure 3.2. As one can see, potential confounding factors are present in all five main stages of gene expression profiling. Following careful experimental design principles can avoid many of these sources of technical differences.

3.3 Batch effects and experimental design

The ability to detect and correct batch effects is highly dependent on the way subjects were assigned to batches [58], *i.e.,* the *experimental design process*. Consider a scenario where we have 16 male patients and 16 female patients and we want to estimate a treatment effect using these patients. Here, we might want to assign half of the patients to a new adjuvant therapy and the other half to the current commonly used therapy. Assume we also know the age of the patients. We randomly assign half of the male and half of the female patients to case group and the other half to

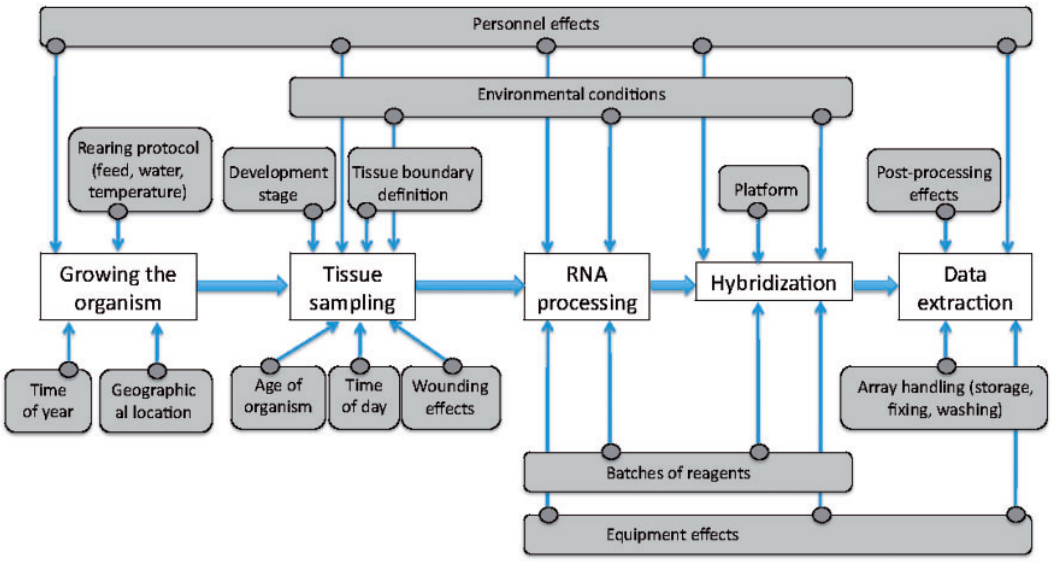


Figure 3.2: Potential sources of batch effect influencing the expression intensity in each of the four main stages of expression profiling, from Lazar et al. [2].

control. Here the treatment factor (case versus control) is the *primary variable of interest*, meaning that the primary goal of experiment is to study the influence of this variable on the gene expression values. The other biological factors (gender and age), which are known as *biological confounding factors*, are not of direct interest of study but they could potentially influence the expression values.

Furthermore assume, because of technical limitations, the maximum number of subjects we can process simultaneously is 8. Thus our batch size is limited to 8. Here we have to group our 32 patients into 4 batches of size 8. The batch assignment is a *technical confounding factor* that, similar to the biological confounding factors, might affect the expression values. In order to model the effect of the primary variable of interest correctly, the confounding factors need to be incorporated in the analysis. [2] Now consider these two designs:

- One might decide to form batches by grouping similar patients together, *i.e.,* we make the batches as homogeneous as possible. So one batch includes 8 male cases and another batch includes the 8 male controls. The two other batches are formed by grouping female patients in the same way.

- Another way to order this experiment is to test the outcome of interest on homogeneous subgroups, called *blocks*. First we form 4 blocks by assigning

[2]Factors that are not of direct interest but which might influence the outcome of experiment are known as "nuisance variables" in the experimental design literature [96].

	Male	Female
Batch1	Case(8)	
Batch2		Case(8)
Batch3	Control(8)	
Batch4		Control(8)

(a) Design in which case/control and sex may be confounded by batch effects

	Male	Female
Batch1	Case(4)-Control(4)	
Batch2	Case(4)-Control(4)	
Batch3	Case(4)-Control(4)	
Batch4	Case(4)-Control(4)	

(b) Design in which any batch effects can be identified and minimized

Figure 3.3: Two different settings for experimental order of 32 patients, half male half female, into 4 batches of size 8 and assigning them into case and control groups.

4 male and 4 female patients randomly to each block. Each block will be processed as one batch. Then within each block we assign 2 men and 2 women to case group and the other 4 to the control group.

Figure 3.3 summarizes these two design schemas. Now we want to analyze the output data of these two different designs, for example we want to run a significant analysis to find the genes most significantly related to the treatment factor. In the former case (Figure 3.3(a)), it is not possible to compare the gene expression of cases versus control without the influence of the technical confounding factor, *i.e.,* the batch effect. In fact, it is impossible to separate differences in gene expression values due to the treatment factor from differences that might be due to batch effects. Here when we pool all case subjects together (batch 1 and 3) and compare them with all control subjects (batch 2 and 4) the result will potentially be confounded by the batch effect. As a result, one might mistakenly declare some genes significantly associated with the treatment factor not because of biological differences but because of the batch effect. Subset comparisons also have the same problem. For example if one wants to evaluate the effect of the treatment factor only on male patients then batch 1 needs to be compared with batch 2. Again in this case, the outcome of interest is confounded with the batch effect because the experimental design did not take account of possible batch effects.

This is a common problem observed in many high throughput molecular studies, with examples published in prestigious journals such as Nature [97, 98]. The confounded design schema is appealing from a practical point of view as it is common practice to organize biologically similar samples in groups and hybridize them together [98]. Especially in large studies involving several research centers, one might think that assigning a treatment group to each center will make quality control eas-

	Male	Female
Batch1	Case(8)	
Batch2	Case(8)	
Batch3		Control(8)
Batch4		Control(8)

Figure 3.4: Another flawed design in which all male patients are assigned to the case group and all female patients are assigned to the control group.

ier. However, this design makes the treatment effect completely confounded with the technical differences between centers. This design is as flawed as choosing all male patients in a study for case and all female patients for control as shown in Figure 3.4. This design would confound the treatment effect with gender effect, making it impossible to estimate the treatment effect correctly. In such cases that experimental factor is confounded with technical factors, using any correction algorithm will effectively remove some of the biological signals [98].

In comparison, the later design schema (Figure 3.3(b)) does not suffer from the limitation of design (a). In this design, both biological and technical confounding factors are "blocked", meaning that the subjects are grouped into homogeneous subgroups, known as *blocks*, according to their gender and their processing batches. Ideally the variability within each block is less than variability of the entire sample (set of subjects). This will make the estimation of treatment effect more effective in each block, compared to the entire sample [99]. Pooling these more effective estimations across blocks will result in better estimation compared to the no blocking scenario.

The other biological confounding factor in this experiment, age, is randomized in the second design. We randomly assign 4 men and 4 women to each block regardless of their ages. In some other circumstances we could block this variable too. For example if we had 128 patients and batch sizes were 32 instead of 8, then we could rank 64 male and 64 female patients based on their age and group them into 4 sets. The youngest male and female group form the first block and will be processed as a batch. The second youngest group form the second block and so on. Now within each block, we randomly assign half patients to case and the other half to control. Blocking the age makes the subgroups more homogeneous compared to randomizing the age. Thus the estimation of treatment effect will be more effective.

The ultimate lesson is *"blocking what you can and randomizing what you can*

not" [97]. This is the most effective way to reduce the confounding influence of nuisance factors on the primary factor of interest. By blocking the measured biological and technical confounding factors, we minimize the chance that their effect will be mixed with the effect of primary factor of interest. However, since there might be many more unmeasured subtle potential biological confounding factors (such as race, smoking, weight, *etc.*) and technical confounding factors (technician who performed the hybridization process, the temperature and humidity of lab, the condition that specimens were preserved, ozone level [100], *etc*), randomly assigning instances to blocks, after counting for measured confounding factors, is the best strategy for reducing the chance of mixing the effect of the primary factor with the effect of confounding factors. For experiments that contain two or more treatment factors or two or more controlled nuisance factors, there are standard design techniques such as *split-plot* and *latin square* respectively [101].

The influence of biological confounding factors on the response variable is a well known fact in science. Therefore obvious biological confounding factors are properly blocked in most clinical studies, *e.g.,* it would by highly unusual to find a study where all cases are male patients and all controls are female patients. However, neglecting to block technical confounding factors, similar to the methodology of Figure 3.3(a), remains a major problem in high-throughput omics studies. Researchers have shown that significant gene lists corresponding to flawed experimental designs were substantially longer than gene lists obtained from similar studies where the same technical confounding factor was not present [98]. In fact ignoring the technical confounding factors makes the data set have a technical bias and consequently the result of downstream analysis will be misleadingly "too optimistic"–*i.e.,*

- In significant gene set analysis, when treatment groups are profiled on different batches, too many will appear to be differentially expressed between treatment groups. Here, most of the significant genes are caused by the batch effect.

- For the survival analysis, when the shorter survival and longer survival patients are profiled on different batches, too many genes will have significant p-values under the Cox model.

- In the case of training a classifier to distinguish between two phenotypes, when each batch contains only instances of one phenotype, the classifier's apparent performance (*e.g.,* 10 fold cross validation) will be artificially too high.

In all of these cases, since the primary factor of interest is "reinforced" with the batch effect, the performance of learning algorithms appear better than the situation where the study was blocked properly. In all three mentioned cases, the learning algorithms fit to a signal that is a mixture of the primary biological signal of interest and the signal of the technical factor. Therefore the results of such studies are not reproducible and the learned models are not applicable to other data sets. In other words, using an independent validation set to evaluate the findings of these learning algorithms will show a significant degradation in their performance. This is because we anticipate that the independent data set will not "benefit" from the same flawed experimental design as the original data set. Because of the complicated nature of technical confounding factors and their unpredictable effect on expression values, even if the new data set suffers from bad experimental design too, it is very unlikely that its technical bias is exactly the same as the original study's bias (*e.g.,* the exact same temperature and ozone level) and thus the bias will not affect the expression values of the new data set in the exact same way that technical bias of original data set did. We believe this might be one of the reasons that there is not much consensus between several significant gene sets found for the same biological phenomenon [46, 49, 102]

Poor experimental design also makes various batch correction methods inapplicable. Since the technical factors are completely confounding the biological factor of interest, it will be impossible to distinguish between these two signals. In other words, if we observe a significant change in the expression values of a gene, we can not decide whether this change is due to the biological factor or due to the technical factor. Therefore there will be no guarantee that utilizing batch effect correction methods would not result in losing useful biological signals of interest along with removing BE. So in the rest of this chapter we assume that BE correction algorithms are being applied to data sets that are properly blocked for their known technical confounding factors.

As Scherer [58] mentioned, we emphasize that different types of bias might affect the reproduciblity of a gene expression study's discoveries and limit their generalization power. Most notably, *selection bias* (the criterion used to select a "random sample" of population to include in the study) and *specimen collection bias* (how the specimens are selected and prepared to be profiled) each have a major impact on how the results of a study are utilized and interpreted. Note that these biases and

their effects on gene expression values is different from batch effects. Attempting to integrate a breast cancer study where all patients with poor clinical outcomes were excluded from the final results, with another breast cancer study where no filtering was applied to the set of participants, is fundamentally wrong and no batch correction method can fix this discrepancy. These types of bias are better studied in the context of *covariate shift* [103]. This thesis, however, will assume that batch effect correction algorithms are applied to data sets that include unbiased random samples of the same population.

3.4 Evaluation of batch effect minimization methods

Proper evaluation of batch effect correction algorithms, before running any downstream analysis on the corrected data, is as important as batch correction in the first place [2, 9]. Batch correction algorithms are utilized in order to remove the unwanted influence of technical factors and consequently magnify the biological signal in the data. "Under correcting" the batch effect will result in a situation where data is still confounded by technical factors and thus different batches are distinguishable only on the basis of the non-biological signal [104]. "Over correcting" the batch effect, however, might result in a distorted signal of interest, which might mislead the downstream analysis into conclusions that are even worse than what we would get without any batch correction. In this case, the combined data set contains less biological signal than its constituents [104]. Furthermore, as all sources of batch effects are usually not known, it is impossible to insure all removed signals are batch related and no useful biological information is lost during the correction procedure. This means a good BE algorithm should simultaneously removed confounding signals caused by technical factors and leave the biological signal unaffected. Therefore, assessing the performance of batch correction algorithms involves two competing criteria:

1. Confounding signal caused by technical factors is removed from data (batch effect is removed).

2. Biological signal of interest is left unchanged by batch correction algorithm (biological signal is retained).

Note the competition between these criteria: in the first criterion the batch correction algorithm attempts greater modification of the batches (to remove more batch

effect) while the second criterion attempts to modify the batches less (to retain the biological signal). We will see most of the performance assessment methods evaluate batch correction algorithms in only one aspect and ignore the other one. We believe that not evaluating both of these aspects might result in misleading down stream analysis of data. Lazar et al. [2] has a full review on the evaluation methods of batch correction algorithms. They divided available evaluation methods into two main groups: *visualization tools* and *quantitative tools*.

Visualization tools, mostly used for a fast inspection of the results, provide a crude assessment of batch correction efficiency. Visual evaluation can be done in either the "gene-wise" or "global" method. Gene-wise methods compare the distribution of genes across batches, one at a time. For example the probability density of a gene's expression should be the same across batches after batch correction. This is problematic as, unless the samples of two batches are selected from very similar populations with the same phenotype proportions across batches (*e.g.,* the tumor to healthy ratio or the cancer sub-type proportions), the distribution of gene expression values for two batches would not be the same even in the absence of batch effects.

The global visualization methods are less sensitive to the similarity of sample batches. These methods utilize a unsupervised learning algorithm (such as clustering and PCA plots) to depict the relation of instances of different batches with respect to each other. In the absence of batch effects we expect that instances of the same phenotypic class will group together regardless of their batches. For example, all male-female, tumor-healthy, or cancer sub-types – depending on the type of study – are expected to form specific clusters of their own. On the other hand, strong batch influence will make the instances of each batch to form a separate cluster, regardless of their biological properties.

As opposed to visualization tools, quantitative methods measure the "overlap" between the batches [2]. These methods attempt to measure how much the "clouds" of data points belong to each batch are mixed with each other. For this purpose, they either measure the distance of the closest instance of other batch to each data point or they count how many of an instance's K closest neighbors belong to another batch. By comparing these values before utilizing each batch effect correction algorithm and then after applying it, they can assess how effectively it minimizes the batch effect. The main shortcoming of these methods is that they fail

to look at both the aforementioned evaluation criteria; more specifically they fail to check whether the biological data is preserved after batch correction. Thus if a naive batch correction method simply transforms all gene expression arrays to one point, these methods evaluate its performance as outstanding. In the same category, Lazar et al. [2] reviewed a method that measures the symmetry of the distribution of gene expression values before and after the batch correction. This method declares a batch correction algorithm effective if it keeps the cumulative density functions before and after correction, symmetric. Again this method fails to evaluate both of the aforementioned criteria – here, this evaluation method does not check for effective removal of batch effect.

Lazar et al. [2] also reviewed a couple of quantitative evaluation methods based on differentially expressed genes. One of the methods uses a set of a priori known "positive control genes" – *i.e.,* genes that are known to be related to the biological phenomenon under study – in order to evaluate how effective a particular batch correction algorithm is. This method compares the proportions of significant differently expressed genes in common with the positive control gene set before and after the batch correction algorithm. A batch effect correction algorithm is effective if it increases the proportion of positive control genes that are declared significant.

Lazar et al. [2] reviewed another evaluation method based on differentially expressed genes, which was originally proposed by Sims et al. [86] and does not need any prior knowledge about positive genes. This method assesses the performance of a batch correction algorithm by finding the set of significant genes "between" batches and "across" batches, and compares them before and after applying the correction algorithm. Their method explicitly evaluates how effectively the batch effect is corrected, but it does not check for retaining the biological signal. In Chapter 5 we will extend this method to include both criteria explicitly in the evaluation method.

Another evaluation method, proposed by Shabalin et al. [104], is based on integrative correlation [105] (we study this method in detail in Section 4.2.1). It uses integrative correlation analysis to quantify the similarity between batches before and after applying correlation algorithm. This evaluation method ignores the second criteria and does not evaluate how much of the biological signal is preserved. Chapter 5 will introduce three methods for evaluating the performance of batch correction algorithms which explicitly consider both of these competing criteria.

In addition to these measures, Autio et al. [62] introduced six key properties for normalization methods that determine their influence on the gene expression matrices: (1) profile-wise normalization, (2) gene-wise normalization, (3) does it consider the array platform? (4) does it include scaling? (5) does it use distribution? (6) does it change the order of the values within a profile?

3.5 Summary

This chapter introduces the batch effect and looks at its different sources, specifically in gene expression data sets. By studying the confounding role of batch effects and the possibility to correct them, this chapter proposes a set of general guidelines for conducting gene expression analysis according to proper experimental design principles.

This chapter also looks at several different evaluation methods for measuring the effectiveness of batch correction algorithms. Following Lazar et al. [2], we group evaluation methods into two main sets: visualization methods and quantitative methods. We also propose two competing criteria that evaluation methods need to consider for proper quality assessment of batch correction algorithms. We will use these two criteria later in Chapter 5 and introduce 3 evaluation methods.

Chapter 4

Feature Selection in microarrays

In a typical learning setting, the number of subjects in the study (n), also known as sample size, is at least an order of magnitude larger than the number of explanatory variables (p), also known as features. In the other words, statistical learning algorithms are designed for $p \ll n$ situations. In high-throughput molecular approaches, however, we are typically faced with the opposite setting $n \ll p$. Gene expression studies, similar to other high throughput biological measurement technologies, result in a matrix whose rows each correspond to an individual, and whose columns each correspond to a feature; here there are many such features and relatively few instances. Applying standard statistical learning algorithms to these data matrices and getting useful results is very challenging if not impossible. One way to address this problem is to use only a subset of the features that most closely represent the biological signals of interest; this is called "feature selection". In this chapter we will be looking at the effects of different feature selection methods on the similarity between gene expression profiles.

This chapter first looks at three different ways to reduce the number of features and empirically studies their effect. Based on the experimental results, we advocate one method – variance-based feature selection – for gene expression studies. Section 4.4 then considers the effect of reducing features based on their variability within each batch. Section 4.5 then looks at the similarity of gene variances as a similarity measure between gene expression studies.

The main findings of these chapter can be summarized as follows:

- Gene expression data sets almost always contain some genes that artificially increase the correlation score, even between completely unrelated profiles.
- Variance-based gene selection is an effective way to reduce the feature space,

as well as improving the estimation of correlation scores.

- Comparing gene expression data sets based on their gene variance ranking is a good measure of their underlying differences.

4.1 Distribution of gene expression intensities

The main goal of this chapter is to find a way to order genes by their statistical properties in gene expression data sets and use this ordering to reduce the number of genes by removing genes with lower rank. The main criterion for assessing genes is their role in the relation between the gene expression profiles of two or more data sets. This means if a gene exhibits similar intensities in the expression profiles of two biologically similar instances and different intensities for two not-so-biologically-similar instances, regardless of the data set of origin of the profiles, then we rank that gene high. We are specifically interested in genes whose behavior is consistent *across data sets*, as this entire thesis is about relation across data sets and finding a way to combine these data sets.

A great way to visualize the relationship between genes for one pair of profiles is by means of scatter plots. Here, each gene is shown as one point in a two dimensional plane whose x-value represents the expression intensity of a gene in one profile and whose y-value represents its expression intensity in the other profile. There will be p points in the plot, one for showing the intensity of each gene in the profiles. As with all other scatter plots, points that are close to the $y = x$ line are those with high concordance between the two profiles and those closer to upper-left or bottom-right corners are those that have high intensity in one profile and low intensity in the other profile.

The main limitation of scatter plots is the fact that they only show the relation between *a single pair* of gene expression profiles. So if we want to compare two data sets, one containing n profiles and one containing m profiles, then we end up with $n \times m$ scatter plots, each containing p points. In this section we propose a novel way to summarize all these scatter plots in one scatter plot, which we call *multi-scatter plot*. In order to do this, we extract two summary statistics for each gene in each data sets, the mean intensity value (measure of its *location*) and the standard deviation (measure of its *dispersion*). Thus we summarize the $p \times n$ and $p \times m$ expression matrices with four $p \times 1$ vectors, two mean vectors and two standard

deviation vectors, one for each data set.

Using this summarization schema, we have four values for each gene. In order to show these four values in multi-scatter plots, we represent each gene with a *cross* rather than a point as in ordinary scatter plot. The xy-value of the center of each cross, *i.e.,* the *location*, shows the mean intensity of one gene in the pair of data sets and the length of the vertical and horizontal hands, which represent the *dispersion* of the gene, show the standard deviation of the genes in the pair of data sets. We used this technique to show the relation between four pairs of gene expression data sets in Figure 4.1.

In the experiments of Figure 4.1 and Figure 4.2 we used five gene expression data sets. In the rest of this chapter we will use these data sets again. All these gene expression studies that were conducted using the same technology, namely Affeymetrix U133A GeneChips. Normalization was performed using BrainArray [82] custom cdf files (entrez) version 17.0.0 and the RMA [106] function in R version 3.0.1 (known as *GoodSport*) and bioconductor version 2.12. This version of the cdf files contains 12098 probesets for U133A GeneChips and 18960 probesets for U133 plus 2.0, the other commonly used Affymetrix GeneChip. These two platforms share 12092 probesets using this cdf file. All of the experiments in this thesis use these 12092 probesets as the feature set for both types of Affymetrix GeneChips. This way we are able to merge data sets conducted on both U133A or U133 plus 2.0 GeneChips. Here is a brief summary of each of the five data sets:

- **Ovarian cancer**: `GSE26712` [107] includes 185 primary ovarian tumors (90 optimal and 95 suboptimal) and 10 normal ovarian surface epithelium.
- **Prostate cancer**: `GSE8218` [108] includes 148 prostate specimens with various amounts of tumor, stroma, BPH and atrophic gland.
- **Lung cancer**: `GSE10072` [109] includes 107 lung specimens that are based on 122 samples, of which 15 duplicates were averaged. There are 58 tumor and 49 non-tumor tissues in the data set from 20 never smokers, 26 former smokers, and 28 current smokers.
- **Breast cancer1**: `GSE2034` [110] includes 286 lymph node negative breast cancer specimens of which 180 were relapse free and 106 patients developed distant metastasis. 77 patients, out of 286 patients, are ER negative ($\approx 27\%$)

in this study.

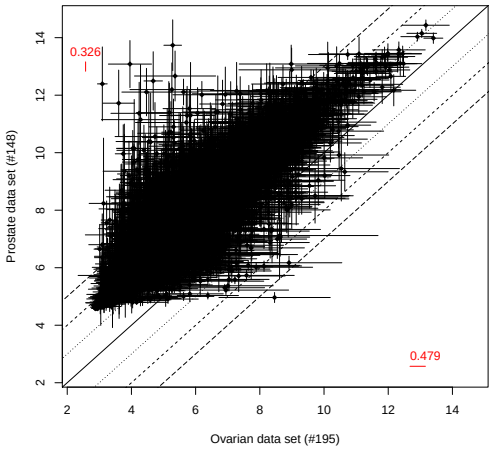
- **Breast cancer2**: `GSE7390` [111] includes 198 frozen primary breast cancer specimens of which 32% are pathologically tagged as ER negative.

The position of the crosses in Figure 4.1 represent the mean expression intensity of genes in the pair of data sets compared together and we interpret them the same way we interpret the points in an ordinary scatter plot. The length of the cross's hands is what makes this type of scatter plot different. Larger crosses represent genes with high variance in their expression intensities and smaller crosses represent genes with more steady expression values. When both hands of a cross are approximately the same length we can infer this gene has similar expression variances in the two data sets. However, when one hand of the cross is significantly longer than the other hand, we can infer that gene has significant different expression variances in the two data sets.

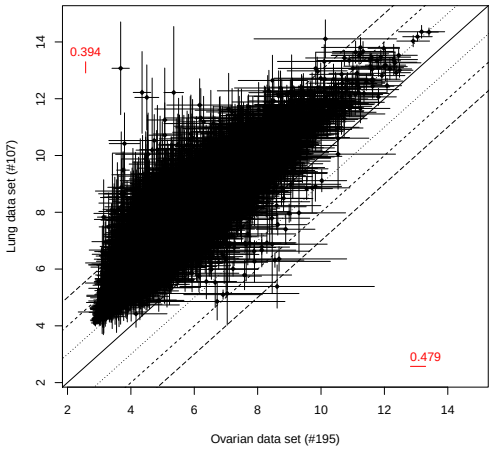
To better visualize the difference in the variance of genes across two data sets, one can remove the hand of the cross representing the data set with the lowest variance, *i.e.,* the shorter hand. Thus a horizontal line for a gene shows that specific gene has higher variance in the data set on x-axes. A vertical line shows that the vertical hand of that cross was longer so the variance of that gene in the data set belong to y-axes had higher variance. This modified version of multiple scatter plots of Figure 4.1 is shown in Figure 4.2.

There are a couple of interesting observations about Figure 4.2. One is about the relation between the expression intensities of genes and their variance. For example consider panels (a), (b), and (c) in Figure 4.2. As one can see, points under the $y = x$ line tend to be mostly horizontal lines and those that are above the $y = x$ line are mostly vertical lines. This means that genes with significantly higher intensities in one data set usually exhibit higher variances too. In the other words, genes with significantly different variations across two studies also have significantly different intensities.

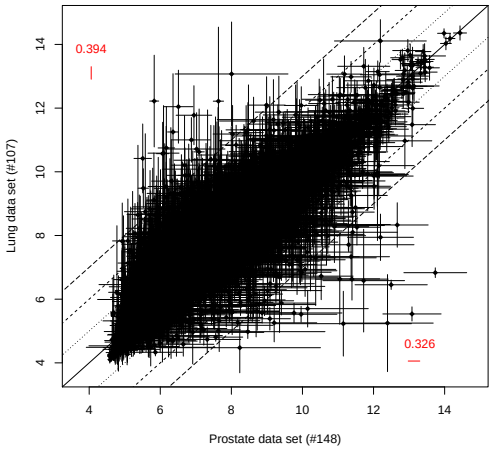
The other interesting observation relates to the average remaining standard deviation of data sets (written in red color in top-left and bottom-right corners of each panel) in panel (a), (b), and (c) in comparison to panel (d). These average values comparing the breast cancer data set 1 and 2 in panel (4.2d) are significantly smaller than comparisons between data sets of different cancer types the other three panels.



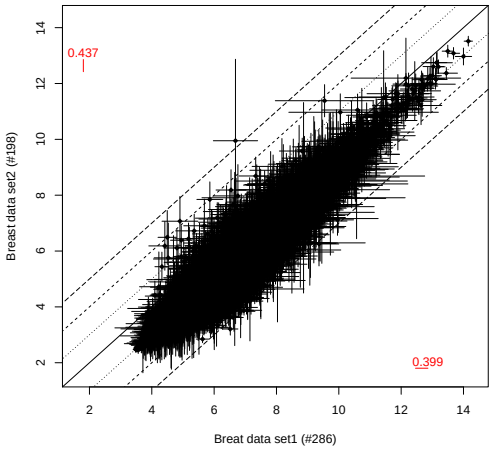
(a) Ovarian cancer versus prostate cancer



(b) Ovarian cancer versus lung cancer

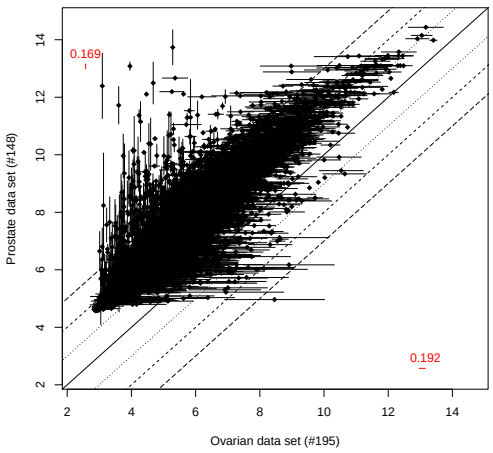


(c) Prostate cancer versus lung cancer

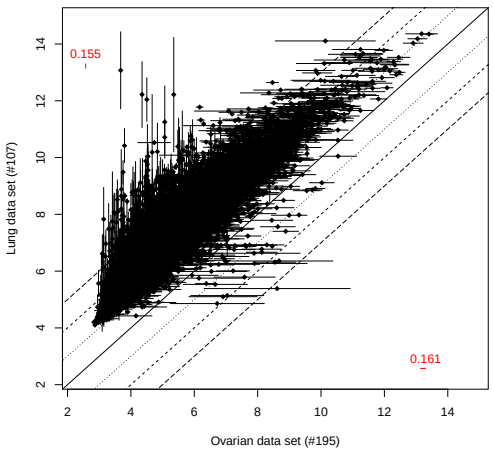


(d) Breast cancer1 versus breast cancer2

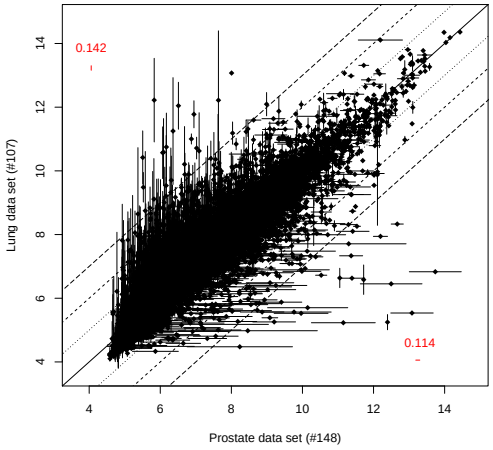
Figure 4.1: Four *multi-scatter plots* showing the pairwise relationship of the mean expression of 12092 genes between four pairs of data sets of the same or different cancer type, as indicated for each panel. The number of instances in each data set is shown in the axis labels. The red bar in top-left and bottom-right corners, shows the mean value of dispersion (mean of standard deviation of gene expression values) for the pair of data sets compared in each panel. The six lines parallel to $y = x$ line mark the ± 1, ± 2, and ± 3 intervals.



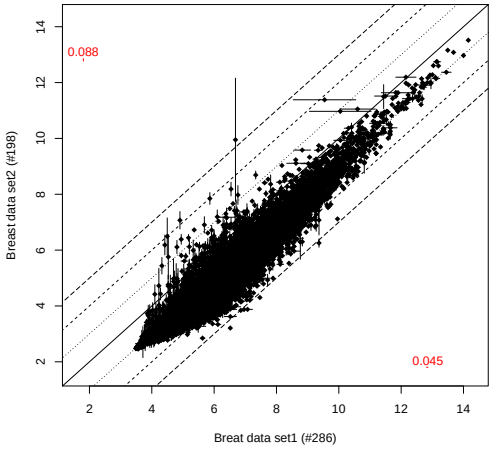
(a) Ovarian cancer versus prostate cancer



(b) Ovarian cancer versus lung cancer



(c) Prostate cancer versus lung cancer



(d) Breast cancer1 versus breast cancer2

Figure 4.2: Half crosses of multi-scatter plot – either horizontal or vertical lines depending on which data set standard deviation was larger. Horizontal lines represent the genes that have larger variance in the data set shown on x-axes. Vertical lines represent the genes that have larger variance in the data set shown on y-axes. The average of horizontal and vertical lines are shown by red color in bottom-right and top-left corners of each panel.

For the breast cancer pair the length of most of the lines is close to zero, which means in the original multi-scatter plot, *i.e.,* Figure 4.1(d), most of the crosses had hands with approximately equal lengths. Later in Section 4.4, we will see that using this observation, we can accurately determine the similarity of two data sets.

The following section introduces three feature selection methods that use the intrinsic statistical properties of genes to rank them. Subsets of genes chosen by these three algorithms will be assessed to examine the correlation coefficients across the paired data sets.

4.2 Feature reduction in gene expression data sets

There are many incentives to perform feature selection on gene expression data sets. As mentioned earlier, the large gap between the number of genes p and the sample size n is a major obstacle on applying most of the machine learning algorithms to this type of data. Thus reducing the size of feature set will improve the performance of learning algorithms. Some genes are only expressed in certain tissues or at certain times, in addition there may be artifacts, such as mislabeled spots on the arrays or chip-specific problems induced by the experimental conditions, that makes some of the measured transcription values unreliable [105]. It is therefore useful to utilize special techniques called *detection calls* [112–114] to answer the question "Is the gene detectable in the given sample?" [113]. Filtering genes based on the results of detection calls increases the number of differentially expressed genes while reducing the number of false positives [112, 113, 115]. Detection calls are also an effective method to detect transcripts whose expression levels are close to the limits of detection of the platform [112]. Running a feature selection step in order to remove features that are not detected is an important, but sometimes ignored, step in the downstream analysis of gene expression studies, which can improve interpretation of the findings.

Moreover, performing cross-platform or cross-study comparisons yields better results when we limit the analysis to a subset of substantial biologically relevant genes [116]. The subset of relevant genes depends on many different factors, such as the platform used, the tissue under study, the demographics of population and the way each study sampled from it. Nevertheless, identifying the relevant genes using the detection call filtering or *integrative correlation* has shown to result in more meaningful correlation scores across data sets [115, 117]. The result of different

studies have shown that identifying a subset of genes whose transcripts are reliably and consistently measured across studies is the first and most critical step in cross-study analysis of gene expression data sets.

In this section we empirically compare the reliability of gene expression measurements by applying different feature reduction methods and then quantifying to what extend they estimate the Pearson correlation coefficients across studies. As Pearson correlation coefficients do not rely on direct assimilation of expression intensities, they are commonly used for cross-platform and cross-study performance analysis of gene expression samples [105, 115, 117]. There are different ways to represent the Pearson correlation coefficients across two studies. More specifically, when we are comparing two studies, one with n instances and one with m instances, then we will be dealing with a $n \times m$ correlation matrix. In the following section we summarize this $n \times m$ matrix with the mean correlation value along with the 10 and 90 percentiles. The mean represents the average correlation score and it can be interpreted as the average similarity between the n instances of the first study to the m instances of the second one. The 10 to 90 percentile range represents the dynamic range of the correlation values across the two studies.

In the following section we introduce three novel methods to select genes across the gene expression data sets and compare them empirically with two widely used gene selection algorithms, namely *integrative correlation* and *detection calling*. Two of our methods, *intensity-based* and *variance-based*, are proposed based on the intuition of how we expect an informative gene should be distributed. The third method, *correlation increment*, is purposefully designed to choose a subset of genes that poorly represent the biological signal of interest. The performance of genes selected by this method will prove whether there are some genes in the gene expression data sets whose inclusion in downstream analysis will distort the results. In the following sections we first briefly introduce the available gene selection algorithms and then we explain our three proposed methods.

4.2.1 Integrative correlation analysis

Evaluating the consistency of transcript measurements of a gene across a pair of studies is easy if they contain matched instances. In this case, we expect the expression value patterns for a gene to be the same for the set of matched instances across studies – called "technical replicates". Therefore we can estimate the reliabil-

ity of measured values for a particular gene in each study and its consistency across studies by calculating the correlation between the two matched vectors of measured expression values of the gene in each study.

However, this luxurious condition does not exist in most cases, as our goal is typically to combine multiple gene expression studies to increase the sample size and consequently increase the statistical power. For this more practically interesting situation, Parmigiani et al. [118] proposed an idea that works based on *matching studies at the gene level*: While a pair of studies do not contain any matched instances, they do contain the same set of genes and assuming that their instances are sampled from the same population then the relationship between genes within each study should be similar across the studies. This method detects the inconsistencies of genes expression values by looking at how genes vary in relation to other genes within a data set. In their approach, in order to quantify gene's consistency and reproducibility across studies, they construct a "virtual sample" corresponding to each gene in each study and then calculate the correlation between these virtual samples across studies.

The virtual sample for a particular gene is constructed by calculating the correlation between its expression values and the genes expression values of each other gene over the set of arrays in each study. Assume the studies are $\mathbf{X}$ $[n \times p]$ and $\mathbf{Y}$ $[m \times p]$, each containing the expression values of the same p genes. For each gene i in the first study, they construct a sample with size $p-1$. Each of these $p-1$ values is the Pearson correlation between the expression vector of i^{th} gene and another gene calculated over the n instances of the first study. We show this sample as

$$\rho_i^{\mathbf{X}} = \left[\mathrm{Cor}^{\mathbf{X}}(i,1),\ \mathrm{Cor}^{\mathbf{X}}(i,2),\ \cdots,\ \mathrm{Cor}^{\mathbf{X}}(i,i-1),\ \mathrm{Cor}^{\mathbf{X}}(i,i+1),\ \cdots,\ \mathrm{Cor}^{\mathbf{X}}(i,p)\right]$$

where $\mathrm{Cor}^{\mathbf{X}}(i,j) = \mathrm{Cor}\left(\mathbf{X}[\cdot,i], \mathbf{X}[\cdot,j]\right)$ is the correlation between the i^{th} and j^{th} gene over the n instances of data set $\mathbf{X}$. Using the same process they construct the virtual sample for genes in the second study $\rho_i^{\mathbf{Y}}$. The size of the virtual samples of genes are again $p-1$ however each element of this virtual sample is calculated by calculating the correlation over m instances of $\mathbf{Y}$, *i.e.,* $\mathrm{Cor}^{\mathbf{Y}}(i,j) = \mathrm{Cor}\left(\mathbf{Y}[\cdot,i], \mathbf{Y}[\cdot,j]\right)$. The reproducibility of the i^{th} gene across the two studies is estimated by calculating the correlation of its virtual samples.

$$\rho_i = \mathrm{Cor}\left(\rho_i^{\mathbf{X}}, \rho_i^{\mathbf{Y}}\right)$$

Since the score of each gene is found by calculating the *correlation of correlation coefficients*, the method is called *integrative correlation* [105, 116, 118]. Integrative correlation identifies genes with concordant expression across studies [104], this idea is used in several studies to measure the consistency and reproducibility of cross-study and cross-platform gene expression profiling [119, 120], which show empirically that filtering of genes using integrative correlation can effectively increase the ratio of shared significant differentialy expressed genes across data sets.

4.2.2 Gene ranking analysis

In this section we introduce two of our proposed gene selection algorithms that work by matching ranked gene lists across two studies and then choosing genes by aggregating paired ranked lists. By extracting meaningful ranked lists from each study, we compare the patterns of gene expression values across studies. This allows us to perform gene selection across different types of chips (*e.g.,* Affymetrix oligonucleotide GeneChips versus spotted cDNA arrays), different generations of chips (*e.g.,* Affymetrix U133A versus U133 plus 2.0), and different studies that use same GeneChips arrays.

The core of the idea lays in an algorithm for combining two ranked gene lists. Consider two gene expression studies that share p common genes. We sort genes in each study according to some meaningful criterion. If we want to pick the top, say $p_1 < p$, features, we look at top of the two ranked lists of genes and find p_1 genes that are ranked high in both lists. Given the two ranked lists and the desired number of top genes p_1, algorithm 2 simply finds the top p_1 genes that both ranked lists agree on. This algorithm can easily be extended to handle more than two ranked lists.

Before discussing the way we rank genes in each study, we must consider a better implementation of Algorithm 2. Algorithm 2 is a very straightforward implementation for combining two ranked lists. However, it is not the most efficient one. With an efficient implementation of `intersect` function the worst case complexity of Algorithm 2, assuming $p_1 \ll p$, is $\mathrm{O}\left(p_1 p^2\right)$. The complexity of `intersect`$(A, B)$ where $||A|| = n$ and $||B|| = m$ is $\mathrm{O}\left(mn\right)$. However, if A and B are *partially ordered sets*, then we can find their `intersection` by first `sorting` them and then `merge` the sorted lists. Here the time complexities of the two steps are $\mathrm{O}\left(n\log(n) + m\log(m)\right)$ and $\mathrm{O}\left(m + n\right)$, respectively. Thus the time complexity of `intersect` function can be improved to $\mathrm{O}\left(n\log(n) + m\log(m)\right)$ if we can sort the two sets. In the case

Algorithm 2 How to combine two ranked list and find top n_1 elements that they both agree on. One can easily extend this algorithm to combine more than two ranked lists.

Require: $RankedList1\ [1 \cdots p]$
Require: $RankedList2\ [1 \cdots p]$
Require: number of desired features $q \leq p$
  $Threshold \leftarrow q$
  $TopRank \leftarrow RankedList1\ [1 \cdots Threshold] \cap RankedList2\ [1 \cdots Threshold]$
  while $length(TopRank) < q$ **do**
    $Threshold \leftarrow Threshold + 1$
    $TopRank \leftarrow RankedList1\ [1 \cdots Threshold] \cap RankedList2\ [1 \cdots Threshold]$
  end while
  return $TopRank[1 \cdots q]$

of ranked gene lists, we will be working with the index of genes so we can benefit from this more efficient implementation. Running an algorithm that has quadratic dependence on the number of genes is computationally very expensive. To solve this problem we proposed the following algorithm that produces the same results as Algorithm 2 with more efficient time complexity.

Algorithm 3, using the indexing variable i, analyzes ranked genes one by one, starting from the highest ranked genes in the two lists. It also keeps track of the genes that were claimed highly ranked in one of the two lists, but not both of them, using the *Unused* array. If the i^{th} element of the two ranked lists are the same then we add it to the result list. Otherwise we compare each of the two genes to the *Unused* list. If we can find each of them in the *Unused* list, it means this gene was added to the *Unused* list as it was considered highly ranked in the other list. Thus we add it to the result list and we remove it from the *Unused* list (only for better performance for future `find` calls). We repeat this process until the result list has a sufficient number (p_1) of elements.

Using an efficient implementation of the three main functions in Algorithm 3, *i.e.,* `find`, `add`, and `remove`, each iteration of the main `while` loop will take $\mathrm{O}(\log(p))$. [1] Since the main loop will execute at most $p/2+p_1$ times, the overall time complexity of the second implementation is $\mathrm{O}((p+p_1)\log(p))$, which is better than the quadratic performance of earlier implementation in Algorithm 2. One can easily extend this implementation to apply it to more than two ranked lists by utilizing a counter for

[1] A max (min) heap is able to `add` and `remove` elements on average with $\mathrm{O}(\log(n))$ where n is the length of the heap array. These two functions perform their task (adding a new element or removing an existing element, respectively) in a way that they preserve the special structure of the heap array. As a result, the `find` function is able to search for elements on average in $\mathrm{O}(\log(n))$.

Algorithm 3 More efficient implementation for combining two ranked list and find top p_1 elements that they both agree on. Here we showed the set difference operator by $\setminus$ symbol.

Require: $RankedList1\ [1 \cdots p]$
Require: $RankedList2\ [1 \cdots p]$
Require: number of desired features $q \leq p$
  $TopRank \leftarrow RankedList1\ [1 \cdots q] \cap RankedList2\ [1 \cdots q]$
  $All \leftarrow RankedList1\ [1 \cdots q] \cup RankedList2\ [1 \cdots q]$
  $Unused \leftarrow All \setminus TopRank$
  $i \leftarrow q + 1$
  while $length(TopRank) < q$ **do**
    if $RankedList1\ [i] == RankedList2\ [i]$ **then**
      $TopRank \leftarrow TopRank \cup RankedList1\ [i]$
    else
      if $(RankedList1\ [i] \in Unused) ==$ **true then**
        $TopRank \leftarrow TopRank \cup RankedList1\ [i]$
        $Unused \leftarrow Unused \setminus RankedList1\ [i]$
      else
        $Unused \leftarrow Unused \cup RankedList1\ [i]$
      end if
      if $(RankedList2\ [i] \in Unused) ==$ **true then**
        $TopRank \leftarrow TopRank \cup RankedList2\ [i]$
        $Unused \leftarrow Unused \setminus RankedList2\ [i]$
      else
        $Unused \leftarrow Unused \cup RankedList2\ [i]$
      end if
    end if
    $i \leftarrow i + 1$
  end while
  return $TopRank[1 \cdots q]$

each element in the *Unused* list. These counters keep track of the number of ranked lists that this gene has appeared in them as highly ranked. As soon as all lists (or maybe a majority of them) declare a gene as highly ranked, we will add it to the result array *TopRank*.

We utilized Algorithm 3 for two gene selection algorithms. The only difference between these two gene selection algorithm is the criterion that was used for ranking the genes in the paired studies. The first one, called *intensity-based* feature selection, ranks the genes based on their mean expression intensity (from highest to lowest) in each data set to form two ranked lists. These lists are fed into the Algorithm 3 to select the top p_1 genes. Here we assume that higher mean intensity values are due to real biological artifacts while low intensities are due to noise in measurements. Thus

by selecting genes with higher mean expression intensities, we are able to capture the biological differences between the instances more clearly. When we combine the ranked lists of two data sets using the Algorithm 3, we will be able to see these biological distinctions across data sets.

The second gene selection method, called *variance-based* feature selection, sorts the genes in each study using their variance and then selects the top genes with higher variance. Here we assume that variations of genes with low variance is likely due to random noise not caused by biological signals. Therefore genes with low variance will not be helpful in distinguishing between biological classes. This means genes that show significant variability across samples gain their high variance from the biological heterogeneity of samples and thus they are able to distinguish between the biological classes within each data set. Genes that have this property across data sets are expected to be able to distinguish between the biological classes across the data sets. Later in Section 4.3 we will compare the performance of these two methods.

4.2.3 Correlation increment gene selection algorithm

Our third gene selection method is purposefully designed to pick the most uninformative genes, to demonstrate how irrelevant some of the genes are, especially for learning tasks. This emphasizes the importance of performing feature selection on gene expression data sets before conducting any downstream learning analysis on data in order to remove these irrelevant genes. As we will see in Section 7.1, these genes do not contain any biological information and their expression intensities are highly correlated across different data sets, regardless of the biological phenomenon under study. Removing these genes will significantly increase the average correlation coefficients between the samples across data sets.

This feature selection method, which we call *correlation increment*, applies a heuristic search to find genes that increase the average correlation between gene expression profiles across a pair of data sets as much as possible. For this purpose, we use the following formulation of Pearson correlation between two vectors of $\mathbf{x} = [x_1, x_2, \cdots, x_q]^T$ and $\mathbf{y} = [y_1, y_2, \cdots, y_q]^T$

$$\mathrm{Cor}(\mathbf{x}, \mathbf{y}) = \frac{1}{q-1} \sum_{i=1}^{q} \left(\frac{x_i - \bar{\mathbf{x}}}{S_{\mathbf{x}}} \right) \left(\frac{y_i - \bar{\mathbf{y}}}{S_{\mathbf{y}}} \right) \tag{4.1}$$

where $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ are the mean value of vectors $\mathbf{x}$ and $\mathbf{y}$ and $S_{\mathbf{x}}$ and $S_{\mathbf{y}}$ are their

respective standard deviations. Equation 4.1 expresses the Pearson correlation as the average of multiplication of standardized features (z-scores) of the components of $\mathbf{x}$ and $\mathbf{y}$. We use this formulation to estimate the effect of adding a new feature, x_{p_1+1} and y_{p_1+1}, on the correlation value between $\mathbf{x}$ and $\mathbf{y}$. If we want to find *the exact* value of correlation after adding the new feature, we have to recalculate the mean and standard deviation values and then use Equation 4.1, this time with p_1+1 terms inside the sum. However, if we just want to approximate the effect of adding the new feature, we can assume that means and standard deviations of two vectors ($\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$, $S_{\mathbf{x}}$, and $S_{\mathbf{y}}$) are unchanged. Using this simplifying assumption, including a new feature only adds one new term to the sum. Here is the approximated new correlation value between two vectors of $p_1 + 1$ features.

$$\mathrm{Cor}\left(\,[\mathbf{x}, x_{q+1}]\;,\;[\mathbf{y}, y_{q+1}]\,\right) \approx \frac{q-1}{q}\mathrm{Cor}(\mathbf{x},\mathbf{y}) \;+\; \frac{1}{q}\left(\frac{x_{q+1}-\bar{\mathbf{x}}}{S_{\mathbf{x}}}\right)\left(\frac{y_{q+1}-\bar{\mathbf{y}}}{S_{\mathbf{y}}}\right) \tag{4.2}$$

In fact, if we only intend to compare the effect of adding the $(p_1+1)^{th}$ feature to $\mathbf{x}$ and $\mathbf{y}$ for all available features (x_i, y_i), we only need the numerator of the second term. The following equation sets $\underset{(x_i,y_i)}{\Delta\mathrm{Cor}}(\mathbf{x},\mathbf{y})$ to approximate the effect of adding a new feature (x_i, y_i) on the correlation between the two vectors of $\mathbf{x}$ and $\mathbf{y}$:

$$\underset{(x_i,y_i)}{\Delta\mathrm{Cor}}(\mathbf{x},\mathbf{y}) \approx (x_i-\bar{\mathbf{x}})\,(y_i-\bar{\mathbf{y}}) \tag{4.3}$$

The complexity analysis of Equation 4.3 shows why we need this approximation. We assume the total number of features is p, the initial length of $\mathbf{x}$ and $\mathbf{y}$ is p_1, and we want to add k new elements to these two vectors where $p_1 \ll p$ and $k \ll p$. Adding each new element requires calculating Equation 4.3 for all p available genes $\mathrm{O}\,(p)$ and finding one feature that has maximum the value of Equation 4.3.

$$index = \underset{i}{Argmax}\left(\underset{(x_i,y_i)}{\Delta\mathrm{Cor}}(\mathbf{x},\mathbf{y})\right)$$
$$\mathbf{x} = [\mathbf{x}, x_{index}] \quad \mathbf{y} = [\mathbf{y}, y_{index}]$$

After finding this feature, we concatenate it to the existing $\mathbf{x}$ and $\mathbf{y}$ and then need to recalculate the mean and standard deviations $\mathrm{O}\,(p_1+k)$. As we need to repeat these two steps k times, the time complexity of using Equation 4.3 is $\mathrm{O}\,(k(p+p_1+k))$.

In comparison, if we want to calculate the exact value of correlation using Equation 4.1, we need to recalculate the means and standard deviation for evaluating each feature and the complexity would be $\mathrm{O}\,(kp(p_1+k))$. In this implementation

for adding each of the k new elements, we need to consider all p available features, which for each we need to recalculate the mean and standard deviation. Thus adding one new element will take $\mathrm{O}\left(p(b+k)\right)$ and we need to repeat this k times. Comparing this complexity with the approximate version complexity shows we are saving a lot in the computations using the approximation of Equation 4.3. This saving is more significant when we take into account that we calculate the cross-correlation matrix between several $\mathbf{x}$ vectors and several $\mathbf{y}$ vectors rather than only one correlation score.

We can use Equation 4.3 to estimate how much a new feature on average changes the correlation values between a set of $\mathbf{x}$ vectors a the set of $\mathbf{y}$ vectors, *i.e.,* the cross correlation between two data sets $\mathbf{X}$ and $\mathbf{Y}$. Our third proposed feature selection algorithm will use this heuristic to reduce the number of features. At each step, it starts with the current set of selected features then uses Equation 4.3 to find the one feature that increases the correlation the most and add it to the set of selected features. We recalculate the $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$, $S_{\mathbf{x}}$, and $S_{\mathbf{y}}$ for the new set of features and continue until we reach the desired number of features. We compare the effect of these three feature selection algorithms empirically on the cross correlation between two sets of gene expression profiles.

4.3 Comparing the performance of feature selection methods

In this section we compare the performance of the five feature selection methods applied to gene expression data sets. For performance analysis we conduct the correlation analysis that is common in the context of gene expression data [90, 105, 116, 118]. For this purpose we use the cancer data sets we mentioned in Section 4.1. We pair two gene expression data sets and then study the distribution of cross correlation coefficients. Hence if one data set has n instances and the other data set has m instances, there will be a $n \times m$ correlation coefficient matrix. In the following we look at the distribution of these values using three summarizing values, the mean of $n \times m$ correlation coefficients plus the 10% and 90% intervals.

In these comparisons we pair apparently completely unrelated data sets together so a feature selection method that leads to a lower mean correlation is assumed to be more accurate. Also a larger $10\% - 90\%$ interval shows a larger dynamic range of correlation coefficients, which suggests the feature set is able to capture the fine

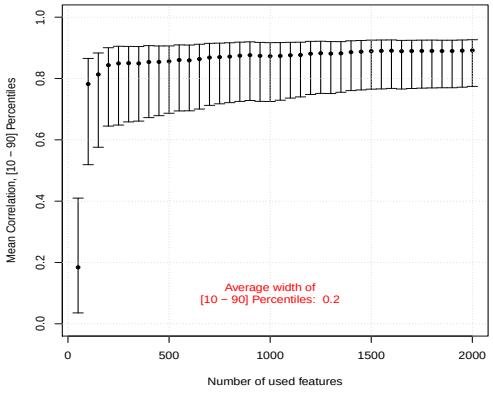
similarities between pairs of gene expression profiles. Thus feature sets that result in larger intervals are more desired comparing to feature set with shorter intervals.

Figure 4.3 shows the cross correlation coefficients between the ovarian cancer data set and prostate cancer data sets. We measure the similarity between two gene expression profiles using the Pearson correlation. The cross correlation matrix includes $195 \times 148 = 28860$ values. We plot the mean correlation value of 28860 correlation scores (black dots) along with their 10% to 90% percentiles as the lower and upper bars in Figure 4.3. In this figure we measure the correlation values using several sets of features with different sizes chosen by four gene selection methods of Section 4.2. In all 6 panels we started with a same set of 50 base features (which were selected using the variance ranking method to have a fair comparison between all methods) and then we incrementally added 100 more features each time, up to the limit of 2000. Panel (a) shows the result of integrative correlation feature selection, panel (b) shows the correlation increment gene selection method, panel (c) shows the intensity-based ranking feature selection, and (d) show the variance-based feature selection results. In each of these four panels, the average $10\% - 90\%$ interval length is shown with red color.

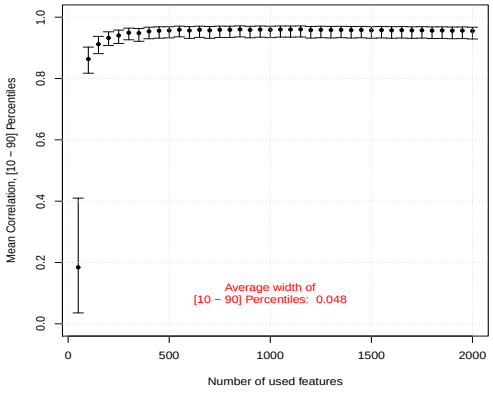
In order to better compare the behavior of the four feature selection methods, we plot the average of 28860 correlation scores between ovarian samples and prostate samples using different feature sets in panel (e) and (f). Panel (e) summarizes the mean correlations coefficients shown in panel (a)-(d) for the gene set of size up to 2000. Panel (f) shows the same result over the course of entire possible gene set sizes $1 - 12092$. As one can see, all four lines start from a same point with average correlation of approximately 0.18 as we start with a same subset genes for all four feature selection methods. They also converge to the same correlation value of 0.83 for which all 12092 genes are used. The correlation increment gene selection method manages to increase the average correlation to around 0.95 by using 1000 irrelevant genes. This artificially high average correlation score decreases later on when meaningful genes are incorporated in the model. Also note that variance-based method has significantly low correlation values in the range of [1, 3000] and after that it converges towards the mean-based feature selection results.

There are some important observations related to Figure 4.3 that one needs to consider:

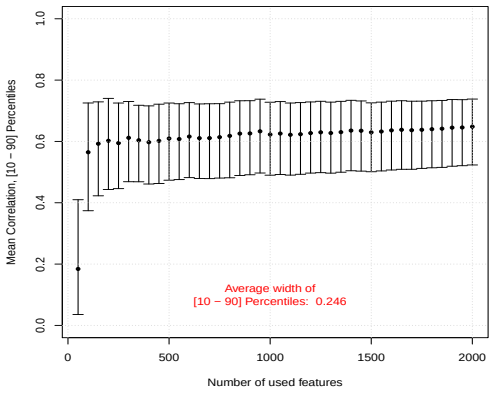
- There are many genes in these data sets that according to them unrelated



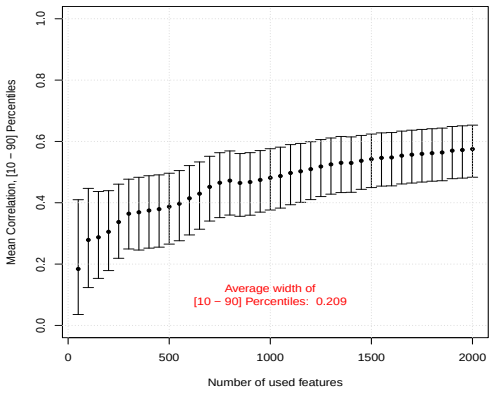
(a) Integrative correlation feature selection



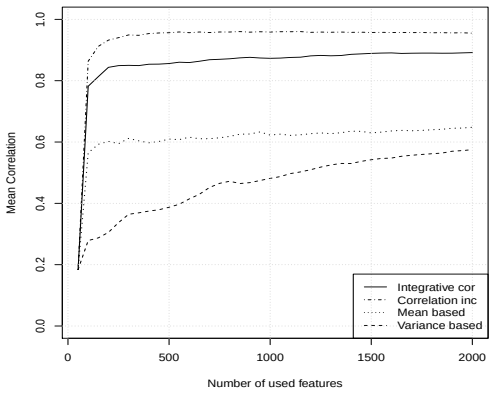
(b) Correlation increment feature selection



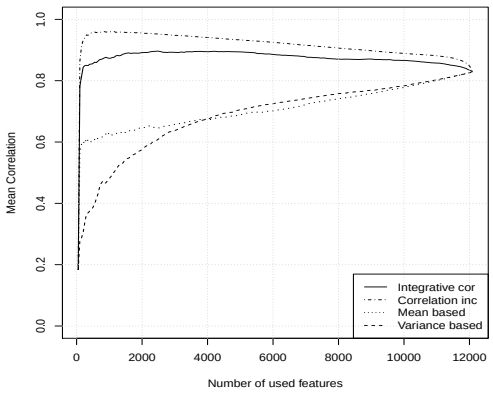
(c) Intensity-based ranking feature selection



(d) Variance-based ranking feature selection



(e) Comparison of all four methods



(f) Comparison of all four methods over the range of all genes

Figure 4.3: Cross-correlation scores between 195 ovarian cancer instances and 148 prostate cancer instances are plotted as a function of the number of selected genes.
(a)-(d)The $195 \times 148 = 28860$ correlation scores are summarized by three values: the mean (black dots) and the 10% and 90% percentiles (upper and lower bars) for 4 gene selection algorithms, as indicated in the captions. The average $10\% - 90\%$ interval length is written in red on each panel.
(e),(f) Average of cross correlation coefficient (without the $10\% - 90\%$ bars) between ovarian cancer samples and prostate cancer samples plotted as a function of the number of genes selected by four feature selection algorithms (a)-(d) panels for easier comparison of their performance. (e) has limited x-axis range ($1 - 2000$ similar to (a)-(d)) while (f) covers the entire available genes $1 - 12092$.

profiles appear to be highly correlated. Panel (f) of Figure 4.3 shows this phenomenon, as the plots of integrative correlation feature selection and correlation increment feature selection method increase to respectively 0.95 and 0.98 using a limited number of genes (less than 2000) and later decreases to 0.82 when all genes are included. This indicates these two feature selection methods are able to select a subset of "problematic genes" genes that claim the average correlation between 195 ovarian cancer specimens and 148 prostate cancer specimens is more than 0.95! By incorporating other genes in the calculations, the effect of these problematic genes is reduced and the average correlation score decreases to 0.8.

- Using misleading highly-correlated genes also decreases the "dynamic range" of correlation values. The average width of 10% to 90% percentile for correlation increment method is 0.048 while for intensity-based, variance-based, and integrative correlation gene selection method this value is 0.246, 0.209, and 0.200 respectively.

  This means that the whole 28860 correlation values tend to converge to a very small range of values (*i.e.,* very short 10% and 90% bars) when we use correlation increment gene selection method. This subset of genes fails to capture the fine differences between pairs of profiles as they suggest that all correlation scores are almost the same.

- Among these three methods, the genes selected by variance-based method tend to result in the lowest average correlation coefficients.

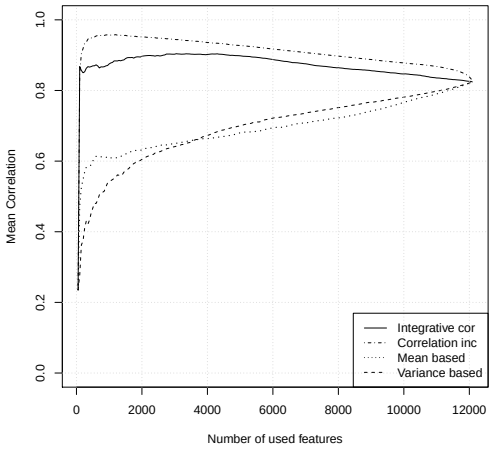
  Using the variance-based feature selection we were able to select 3000 genes that result in correlation coefficients that are significantly less than the correlation values resulted by the other three feature selection methods. Moreover, the dynamic range of correlation values is almost as high as the mean-based feature selection method.

- Panels (e) and (f), which compare the integrative correlation feature selection method with the novel correlation increment feature selection method, shows this new method is more successful in finding genes that can increase the cross correlation coefficient. Despite the fact that integrative correlation feature selection method was originally proposed to find the set of genes that increase

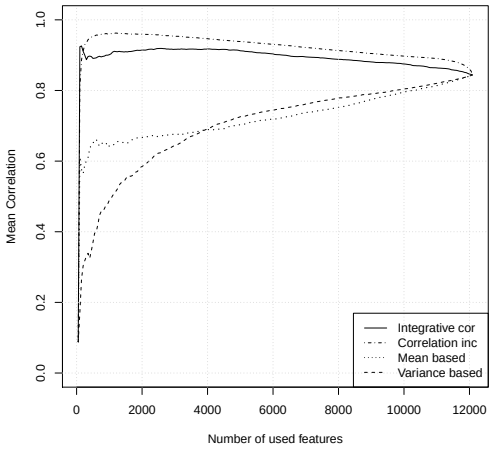
the cross correlation between the two biologically related data sets, here our simple heuristic is more successful in finding these genes.

More meaningful correlation scores when using only the high variance genes is not a surprise and can be explained by biological knowledge. Genes with high variance are in fact *genuinely expressed* genes in the biological specimen under study. The variation of expression values is due to existing differences of mRNA abundance in the population, *i.e.,* a transcribed gene in one specimen has a different expression value in another specimen due to the intrinsic differences between two individuals. In comparison, low variance genes are most likely just noise, whose variability is due to inaccuracies of measurement. The intuition behind the variance-based gene selection is very similar to the intuition behind the well known feature selection method *principal component analysis (PCA)*, which assumes that high variance directions contain more information that low variance directions. De Bie et al. [7] explains that, if we assume noise is uniformly spread, then high variance directions will have higher signal to noise ratio.

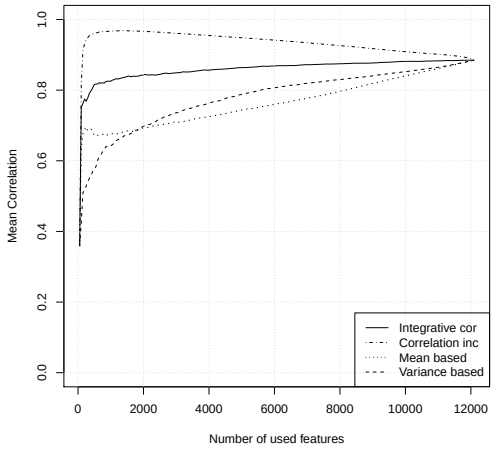
This is a general trend, which we observed for every pair of studies that we compared. In Figure 4.4 we show three plots similar to Figure 4.3 panel (f). Panel (a) compares the 195 ovarian samples with 107 lung cancer specimens, which were both profiled using the same Affymetrix U133a microarray GeneChips. Panel (b) compares the 148 prostate cancer samples to the lung cancer ones. These are two other examples of gene expression data sets that are expected to have lower cross correlation coefficients. We again observe that there are some genes that make the average cross correlation values very high and close to 1, and these genes are selected by our correlation increment gene selection. In these two cases, again, variance-based feature selection is able to successfully choose up to 3000 genes that make the average correlation score significantly smaller and depict the expected cross correlation better than other two methods. Panel (c), unlike the other two panels, compare two similar data sets, *i.e.,* two breast cancer data sets. Comparing panel (c) to panel (a) and (b) shows that the variance-based feature selection has significantly higher cross correlation coefficients while the other 3 methods exhibits approximately similar behaviors for similar paired data sets (c) and different paired data sets (a) and (b).



(a) correlation between 195 ovarian cancer and 107 lung cancer cases



(b) correlation between 148 prostate cancer and 107 lung cancer cases



(c) correlation between 286 breast cancer and 198 other breast cancer cases belong to another data set

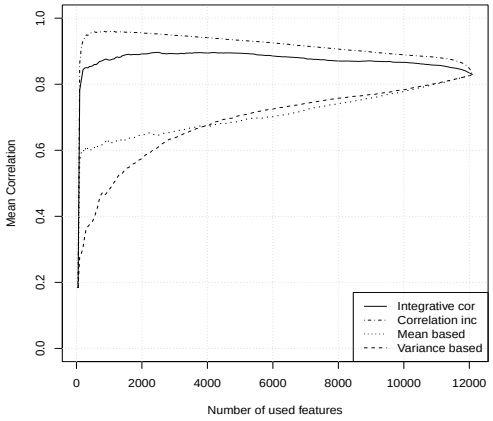
Figure 4.4: Average of cross correlation scores between three pairs of gene expression data sets, plotted as a function of the number of used genes selected by four different gene selection methods. The top row (a), (b) compares two unrelated data sets while the bottom row (c) compares a pair of related data sets.

Comparing proposed feature selection methods when combined with Detection calls

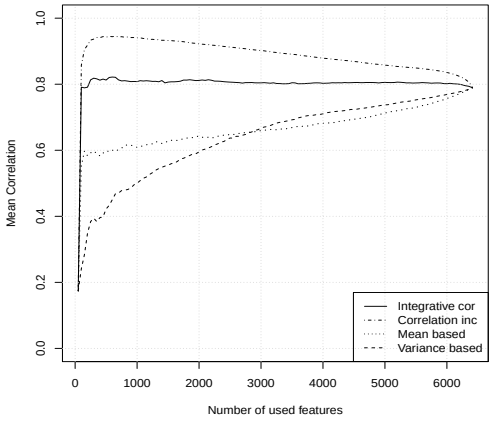
In order to incorporate the effect of detection call algorithm and removing genes labeled as "absent", we repeated the experiment of Figure 4.3(f). This time, before applying any gene selection algorithm, we first apply the detection call (DC) algorithm, which removes all genes that seemed to have very little amount of detectable mRNA abundance. We ran the DC algorithm using each of the five different threshold values: 0.25, 0.40, 0.55, 0.70, and 0.85. The threshold value determines the proportion of arrays needed to declare a gene as "present" in order to retain that genes. For example, picking the threshold to be 0.25 means that we tag a gene as present if at least 25% of arrays in a data set mark that gene as"present" (*i.e.,* the expression of gene is above a certain level)and we retain it as a feature in the data set. When we pair two data sets, we only retain genes that are marked as present in both data sets. Figure 4.5 shows the results.

As one can see, applying detection calls at different threshold levels reduced the number of genes used in analysis from 12092 down to 6378, 5705, 5028, 4220, and 3025 respectively for threshold values of 0.25, 0.40, 0.55, 0.70, and 0.85. Note that deploying detection calls slightly decreases the average correlation values, compared to tagging all features as "present". The first row of Table 4.1 summarizes the average correlation coefficients using all genes.

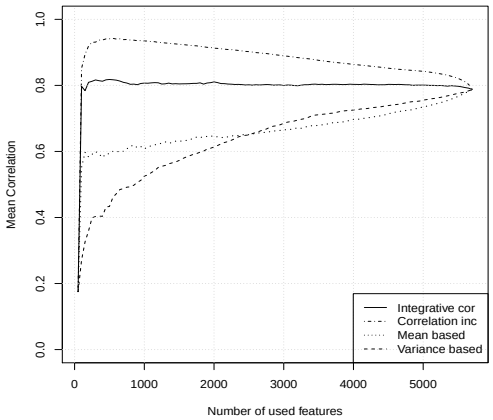
Figure 4.5 shows that more aggressive detection call threshold values push the lower curves (mean-based and variance-based feature selection methods) toward higher curves (correlation increment and integrative correlation methods) and decreases the gap between them. In order to better observe this phenomenon, Table 4.1 summarizes the average correlation values of 2000 genes selected by 4 different feature selection methods for each detection call threshold. This table shows only one data point of Figure 4.5 for each curve in each panel, *i.e.,* the value of curves at $x = 2000$. The last row of this table includes the "maximum gap", which is the difference between the lowest value and the highest value of each column. Examining the values of "maximum gap" shows that deploying detection call decreases the gap between the high and low curves. As one can see, the gap size shrinks when we deploy detection call algorithm with more aggressive threshold values. This means that applying detection call reduces the significant difference between the performance of different features selection algorithms, which in turn means that deploying



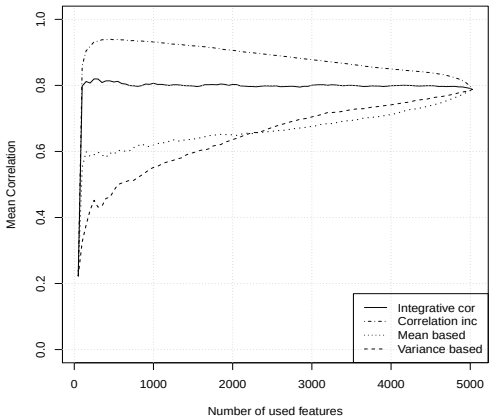
(a) Without Detection Call (12092)



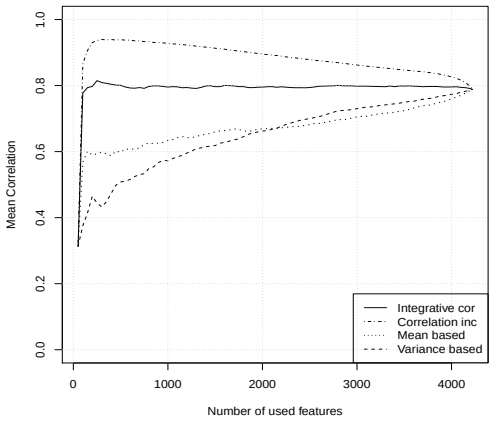
(b) At least 25% *Present* (6413)



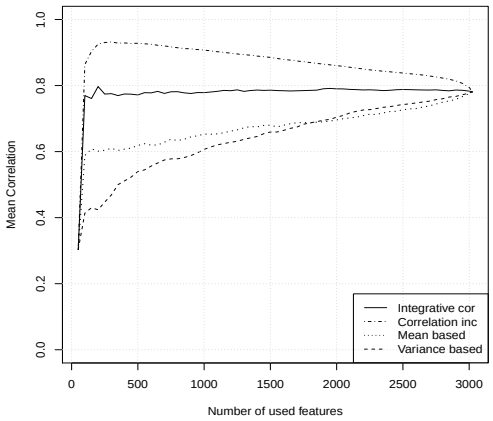
(c) At least 40% *Present* (5705)



(d) At least 55% *Present* (5028)



(e) At least 70% *Present* (4220)



(f) At least 85% *Present* (3025)

Figure 4.5: Utilizing the 4 aforementioned gene selection methods after applying the detection call algorithm to the ovarian cancer and prostate cancer paired data sets. In (a)-(f), different threshold values were applied for detection call algorithm to filter out "absent" genes and retain "present" ones. The number of remaining genes after applying the detection call is written under each panel in the brackets. Note the 6 panels have different x-axes scales.

	No DC	DC=25%	DC=40%	DC=55%	DC=70%	DC=85%
No of remaining genes	[12092]	[6413]	[5705]	[5028]	[4220]	[3025]
Avg Cor - all remaining genes	0.829	0.789	0.787	0.788	0.789	0.781
Integrative correlation	0.956	0.924	0.915	0.907	0.897	0.863
Correlation increment	0.891	0.811	0.808	0.800	0.795	0.791
Mean-based ranking	0.646	0.641	0.648	0.651	0.668	0.692
Variance-based ranking	0.572	0.590	0.610	0.632	0.660	0.698
Maximum gap	0.384	0.334	0.305	0.275	0.237	0.171

Table 4.1: The average cross correlation coefficient of 195 ovarian cancer instances and 148 prostate cancer instances using 2000 genes selected by 4 different feature selection algorithms under different threshold values for detection call (DC) method. Values in this table are extracted from Figure 4.5 by recording the values of each curve in each panel at $x = 2000$.
The top row shows the number of genes that passed the criterion of detection call algorithm, as well as the average cross correlation values over all genes that pass the detection call threshold.

the detection call algorithm reduces the need to perform feature selection.

However, examining Figure 4.5 and especially the correlation increment feature selection curve indicates that the detection call algorithm does not appear to remove genes that increase the average cross correlation values artificially. In other words, correlation increment curves in panels (b)-(f) follows the same trend as its curve in panel (a), where no detection call is applied. In all panels shown in Figure 4.5, each correlation increment curve has a pick point when approximately 10% of the genes are selected and then it decreases to the final value (the average cross correlation value produced when using all available genes). This shows that filtering genes based on DC metrics does not remove all "problematic" genes that cause high correlation across biologically different tumor types and indicates that deploying the DC algorithm, while it removes some of these genes, it does not completely eliminate the problem caused by them and thus applying a feature selection algorithm is still necessary.

We designed another experiment to show that some genes do no distinguish between different biological phenomena, *i.e.,* genes whose expression values seems independent of biological characteristics such as the tissue of origin. This experiment uses the five data sets introduced in Section 4.1. We merge all their instances together to form one data set with $195 + 148 + 286 + 107 + 198 = 934$ instances of 12,092 measured gene expressions, *i.e.,* a 934×12092 matrix of expression values. We then calculate the correlation between all pairs of these 934 instances using all genes. The resulting correlation matrix is shown in Figure 4.6(a) and assesses how

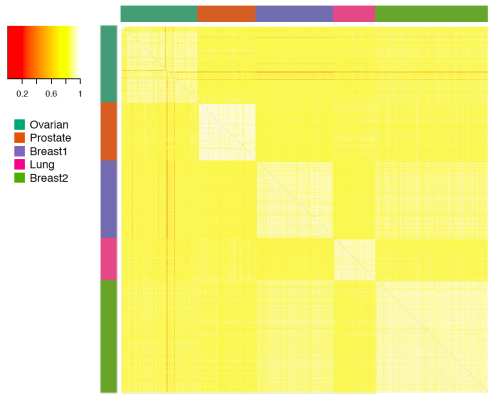
the correlation matrix depends on the feature set used in its construction. Hence we apply the four aforementioned feature selection methods to this large data set and choose four sets of selected genes. For this experiment we chose 150 genes using each feature selection algorithm, however, these results are approximately the same for any sized gene set less than 2000. We used these four sets of genes to reduce the number of genes from 12092 to 150 and then recalculate the correlation between all pairs of instances in 5 data sets.

Note that the resulting 934×934 correlation matrices, similar to Figure 4.6(a), are symmetrical so we only plot half of them in each of the 4 parts of Figure 4.6(b). *(b)-(i)* plots the correlation matrix between the 934 gene expression instances using the 150 genes selected by the correlation increment feature selection algorithm. *(b)-(ii)* shows the correlation scores resulting from features selected by integrative correlation feature selection. Intensity-based ranking and variance-based ranking results are shown in *(b)-(iii)* and *(b)-(iv)* respectively.

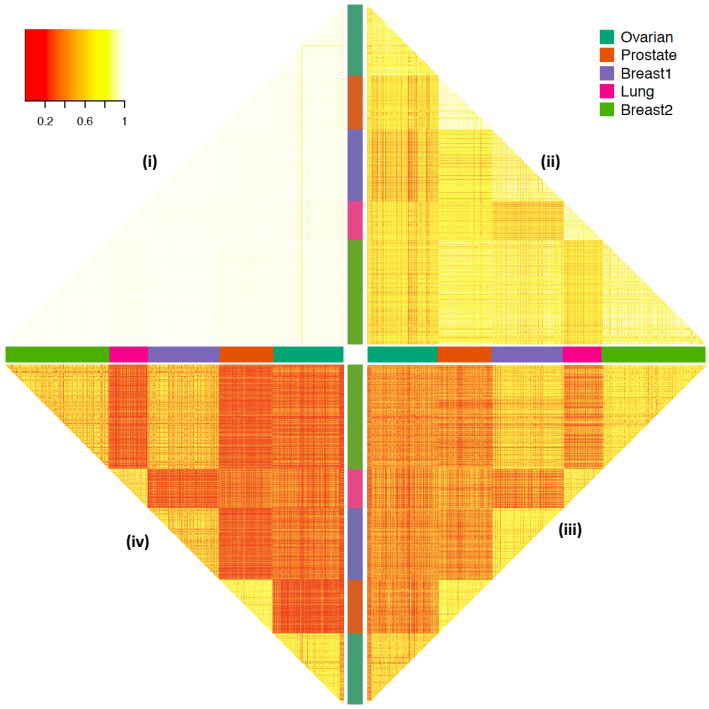
There are a few key properties we ideally expect to observe in this big correlation matrix:

- "Within data set correlation coefficients", should be high. This occurs here, as we see bright yellow squares on the main diagonal of Figure 4.6 (a) and also bright yellow triangles on the edges of each four parts of Figure 4.6 (b).
- "Cross data set correlation coefficients" should be low. This occurs here, as we see red rectangles in the (half) correlation matrices.
- "Cross breast cancer dataset correlation coefficients" should be higher than other cross correlation values but still lower than within correlation values.

Out of the five correlation matrices plotted in Figure 4.6, the one produced by variance-based gene selection (b)-(iv) is closest to our expectations, representing the biological rather than the technical differences. As one would expect, each data set has high a correlation with itself (the 5 "yellow" triangles on the edge of the panel (b)-(iv)) while the correlation across data sets is much lower (multiple "red" rectangular blocks of panel (b)-(iv)). Moreover, the cross correlation between the two breast cancer studies is relatively higher than other cross correlation scores (the only "yellow" rectangular block in panel (b)-(iv)). In contrast to this close-to-ideal behavior, panel (b)-(i) shows essentially no visible difference between correlation



(a) 934 × 934 correlation coefficient matrix calculated using all 12092 genes



(b) Four 934 (half) correlation matrices calculated using four set of 150 genes selected by four algorithms

Figure 4.6: Correlation heatmap between 934 instances of 5 data sets calculated using different sets of genes:
(a) All 12092 available genes
(b)-(i) 150 genes selected using correlation increment feature selection algorithms.
(b)-(ii) 150 genes selected by integrative correlation feature selection algorithm.
(b)-(iii) 150 genes selected by intensity-based ranking feature selection algorithm.
(b)-(iv) 150 genes selected by variance-based ranking feature selection algorithm.
Yellow to white colors represent high correlations, orange and red represent the lowest correlations.

scores within data sets and across data sets. In other words, according to the 150 genes selected by correlation increment gene selection method, gene expression profiles of different cancer types, originating from different tissue specimens, belonging to different patients, are all similar! This clearly shows that these irrelevant genes artificially increases the correlation coefficients between gene expression profiles and looses all meaningful biological distinction between gene expression profiles.

Between these two extreme cases lays the integrative correlation and intensity-based ranking feature selection methods in panels (b)-(ii) and (b)-(iii) respectively. Sets of 150 features selected by these two feature selection methods are able to distinguish between within data set correlation scores (yellow triangles) and across data set scores (red triangles). However, the distinction is less emphasized (higher correlations between cancers from different sites) comparing to the variance-based selected features.

Finally, by studying panel (a), one can say that using all available features to construct the correlation matrix will definitely eliminate the range of correlation scores, *i.e.,* the gap between scores of similar instances (yellow regions) and not similar instances (red regions) is less emphasized in the correlation matrix (a) as most of the scores are close to 1. This is expected as we are using all genes, including those in panel (b)-(i) that make all instances look highly correlated. This experiment again shows that utilizing a gene selection algorithm to remove these irrelevant genes is a necessary step before conducting any analysis on gene expression profiles. This empirical study also reinforces our previous finding that variance-based gene selection method is an effective way to select informative genes.

Chapter 7 will show that selecting genes with high variance also is an effective way to reduce the amount of the BE confounding the gene expression analysis. The following section will show how we can use the variance of genes in each study and compare it with the variances of their corresponding genes in other studies to define a similarity measure between studies.

4.4 Variance of genes as an indicator of data set similarity

As we saw in the previous section, variance-based feature selection is an effective way to reduce the number of features in gene expression profiles. In this section we will be looking at relation of ranked genes based on their variance across two gene

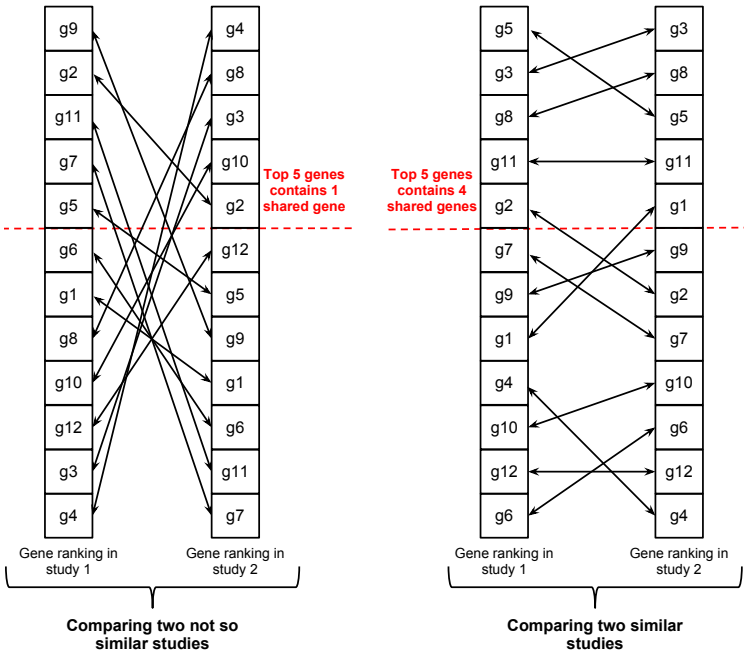


Figure 4.7: Schematically showing the relationship between two pairs of 12 ranked genes. The pair on the right show the greater similarity of ranking than the pair on the left. Genes are marked as g_1 to g_{12}. Arrows connect the ranking of same genes together for easier comparison. The red dotted line shows a cut-off threshold.

expression data sets. As we will see, the similarity of ranked genes across two studies is a strong indicator of their similarity in terms of the biological phenomenon under study. In order to illustrate the idea behind similarity of ranked lists, we show two schematic examples in Figure 4.7.

Each of the columns in this figure contains a ranked ordering of 12 genes based on their variance, marked as $g_1, g_2, \cdots, g_{12}$. On the right side we have a pair of similar ranked lists. For visual ease we connect the corresponding genes in the two ranked lists with arrows to one another. In general, if a gene is ranked as the i^{th} highest variance gene in one study, it is ranked as $(i \pm m)^{th}$ in the second study, where m is an integer that represents some kind of *margin of error*. When these two rankings are very similar, this m will be small. In the case of similar data sets (right pair), the maximum margin is $m = 3$.

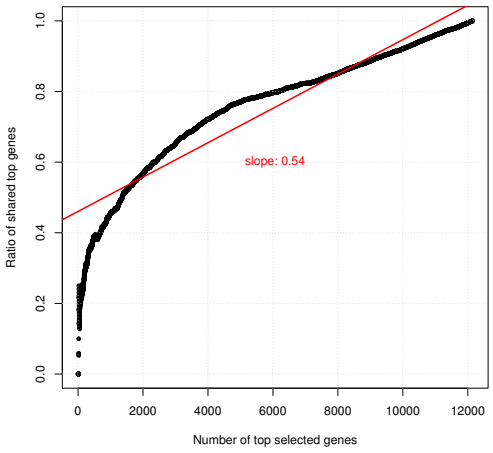
In comparison, on the left side of Figure 4.7 we are comparing a pair of two not-so-similar ranked lists. Here there is a lot of disagreement between the two rankings, *i.e.,* there are genes that have very high ranking on one column and very low ranking on the other column. This fact is visually observable by long arrows that connect

top of one column to the bottom on the other column. Here the maximum value of the margin is $m = 11$, meaning that the highest ranked gene in one study is ranked as the lowest in the other study (g_4).

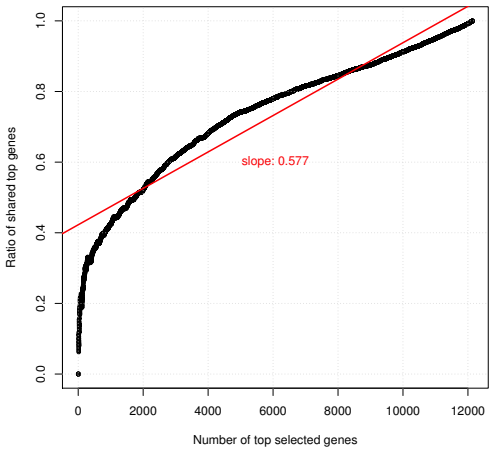
One way to quantize the similarity of two ranked list is to look at the number of shared elements at different cut-offs of ranked list. For example, in Figure 4.7 we draw a red dotted line to show a cut-off at 5. Now we look at the both ranked lists and count how many genes are shared between the two lists above this threshold. On the right side, where we have two similar lists, there are 4 genes shared by the two lists, so the *shared ratio* is $4/5 = 0.8$. This number for the not-so-similar pair of ranked lists on left is $1/5 = 0.2$. These two numbers indicate the similarity between these two pairs of ranked lists. Note that we can find the ratio of shared genes for different cut-off values $1, 2, \cdots, 12$; one shared ratio for each cut-off threshold.

This idea was used to show the relationship between four pairs of gene expression studies in Figure 4.8. The data sets used in this analysis are all conducted using the Affeymetrix U133A GeneChips and have 12092 gene measurements using the custom cdf file. In each plot, there are 12092 shared ratio values (shown by black dots) measured at different s thresholds $1, 2, \cdots, 12092$. Thus the dot with coordinates (2000, 0.6) corresponds to a situation that the top 2000 genes in two ranked gene lists have $0.6 = 1200$ common genes. This plot is know as a "correspondence at the top" plot (CAT) [90] and it was originally used to for comparing two procedures for detecting differentially expressed genes. Dots with y-values close to 1 indicate that the content of two lists are almost identical, *i.e.,* 100% common genes. On the other hand, dots with y-values close to 0 show that the two list are completely different and they contain 0% common genes. We fitted a line to all 12092 points and show its slope in each panel. The slope of this line is an indicator of relation between the paired top genes lists over all possible list lengths.

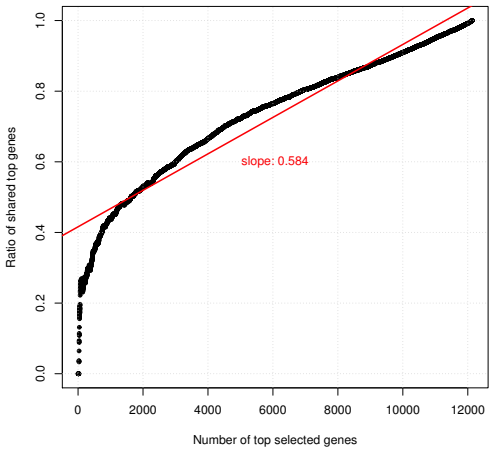
Figure 4.8 includes 4 panels: panel (d) shows the relationship between the variance ranking of two breast cancer studies while the other 3 panels, (a, b, c), show the relationship between pairs of unrelated cancer types, namely (a) ovarian cancer versus prostate cancer, (b) ovarian cancer versus lung cancer, and (c) prostate cancer versus lung cancer. Note, when we compare two similar data sets, panel (d), the shared ratio is close to 1 for all lengths of top genes. Ideally, when two data sets ranked lists agree completely with each other, the shared ratio will be 1 for all cut-off values, meaning that all the black dots will have 1 for their y-value. In this



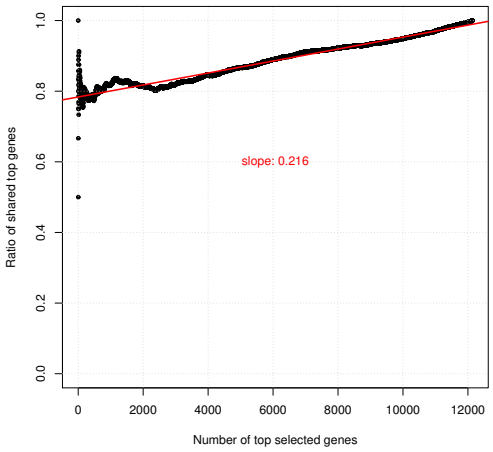
(a) Ovarian cancer ranking vs Prostate cancer ranking



(b) Ovarian cancer ranking vs Lung cancer ranking



(c) Prostate cancer ranking vs Lung cancer ranking



(d) Breast cancer1 ranking vs Breast cancer2 ranking

Figure 4.8: Correspondence at the top (CAT) plots: shared ratios plotted as a function of the number of top selected genes ranked by their variance. The red line depicts the result of linear regression fitted to all data points. The slope of the line is written on each plot.
(a)(b)(c) depict relation between biologically unrelated data sets containing different types of cancer specimens.
(d) depicts the relation between two biologically similar data sets (both containing breast cancer specimens).

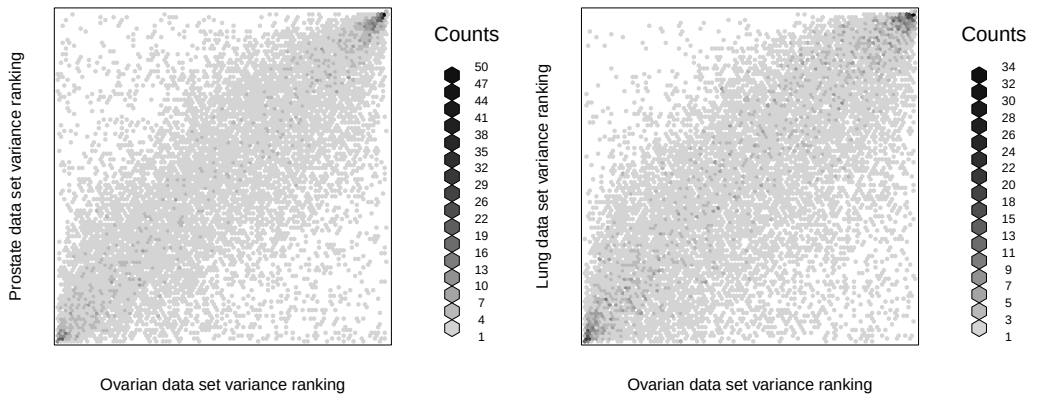
case the slope of the red line will be 0. Thus the closer the slope to 0 is, the more similar two data sets are.

In comparison, panels (a), (b), and (c) of Figure 4.8 show less agreement between the ranked genes. This is not unexpected since we are comparing pairs of different cancer types. In all three cases the shared ratios are below 0.8 for lists with lengths less than 6000. In contrast, in panel (d), almost all points have shared ratio more than 0.8. If two ranked lists are completely random and independent from each other, then we expect the dots will all lie on $y = x$ line, whose slope would be 1. This means that if the two lists do not agree with each other at all, then the slope of the fitted line will be 1.

To summarize the behavior of all shared ratio values by a single number, we use the slope of the line fitted to all 12092 points. Panel (d) has the slope of 0.216 while other panels(a, b, c) have larger slopes, 0.54, 0.577, and 0.584 respectively. The ideal best match between two ranked list will have score of 0, *i.e.,* the red line will be a horizontal line on the top of the plane ($y = 1$) meaning that the shared ratio is 1 independent of the number of selected genes. The shared ratio between two independent random ranked lists will lay on the $y = x$ line and thus its score (slope) will be 1. In the next section we will see how using this scoring system we are able to find distances between 36 gene expression studies and cluster them according to their similarity.

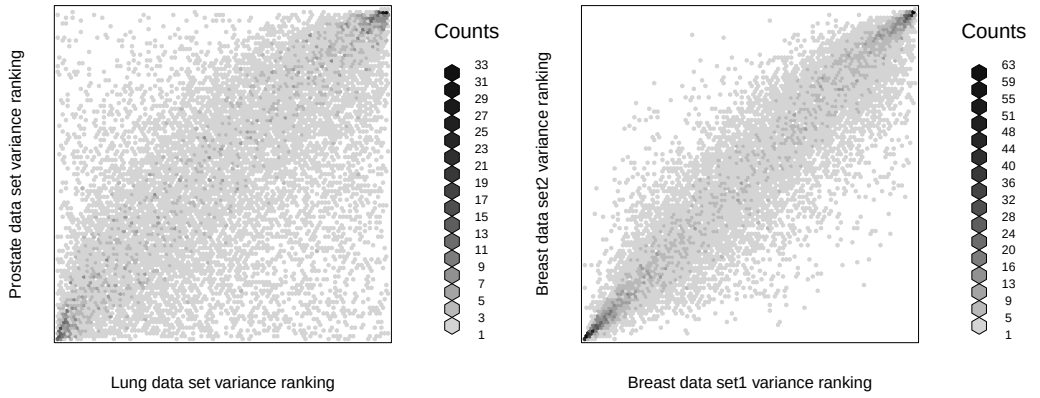
Another way to depict the similarity between two variance-based ranked lists of Figure 4.7 is to use a scatter plot. Figure 4.9 uses this technique to show the relationship between the 4 aforementioned pairs of gene expression studies. Each point in these scatter plots represent one gene, whose x-coordinate depicts its ranked variance in one data set and its y-coordinate depicts its ranking in the second study. As in any other scatter plot, the concentration of points on the main diagonal depicts the high concordance between the two ranked lists. In contrast, having data points randomly scattered all over the plane depicts no linear relationship between the values of x-axis and y-axis which means the two ranked lists do not agree with each other on many genes.

By comparing panel (d) in Figure 4.9 to the 3 other panels, one can see that the ranking of two breast cancer data sets are much more similar. In panel (d), most of the data points are close to main diagonal ($y = x$ line) meaning that most of the points have almost equal x and y coordinate values, *i.e.,* almost equal variance



(a) Ovarian cancer ranking vs Prostate cancer ranking

(b) Ovarian cancer ranking vs Lung cancer ranking



(c) Prostate cancer ranking vs Lung cancer ranking

(d) Breast cancer1 ranking vs Breast cancer2 ranking

Figure 4.9: Scatter plots comparing variance-based ranking of genes for paired data sets. Each point represents a gene whose x-coordinate is its ranked variance in one data set and its y-coordinate is its ranked variance in the other data set. Instead of depicting all 12092 dots (one for each gene) we use a binning technique whose grey level intensity represents the number of dots in each bin.
(a) (b) (c) Pairs of biologically different data sets, comparing different types of cancer.
(d) A pair of biologically similar data sets, both breast cancer.

based rankings of genes in two breast cancer studies. Ideally, when two data sets completely agree about the ranking of genes based on variance, then in their scatter plot all points land on the main diagonal line. In comparison, the other 3 panels of Figure 4.9, (a, b, c) have points scattered all over the plane. In these three panels there are many points close to upper-left and bottom-right corners, which means there are many genes that are ranked high in one data sets while ranked low in the other data set.

Comparing the four panels of Figure 4.9 to Figure 4.2 is also informative. In Figure 4.2 the length of (vertical and horizontal) bars are indicator of the difference between the *value of variance* of each gene in two data sets while in Figure 4.9, the distance of dots from $y = x$ is caused by the *rank of the variance* of genes in two data sets. The relationship is comparable to the relationship between the Pearson correlation and Spearman correlation. Nevertheless, both of these figures indicate the same relationship between the four paired data sets in their four panels. Pairing different cancer tissue types in panels (a) , (b) , and (c) causes a lot of disagreement between the variance of genes and their ranking. Figure 4.2 shows this by many long bars (subtracting the length of two hands of crosses in Figure 4.1 did not result in zero length bar) while Figure 4.9 shows the disagreement by many points scattered close to top-left and bottom-right corners. On the other hand, pairing two biologically related data sets (both breast cancer) results in high level of agreement between the variance of genes and their ranking, thus Figure 4.2 has fewer bars and the Figure 4.9 dots are mostly close to $y = x$.

To quantify the "degree of matching", we calculated the sum of the distance of each point to the $y = x$ line, which is the sum of squared differences between the ranks of the genes in the two sorted lists.

$$distance(X, Y) = \sum_{i=1}^{p} (rank_i^X - rank_i^Y)^2 \tag{4.4}$$

where $rank_i^X$ means the rank of i^{th} gene in data set X. This is a second method to summarize the difference between the ranking of gene's variances between two data sets. In the following section we show how we can use this metric to cluster a large set of gene expression data sets.

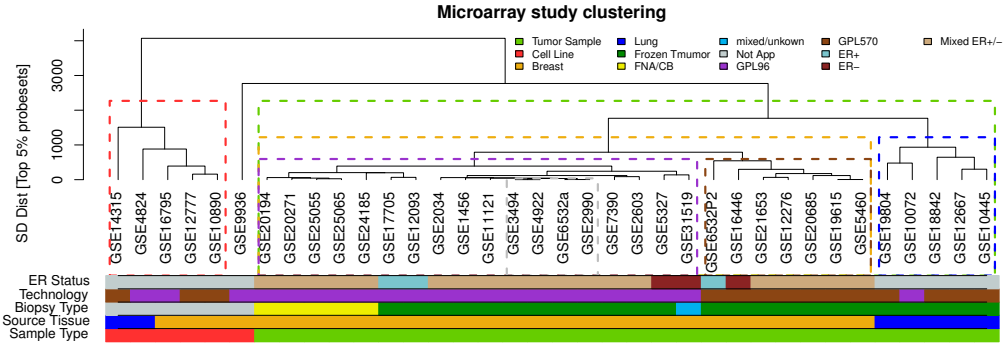


Figure 4.10: Clustering of 36 gene expression data sets (a total of 5744 gene expression profiles) using the variance-based ranking of genes within each data set.

4.5 Clustering gene expression data sets using their variance-based gene ranking

We used the two similarity measures defined in the previous section to cluster 36 gene expression studies representing almost 6000 breast and lung cancer profiles. All studies were conducted on Affymetrix GeneChips, either U133A or U133 plus 2.0. and were processed as explained in Section 4.2. Five of the data sets contain cell lines of breast cancer and lung cancer and the rest of the data sets contain tumor tissue. For some of the cancer studies, the specimen origin is known, either frozen tumor, fine needle aspiration, core biopsy, or a mixture. For some of the breast cancer studies the ER status is known, meaning that breast cancer studies can contain only ER+ patients, only ER- patients, or mixture of both.

We used the two metrics introduced in the previous section to find the pairwise distance between every pairs of data sets, *i.e.*, $\binom{36}{2} = 630$ distance scores were calculated. Then all these distances were fed into a hierarchical clustering algorithm with *average* agglomeration method. The result of the clustering was the same for both metrics and is shown in Figure 4.10.

The main clusters in the Figure 4.10 are highlighted using the triangles with colored dotted border lines. At the highest level we have the **tumor cluster** versus **cell line cluster** showing that according to the variance of genes, these two types of specimens are the most different ones. Within each of these two clusters, there are both breast cancer and lung cancer data sets. This means that the difference between cell line specimens and tumor specimens is more significant than difference between breast cancer and lung cancer.

Within the **tumor cluster**, there are two main clusters; One formed by **breast cancer** data sets and the other one contains **lung cancer** data sets. This is not surprising as data sets with the same tissue type are expected to be similar. The same applies to the cell line cluster where same cancer types are more closely clustered together.

The **breast cancer** cluster further divides into two main clusters that are formed by the technology used for profiling, namely **U133A** and **U133 plus 2.0.**. This shows the effect of chip type on the distribution of gene expression values in a data set. By further analyzing the relation within each of these two clusters, one can see that data sets with only ER+ or only ER- patients are clustered closely together. This is again showing that ranking of the gene's variances within data sets depends on the phenotype of instances in the data sets. Therefore if two data sets contain instances with similar phenotypic distribution then their variance-based ranked genes will be similar. As the last note, the **gray rectangle** containing 4 data sets in the middle of Figure 4.10 shows a group of breast cancer studies that cluster together very closely, indicated by branches with almost zero vertical length connecting them together. Further investigations showed these 4 data sets shared a large number of the same samples.

We would like to emphasize that the likely relationship between these 36 gene expression data sets were known a priori by considering their composition of patients and their phenotypic properties, *i.e.,* this clustering does not contain any new knowledge. What is interesting here is that we were able to extract these relations solely based on the variance of the genes in each data set. This experiment again reinforces our hypothesis that the variance of expression values of each gene is an important indicator of their importance for explaining the phenomenon under study. Also the similarity of the variances of genes across two data sets indicates the similarity of their underlying biological signals. Note that here the batch correction method was not performed as we were comparing the data sets independently without integrating their expression intensities.

Here we emphasize the exceptional ability of gene expression variances to correctly estimate the similarity of microarray data sets does not apply to other statistics of expression values, including their mean expression intensity. This is mainly because additive factors do not affect the variance of gene expression values. In other words, if we show the true expression of gene g of array i in data set j by x_{igj}

and we assume that being profiled in data set j causes an additive factor to the true expression value $x'_{igj} = x_{igj} + b_{gj}$, then under this model the variance of the true expression values and effected expression values will be equal:

$$\mathrm{Var}(x'_{igj}) = \mathrm{Var}(x_{igj} + b_{gj}) = \mathrm{Var}(x_{igj}) + \mathrm{Var}(b_{gj}) + 2\mathrm{Cov}(x_{igj} + b_{gj}) = \mathrm{Var}(x_{igj}) \tag{4.5}$$

This is why comparing the variance of genes across two data sets can accurately represent their similarity. Section 7.1 will later look at this similarity in more detail.

4.6 Summary

In this chapter we introduced and evaluated three novel feature selection algorithms that were applied to pairs of gene expression studies in order to select a feature set that truly represents the correlation across the two studies. Two of these feature selection methods use a simple algorithm to merge the ranked lists of the genes. In one algorithm, we rank genes (within each data set) based on their variance and in the other one, based on their expression intensity. The third algorithm uses a simple heuristic that maximizes the cross correlation scores between the two data sets. We compared our methods with integrative correlation feature selection method. We evaluated these methods with and without utilizing the detection call algorithm.

Given two (or more) gene expression data sets, these algorithms select a subset of genes that have different effects on the cross correlation matrix of the two data sets. We empirically studied the effect of performing each feature selection algorithm on the cross correlation matrices between different tissue/cancer types as well as similar cancer types.

The correlation increment algorithm is purposefully designed to select the genes that according to them arrays become highly correlated regardless of the tissue of origin, cancer type, or the individual whose specimens were profiled. Genes selected by this method fail to reflect the real relationship between gene expression profiles and are unable to capture the fine similarities between expression levels. We empirically showed there are some genes whose expression intensities are very similar across a wide range of different tissues and cancer types. Removing these genes is likely to substantially improve the performance of learning algorithms not only by reducing the gap between the number of features and the sample size but also by discarding misleading signals in those genes. We strongly suggest that

researchers remove these genes before performing any learning analysis on gene expression studies.

Our empirical results show that variance-based ranking feature selection chooses a subset of genes that reflects the expected correlation better than intensity based feature selection and integrative correlation algorithm. These genes were able to capture the fine correlation coefficient changes across data sets with different tissue origins. We also showed that filtering of genes using detection call algorithm, which is a standard filtering method applied to Affymetrix data sets, is not sufficient to remove all the irrelevant genes and utilizing more complicated gene selection algorithm still is needed. We briefly showed why variance of gene expression is a prosperous measure for selecting them as it is robust against additive factors that might affect gene expression values. We also believe that higher variance of a gene means it is biologically "active" in the population under study and thus keeping them is a biologically sensible action.

As an alternative way to support the idea of keeping high variance genes, we conducted a large scale experiment that includes 36 gene expression data sets (5744 gene expression profiles overall). In this experiment we used the similarity between the variance-based ranked gene lists of 36 data sets to cluster them. The result of clustering was highly in concordance with our prior knowledge about the data sets. This experiment reassured us that ranking genes based on their variance is an effective way to assess the importance of genes. Section 7.1 will later show that filtering genes based on their variance also can reduce the confounding role of BE on the biological signals.

Chapter 5

Batch Effect Detection

Gene expression experiments are conducted in order to study how gene expression relates to the underlying biological phenotype or clinical measures. When experiments are performed in different labs, or using different platforms, the biological signal might be confounded by technical factors, leading to so-called "batch effects" (BE). The main obstacle to pooling several gene expression data sets together and study their underlying biological phenomenon is the unwanted influence of technical factors on the expression values. BE distorts the signals of interest in a way that is hard to detect, mainly because it is mixed with the signal of interest, making it difficult to distinguish them from one another. In this chapter we introduce three methods that can be used for detecting the existence of BE. We will also show how these techniques can measure the effectiveness of BE removal methods.

The methods introduced in this chapter mainly serve two purposes:

- Given a gene expression study with known potential sources of BE, this chapter's methods will clarify whether there is any systematic bias due to the batch effects present in the study.
- Given a BE correction algorithm and a set of gene expression studies, this chapter's methods will evaluate the performance of the BE correction algorithm empirically, to allow us to compare various BE correction algorithms.

Chapter 7 compares the performance of our proposed BE correction algorithm (defined in Chapter 6) to several other available BE correction algorithms using this chapter's techniques.

5.1 Using specifically designed data sets to evaluate the performance of BE correction algorithms

In this section we look at gene expression data sets that are specially designed for batch effect analysis. The main characteristics of these data sets is that they include *at least two technical replicates* of some subjects. These technical replicates are profiled under some controlled conditions to study the effect of a specific factor, such as different platforms or different lab technicians. Since each technical replicate is measuring the same biological phenomenon under different technical perturbations, we expect them to have equal values. Therefore the dissimilarity between technical replicates is an indicator of batch effect caused by the technical factor under study. We use this simple key observation to utilize data sets with technical replicates to evaluate the performance of BE correction algorithms. Basically we compare the similarity between technical replicates before BE correction to the similarity after applying BE correction as a way to evaluate the performance of a particular BE correction method. We will measure the similarity between two gene expression profiles using the Pearson correlation coefficient between the two arrays. Since negative correlation shows an (inverse) linear relationship, we use the absolute value of correlation as the score of similarity, meaning that the similarity score will be in the range of $[0, +1]$.

To illustrate how we use the relationship between technical replicates, we use Figure 5.1, which schematically shows the similarity between two batches of nine technical replicates. As one can observe, the correlation between each subject and its technical replicate is very high (shown by almost-white squares). This figure also groups the nine subjects into three sets of biological replicates, marked as S1, S2, and S3. Each set contains a group of biologically very similar instances such as a few mice of a particular mouse strain. Biological replicates also have high similarity with each other since they share very similar biological signals. The similarity of biological replicates are shown with yellow squares, whose similarity values are less than the similarity of technical replicates, which are represented by white squares. Finally the red squares show the low similarity between unrelated subjects. This is expected as these subjects are not closely related to each other, and so we expect them to have lower similarity values.

Figure 5.1 shows the similarity matrix *across batches*. We can construct a similar

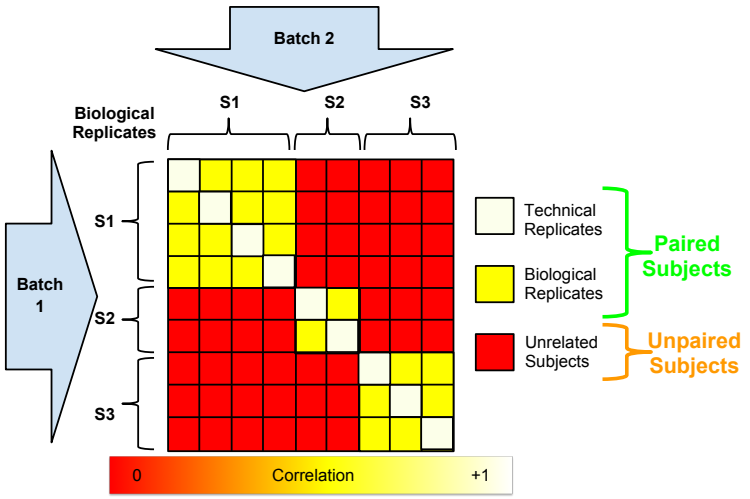


Figure 5.1: The schematic cross correlation between two batches of 9 technical replicated samples. Note that instances belong to 3 sets of biological replicates, annotated with S1, S2, and S3.

matrix *within batches* too. Ideally, in the absence of BE, the within-batch similarity matrices should be exactly equal to across-batch similarity matrix. This means if profiling gene expressions under different technical conditions, say in two different labs, does not have any effect on the measured values, then the technical replicated gene expression arrays will be exactly equal and thus the correlation between them will be exactly equal to 1.0 . The discrepancy between "across" and "within" similarity matrices is an indicator of the BE confounding role.

In order to summarize the discrepancy of within versus across similarity matrices with some numerical scores, we partition matrices into two sets, *paired* and *unpaired*. The paired set includes all correlation values between biological or technical replicates (white and yellow squares) while the unpaired set includes correlation values between unrelated subjects (red squares). When we extract these values from the cross correlation matrix, we call them **across-paired** and **across-unpaired** [1], respectively. Paired and unpaired notions can also be applied to similarity scores of gene expression profiles within batches. Thus we partition the within-correlation matrices of each batch in the same manner and extract two sets of scores, **within-paired** and **within-unpaired**. The relation between density of these 4 correlation sets leads us to a numerical score that indicates the severeness of the batch effect. In Figure 5.2 we show the schematic distributions of these 4 sets of correlation

[1] We will use this color coding for the illustrative purposes in the Figures 5.2 and 5.3 and also in the similar plots of Chapter 7

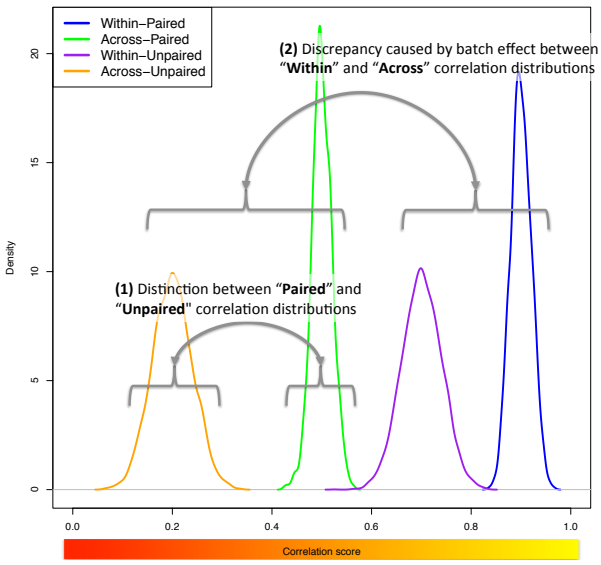
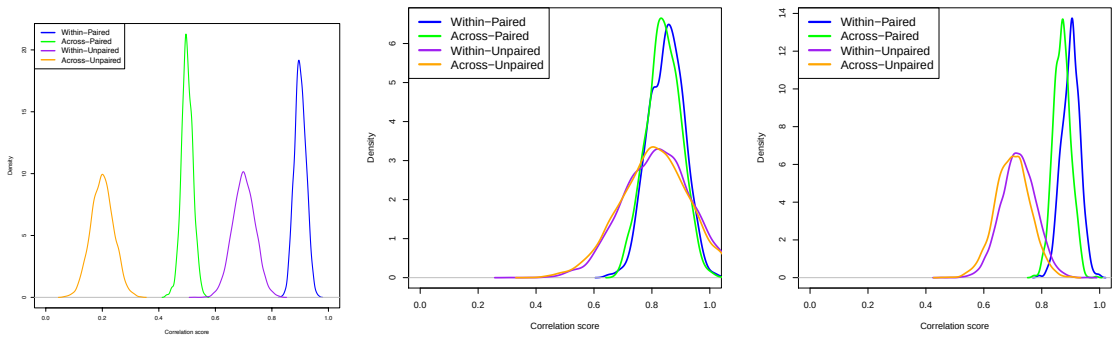


Figure 5.2: Schematic distribution of the correlation coefficient of paired and unpaired sets within and across batches.

coefficients.

The most important observations about Figure 5.2 are the two highlighted gaps between the distributions. One is the distinction between "paired" and "unpaired" curves, which exist in both across and within sets. This gap is caused by the biological signals. Replicated subjects are biologically related and thus have higher similarity scores to each other. However, unrelated subjects are not as biologically related and thus we anticipate they will have lower similarity scores. This gap is desirable and BE correction methods should preserve this distinction. The other gap is the discrepancy between "across" and "within" curves. This gap is caused by BE. For example, let's compare the within-paired versus the across-paired curves. These two curves include the correlation values between the same pairs of gene expression arrays, with only one difference: The across-curve contains the correlation coefficient measured for replicated subjects that were profiled in two different batches. The within-curve contains the correlation between the same set of subjects that were profiled in the same batch. Ideally, in the absence of BE, these two curve should be exactly the same. However, BE confounds gene expression values in a way that makes the technical replicates look less alike each other. A good BE correction method should shifts across distributions towards the within ones.

Figure 5.3 schematically shows how a poor performing (b) and good performing (c) BE correction method changes the relation between the original distributions



(a) Original (b) Bad performance (c) Good performance

Figure 5.3: The outcome of utilizing BE correction methods applied to the original distributions (a).
(b) Poor performance: correction causes mixing "paired" and "unpaired" curves.
(c) Good performance: correction shifts "across" curves towards "within" curves.

(a). On the middle panel we can see that applying poor BE correction shifts the "across" curves towards the "within" curves. It also causes the "paired" and "unpaired" curves to intermix and the clear distinction between them, as shown clearly in Figure 5.2, disappears. This shows that BE correction not only removes the differences caused by BE (shifting the "across" towards "within") but also it removes the biological signals of interest (mixing "paired" and "unpaired"). This situation usually is observed for BE correction methods that aggressively change batches to look alike. We observed this behavior for the DWD method in our experiments.

Figure 5.3(c) shows the ideal behavior of a BE correction method. In this case, the "across" curves are shifted towards "within" curves without mixing "paired" and "unpaired" distributions. BE correction should only remove the differences caused by BE while preserving the biological signals in data. In other words, after removing the BE the correlation between gene expression arrays was very similar to the correlation between them and their technical replicates.

This evaluation method cannot distinguish between a powerful BE correction method that truly removes the confounding factor from two batches and a naive one that simply replace one batch with the other one. In both cases, this evaluation method will indicate a high performance while the naive method technically deletes one of the batches.

We searched the gene expression omnibus (GEO) [21] repository to find data sets with technical replicates and we found two suitable cases. One is GSE17700 [121], a

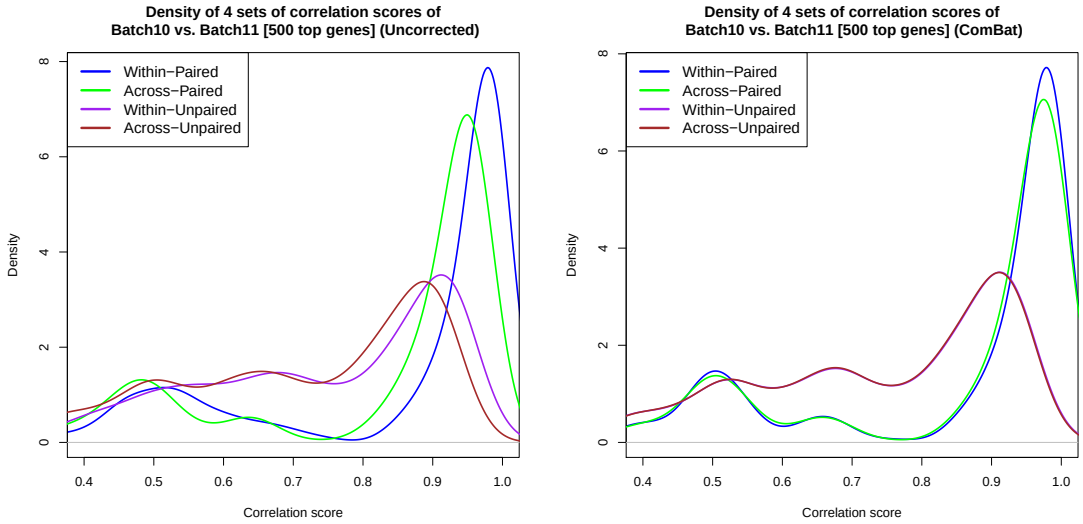
breast cancer study where 16 breast tumor specimens were sent to two labs to be profiled on two Affymetrix gene expression Chips, U133A and U133 plus 2.0 GeneChips. As we explained the details of preprocessing method applied to Affymetrix data sets in Section 4.1, the resulting data sets contain 12092 genes (probesets). This means we have four technical replicates of 16 subjects. For the sample preparation and profiling, the standard protocols proposed by Affymetrix were precisely followed. The goal of the study is to find how reproducible the gene expression profiling can be when there are different labs and different platforms involved. We refer to this date set as *breast cancer* study and represent its 4 batches as *Lab1-96*, *Lab1-570*, *Lab2-96*, and *Lab2-570*.

The second data set with similar properties is GSE38822 [122], which includes 48 mouse brain specimens divided into 2 strains $\times$ 2 drug treatments $\times$ 3 brain regions $\times$ 4 mice. As mentioned in the original study [122], the drug treatment did not have any effect on the gene expression values. Thus they consider their design as 2 strains $\times$ 3 brain regions $\times$ 8 mice. Since the 8 mice are very similar to one another, we can consider them as biological replicates. These 48 specimens were profiled on two different Affymetrix mouse arrays, Mouse Gene 1.0 and 1.1, to produce two technical replicates for each of the 48 mouse brain specimens. We will refer to this study as *mouse* study and its two batches as *batch-10* and *batch-11*.

We used this study to reproduce a real example of Figure 5.2 and plotted the distributions of correlation scores in Figure 5.4(a). Before calculating the correlations between the profiles of GSE33822, we used the variance-based feature ranking (see Section 4.2.2) to reduce the number of genes to 500 [2]. As expected, in this figure the within-batch curves are higher than across-batch curves. This is a clear indication of batch effect influence, as the correlations between the same set of samples is smaller under the influence of technical factors. In Figure 5.4(b) we plotted the same curves after applying ComBat to correct the batch effect. ComBat successfully shifts the within-batch and across-batch curves toward each other and removes the negative influence of technical factors on the correlation scores.

The following sections will use these two studies to explain the ideas of unsupervised and supervised BE detection.

[2] Without performing this gene selection step, because of "problematic genes" (see Section 4.2.3), all correlation scores are very close to 1.0 without any clear distinction between the curves.



(a) Uncorrected data

(b) ComBat corrected data

Figure 5.4: Distribution of four correlation score sets for mouse study (GSE33822). **(a)** Under the influence of the batch effect, the across-batch curves are lower than within-batch curves, *i.e.,* Across-Paired vs. Within-Paired and Across-Unpaird vs. Within-Unpaired. **(b)** An effective batch correction algorithm can shift the within-batch curves towards the across-batch curves. Note, Across-Unpaird and Within-Unpaired curves essentially overlap after correction.

5.2 Unsupervised methods for detecting BE

In this group of methods, we look for evidence of batch effects without knowing anything about the subjects in the batches. This differs from supervised methods, which require that subjects are *annotated*, perhaps as case versus control. In unsupervised methods, clustering, or some other unsupervised separation techniques such as PCA or MDS, is used to group subjects, after which we can look at the relation between the groups found by unsupervised methods and the batches. If the formed groups are highly related to the batches, then one can assume that the prominent signal in the data is the batch effect that is dominating any other more subtle biological effects. In this section we will explain two types of unsupervised methods: one based on k-means clustering and the other one based on hierarchical clustering.

Any method of clustering requires a similarity measure to find the distance between data points that are being clustered, which in this case are gene expression profiles. For this purpose we use the Pearson correlation coefficient, defining the

similarity between two profiles as 1 minus the absolute value of their Pearson correlation, leading to values in the range of $[0, 1]$. When the correlation coefficient between two profiles is 0, then their distance will have its maximum value, 1. On the other hand, when the correlation between two profiles is 1.0, meaning that they completely agree with each other, then their distance will be minimum 0.

Assume we are analyzing a data set containing d batches. In order to determine whether this data is influenced by batch effects, we apply a clustering method to group profiles. For this purpose we use a clustering method that, unlike hierarchical clustering, assigns each instance *exclusively* to one of the *mutually disjoint* clusters, *i.e.,* clusters form a *partition* for the set of instances. These clustering methods require setting a parameter for the number of clusters a priori. We use the number of batches, d, for this parameter. After clustering the data, we measure the concordance between the formed clusters and the batches, where a high concordance shows the prominent effect of a batch effect signal. We considered two ways to measure the concordance: *Corrected Rand Index* (CRI) and *Variation of Information* (VI) index. Comparing the values of these scores before applying some specific BE correction method and after, indicates the quality of this BE correction method.

CRI is a modified version of the rand index that is used to find the similarity between the two different partitionings of n subjects. Imagine we have a k class labeling for n subjects and we use some clustering algorithm to group them into k groups. The random index (RI) is related to the number of pairs of subjects that were in the same group under both partitionings (n_{11}) and the number of pairs of subjects that were in different groups under both partitionings (n_{00}).

$$RI = \frac{n_{11} + n_{00}}{C_2^n} = \frac{n_{11} + n_{00}}{n(n-1)/2}$$

The RI score is supposed to be close to 0 for random assignment of subjects to groups. However, when the number of groups is in the same order of magnitude as the number of subjects, this will not happen. To correct this problem CRI was proposed [123]

$$CRI = \frac{RI - \mathrm{E}[RI]}{max(RI) - \mathrm{E}[RI]}$$

whose value is in $[0, 1]$ and is 0 when RI is equal to its expected value [124].

Variation of information [125] is closely related to the mutual information concept. Again if we assume two partitionings of n subjects into k sets $A = \{A_1, A_2, \cdots, A_k\}$

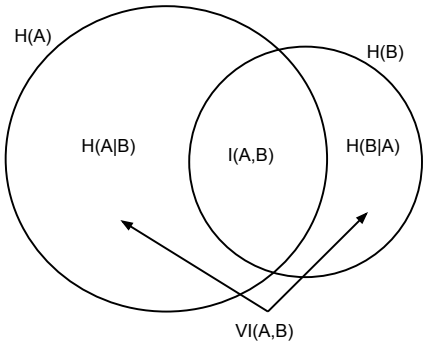


Figure 5.5: The variation of information (VI) depicted as the sum of two areas in the Venn diagram that show the relationship between the two partitionings A and B.

and $B = \{B_1, B_2, \cdots, B_k\}$, then the VI index will be

$$VI(A, B) = H(A) + H(B) - 2I(A, B)$$

where $H(\cdot)$ is the entropy of a partition and $I(\cdot, \cdot)$ is the mutual information between two partitionings. Figure 5.5 explains the VI index using a Venn diagram that shows the relation between two partitionings.

The lower bound of VI is zero, which happens when the two partitionings completely agree with each other. Its upper bound is limited to [125]

$$VI(A, B) \leq min(log(n), 2log(k))$$

The clustering analysis of BE detection is applicable to any gene expression study with known batches. However, for illustrative purposes we study its performance using the two specially designed studies that we mentioned earlier in Section 5.1. In Figure 5.6 we show the result of clustering on the two aforementioned gene expression studies. On the left panel the breast cancer study is grouped into 4 groups and on the right the mouse study is grouped into 2 groups. As one can see, in both cases there is a strong BE influence on the clustering results, *i.e.,* each cluster is formed almost only by the instances of one of the batches. This relationship perfectly exists for the mouse study and therefore both of the CRI and VI scores have their extreme value, which indicates the complete agreement between the clustering and batch labels. Later in Chapter 7 we will see how different BE correction methods changes these scores. For visual comparison we plot gene expression profiles of each study in the 2 dimensional space spanned by the top two principal components in Figure 5.7 (top

	Lab1-96	Lab2-96	Lab1-570	Lab2-570	Sum
Cluster1	14	2	0	0	16
Cluster2	2	14	0	0	16
Cluster3	0	0	13	2	15
Cluster4	0	0	3	14	17

(a) Breast cancer study. (CRI=0.663, VI=0.807).

	Batch-10	Batch-11	Sum
Cluster1	48	0	48
Cluster2	0	48	48

(b) Mouse study (CRI=1, VI=0).

Figure 5.6: Clustering of the two gene expression studies by the partitioning around medoids (PAM) algorithm [126, 127] using correlation-based distance measure. PAM requires the number of clusters a priori, for which we used the number of batches.

row). In the bottom row of Figure 5.7 we show the result of hierarchical clustering of gene expression profiles of each study.

The influence of BE on the mouse study is very obvious in both (b) and (d) panels of Figure 5.7. Members of each batch form a separate "cloud" in principal component plot of Figure 5.7(b). Also in the hierarchical clustering plot (Figure 5.7(d)), members of each batch are clustered together with short vertical branches, indicating small distances between them. As opposed to these short branches, the whole cluster of batch-10 is connected to the cluster of batch-11 with a relatively longer vertical branch, which indicates the significant distance between the two groups. From both of these plots, one can safely assume that mouse study data is confounded by BE due to the platform on which the samples were processed.

However, in the case of breast cancer study, the results are more subtle. The principal component plot (Figure 5.7(a)) shows two main "clouds", one for GPL96 (which includes data from both labs) and one for GPL570. It seems that the chip type dominates as the major batch effect over the site at which the samples were processed. The hierarchical clustering results also show that patients with the same chip type are paired together (the interchanging color pattern in the two main clusters of panel (c)) with relatively short branches, indicating small distance between them. However the GPL96 and GPL570 form two clusters that are connected with relatively longer branches. Later in Chapter 7 we will see that, in the absence of BE, all 4 technical replicates are connected with each other by shorter branches, which indicates the very small distance between them.

We also should note there is BE related to the lab for the samples of the same chip type. However, since this BE is less severe than BE related to chip type, it is not clearly observable in the panel (a) of Figure 5.7. In order to see it more clearly, we plot the samples of each chip type separately in Figure 5.8. As one can

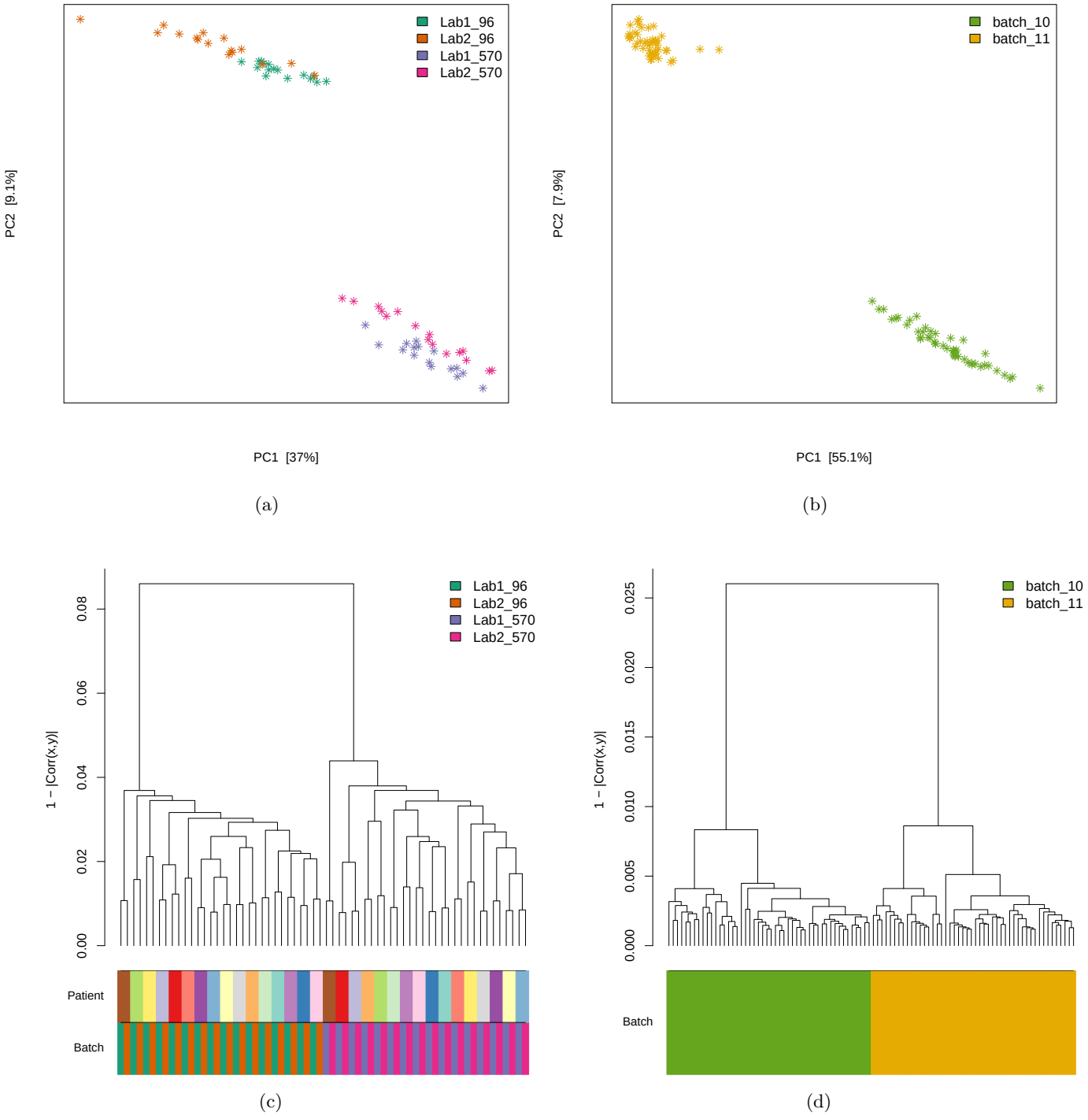
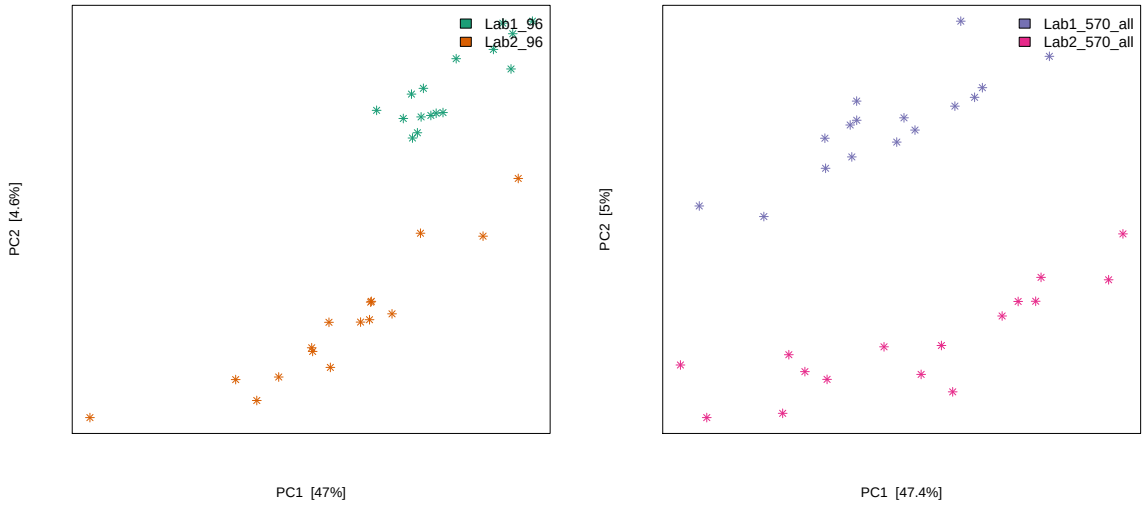


Figure 5.7: Visual analysis of gene expression profiles of the breast cancer study (left column) and the mouse study (right column).
The top row plots each gene expression profile as a dot in the 2 dimensional spaces of the top 2 principal components.
The bottow row shows the result of hierarchical clustering using the correlation-based distance measure.



(a) (b)

Figure 5.8: Principal Component plots for breast cancer study grouped by the chip type.
(a) GPL96 samples, CRI=0.548 VI=0.754
(b) GPL570 samples, CRI=0.758 VI=0.468

see, in this case there is a clear distinction between lab1 and lab2 samples in both plots. This suggests that the lab related BE is partially responsible for the observed variability along the first and second principal components. Later in Chapter 7 we will see that BE correction methods can effectively correct both chip related BE and lab related BE in the breast cancer study.

5.3 Supervised methods for detecting BE

In this section, we study methods of BE detection that need genuine biological differences between instances that are known a priori via some class labeling for instances, and therefore are called supervised. The class labeling can be any biological variable of interest. In the case of cancer patients, for example, we might be dealing with relapse/no-relapse or long/medium/short survival, or subtype labeling such as ER+/ER- for breast tumors, or patient sub-groups that received different adjuvant therapies. We refer to this labeling as *treatment factor* and assume it is binary (*e.g.,* case versus control). However, it can have more than two levels and all the methods in this section will be applicable to such non-binary treatment fac-

tors. The main idea in this category is to apply some sort of supervised learning algorithm (classification, regression, or significant analysis of features) *within the batches* and then extend the learned model *across batches*. Significant differences between within-batch results and across-batch results is an indicator of BE.

First we look at the method based on significant analysis. The main purpose of significant analysis is to find the set of genes that are differentially expressed between two treatment conditions. Most of the methods used for this purpose are some kind of statistical hypothesis test that corrects for multiple testing by using a mechanism to control the false discovery rate (FDR). By using one of these methods, one can find the set of genes with significantly different expression values between the case group and the control group within each batch. We call the union of these sets for all batches the *"within" significant genes*. Next we use the same algorithm to compare case sample in one batch to the control group in the other batch and vice versa to find the *"across" significant genes*. Figure 5.9 shows this idea schematically. Note this idea was first proposed by Sims et al. [86] and later Parker and Leek [128] combined it with cross-validation schema to measure the influence of batch effects on biological labeling. We extended the Parker and Leek [128] ideas by considering all possible combinations of batch and case/control labeling. If there are more than two phenotypic classes in the data sets, we can repeat this process for every pair of phenotype comparisons.

The relationship between "across" significant genes and "within" significant genes is an indicator of BE. In the absence of BE, these two situations should exhibit very similar behaviors, showing that the relation between the case and control group is independent of their originating batch; if this is the case, then within-batch and across-batch significant gene sets will have a large overlap with each other. However, in the presence of BE, there is always a big discrepancy between these two sets and the across-batch set includes many more apparent significant genes. This is because the biological signals are confounded with the BE signal. Thus we observe many more significant genes. The other way to examine the existence of a batch effect is by analyzing the across-batch false positive genes. As shown in Figure 5.9, false positive genes are found by comparing case (control) instances across batches. Comparing the two sets of case (control) instances across batches should essentially result in no significant gene as we essentially are comparing two random samples of a same population. However, in the presence of a batch effect, there will be

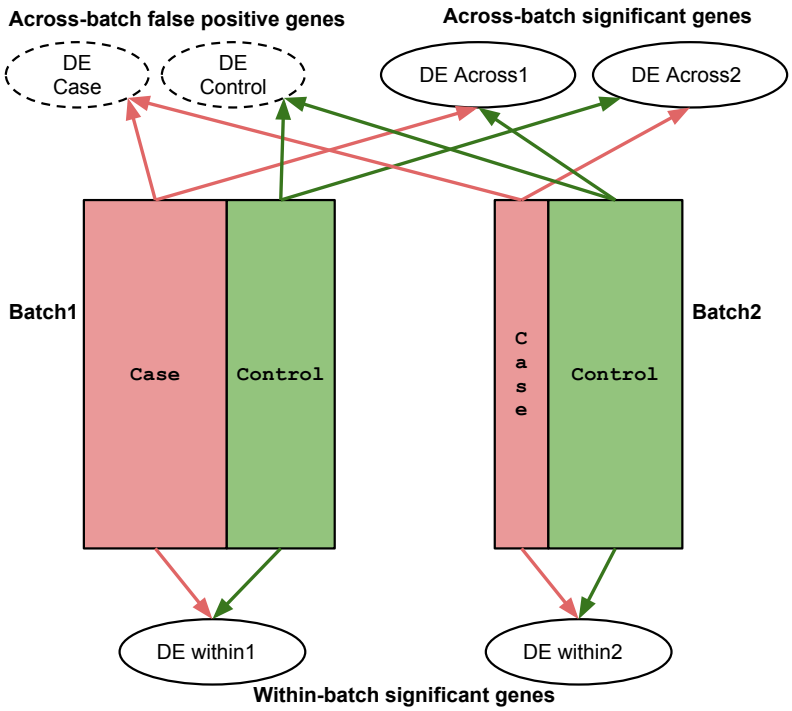
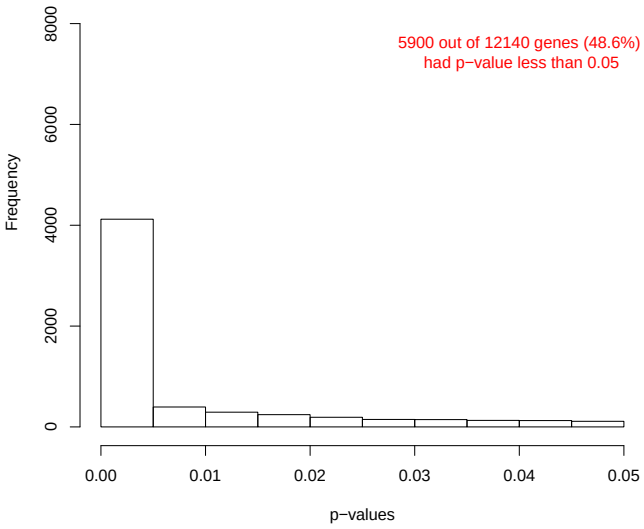


Figure 5.9: Finding the set of significant differently expressed genes between case and control groups, *within-batch* (bottom), *across-batch* (top-right), and *across-batch false positives* (top-left).

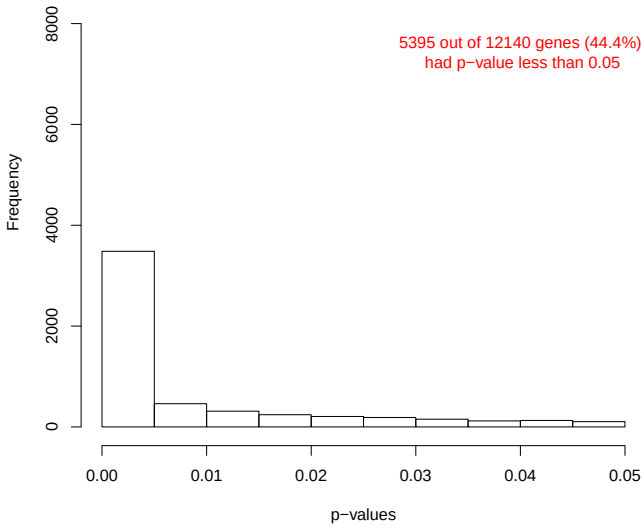
many significant genes when we compare the same treatment conditions. In order to depict this phenomenon and demonstrate the relationship between these sets of significant genes we use the following example.

Two breast cancer studies: GSE2034 [110] and GSE7390 [111] (these two data sets were introduced in Section 4.1 as Breast cancer1 and Breast cancer2) are analyzed here to find genes that are differentially expressed between the two subgroups of patients, ER+ versus ER-, identified by student t-test that is used to calculate the significance of each gene individually. Figure 5.10 illustrates the p-values density of genes that have a p-value less than 0.05. As one can see, in the case of within batch sets (top row), fewer than 50% of the genes are declared significant, which is substantially less than the across batch sets (bottom row), which includes more than 80% of genes. An effective BE correction method will make the bottom plots look like the top plots.

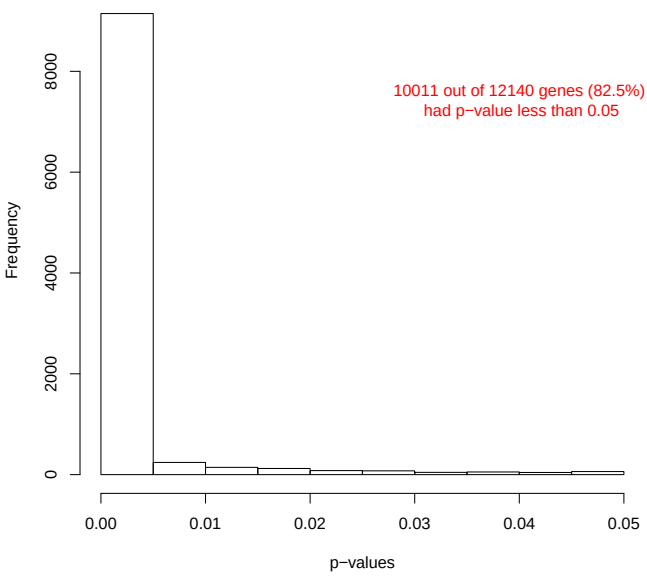
Figure 5.11 summarizes the number of significant genes found by the 3 different types of comparisons shown in Figure 5.9, *i.e.,* within-batch, across-batch, and false positives, for the same two breast cancer data sets. The numbers of the left



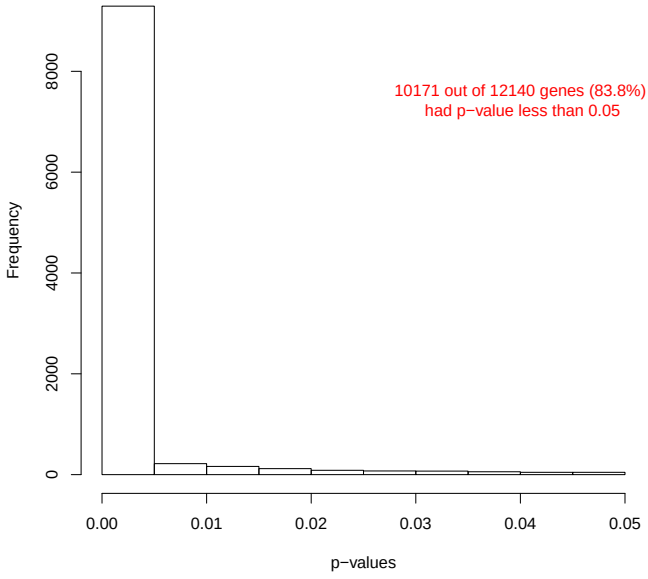
(a) ER+ GSE2034 vs. ER- GSE2034



(b) ER+ GSE7390 vs. ER- GSE7390



(c) ER+ GSE2034 vs. ER- GSE7390



(d) ER+ GSE7390 vs. ER- GSE2034

Figure 5.10: Differentially expressed genes between the ER+ and ER- paitients.
The top row shows within-batch p-value distributions (both case (ER+) and control (ER-) samples belong to the same batch).
The bottom row shows across-batch p-values (case and control samples belong to different batches).

		GSE2034		GSE7390	
		ER+	ER-	ER+	ER-
GSE2034	ER+	-	3323	11640	11630
	ER-	.	-	11532	11847
GSE7390	ER+	.	.	-	2644
	ER-	.	.	.	-
Shared Across		11191 (93.48%) out of 11971			
Shared Within		1974 (49.41%) out of 3995			
Shared All		1585 (13.23%) out of 11982			

(a) Uncorrected data

		GSE2034		GSE7390	
		ER+	ER-	ER+	ER-
GSE2034	ER+	-	3323	3	3088
	ER-	.	-	2534	0
GSE7390	ER+	.	.	-	2645
	ER-	.	.	.	-
Shared Across		2293 (68.84%) out of 3331			
Shared Within		1974 (49.4%) out of 3996			
Shared All		1947 (48.25%) out of 4035			

(b) Corrected data (ComBat)

Figure 5.11: The number of significant genes (out of 12092 genes) at 0.001 significant level fount by comparing ER+ and ER- instances of two breast cancer studies: GSE2034 and GSE7390.
Highlighted cells contain the within-batch numbers. As table is symmetric, redundant values are avoided by putting **.** in some cells. The main diagonal cells are filled with **-** indicating there is no significant genes associated with them.

table (a) indicate the existence of batch effect: while the number of within-batch significant genes are almost the same in each batch (3323 and 2644) and they share 49.41% of their genes, the across-batch sets contain significantly larger number of significant genes (11630 and 11532) which is clearly due to the batch effect. As a result, the within-batch and across-batch significant genes share only 13.23% of their contents. This is evidence of batch effect confounding the gene expression values, which increases the number of "across-genes" significant genes. Also note that the number of false positive significant genes are almost as numerous as the across batch ones. This shows that the significant genes discovered across batches are influenced by the batch effect in addition to the biological signal, here, ER+ breast cancer instances comparing to ER- ones.

To make the comparison easier, we also included the batch corrected results in Figure 5.11 (b). In this case, the number of within-batch significant genes is very close to the number of across-batch significant genes and these two sets include a larger numbers of common genes. Also the number of false positive genes is almost equal to zero, which is much smaller than the number of false positives in the table (a), uncorrected data. In Chapter 7 we will use tables similar to Table 5.11 in order to demonstrate the effect of different batch correction algorithms on the set of significant genes.

The second supervised way to detect the existence of BE is to use a classifier. Here, in a manner similar to Figure 5.9, we collect the case and control samples once within the batches and once across the batches. If we use exact similar settings for

		GSE2034		GSE7093	
		ER+	ER−	ER+	ER−
GSE2034	ER+	-	0.80	**1.0**	1.0
	ER−		-	0.99	**1.0**
GSE7093	ER+			-	0.74
	ER−				-

(a) KNN (k=3) classifier.

		GSE2034		GSE7093	
		ER+	ER−	ER+	ER−
GSE2034	ER+	-	0.89	**1.0**	1.0
	ER−		-	1.0	**1.0**
GSE7093	ER+			-	0.87
	ER−				-

(b) Linear SVN classifier.

Figure 5.12: Leave one out classification accuracy of ER status prediction. Comparing within-batch performance (shaded area) to across-batch perfromance shows a significant difference between them which indicates the existance of BE.
The **highlighted values** are related to the classification of sample of the same biological group. They possess no biological distinction and only BE makes them distinguishable. Note we did not repeat numbers in the lower half of symmetric tables for clarity.

the classifier that separates case and control instances then its performance should be very similar for both situations. In other words, regardless of the origin of the case and control instances, the power to separate them should be the same assuming there is no technical bias caused by BE. However, in the presence of BE, distinguishing between case and control instances that belong to different batches is much easier, as here we are classifying batch1 versus batch2 rather than case versus control.

In order to quantify the difference between within-batch and across-batch classification performance, we measure the performance of classifiers with standard methods such as 10-fold or leave-one-out cross-validation [19]. We use the same learner with the exact same parameter settings for both within-batch and across-batch cases to make sure that their performance differences are based on only BE. The result of such an analysis is shown in Figure 5.12 for two different types of classification algorithms. The left panel shows the k nearest neighbors results with $k = 3$ and the right panel shows the linear SVM results. As one can observe, both classifiers separate the case and control samples across-batches perfectly (except for KNN which has a 0.99 accuracy for one of the across-batch combinations) while they have a lower performance when dealing with within-batch samples. This is strong evidence of the confounding role of BE. In fact, if we classify instances of the same biological group across-batches, for example ER+ instances, we still get perfect prediction accuracy. Those values are **highlighted** in both tables. The fact that classifiers can clearly distinguish between the samples of one batch from another, even though they belong to the same biological group, is an indicator of the presence of a strong batch effect.

The results of both supervised methods indicates that these two data sets have technical bias caused by a batch effect. This means that before combining them together, one should first apply a BE correction method to them. These two supervised techniques can be applied to any other group of data sets that are intended to be combined and they will decide whether or not these data sets have technical bias and so should be corrected for BE before combining them. The only condition is that all data sets in the group are required to have at least one same biological annotation for their instances. This annotation could be any biologically meaningful feature such as healthy tissue versus tumor tissue, clinical outcome such as short survival versus long survival, and tumor subtypes.

5.4 Summary

This chapter introduces two types of techniques that can systematically detect the existence of BE, namely *supervised BE detection* and *unsupervised BE detection*. Before combining a set of gene expression data sets, one can utilize these two proposed methods to determine whether the biological signal is confounded with BE. Supervised BE detection methods require all data sets include at least one biological annotation for their instances.

In addition to verifying the possibility of combining gene expression data sets, these two types of techniques can evaluate the performance of BE correction algorithms. For this purpose, one needs a group of gene expression data sets that are known to have technical bias caused by BE. Using the aforementioned techniques, we can estimate the severeness of BE before and after applying a particular BE correction algorithm. The difference between the two estimations is an indicator of the performance of applied BE correction algorithm.

This chapter also introduces another method for evaluating the performance of BE correction methods. This method is taking advantage of specially designed gene expression data sets that include some technical replicates, that are profiled under different conditions, such as in different labs, different chips, or different platforms. Because of the special design of these data sets and the relationship among technical replicates, we expect their correlation matrix follows a particular pattern. However, because of the BE, the correlation matrix is not exactly what we expect. By applying a particular BE correction algorithm to these specially designed data sets and measuring how the corrected correlation matrix follows the expected

pattern we can estimate the performance of the BE correction algorithms.

Finally this chapter briefly talks about the problems caused by poor experimental design that is fairly common in the microarray community. We explained why using this design confounds the biological signals with the BE and consequently makes BE correction impossible.

Chapter 6

Batch Effect Correction Using Canonical Correlation Analysis

As we have seen so far, large p small n is a major challenge in gene expression analysis. One possible solution – combining available data sets to increase n – is constrained by the batch effect, *i.e.,* the confounding influence of technical factors on the biological signal in gene expression studies. Therefore, removing this confounding component from the biological signal of interest is needed before we are able to combine gene expression studies together.

In this chapter we introduce a novel batch correction method, which is based on *canonical correlation analysis (CCA)* [5, 7, 129]. For this purpose, we took advantage of *transposablility* of gene expression data [130] and applied CCA in a non-standard way, *i.e.,* unlike the usual applications of CCA, in bioinformatics or elsewhere [131–136], where the sample/subjects are matched across two views, we instead match the feature (here genes) across the two views. We call this method *BECCA*. BECCA uses CCA to extract the common signal across two data sets and separate it from the confounding component in each data sets.

As we will see in CCA formulations, its performance mainly depends on the accurate estimations of covariance matrices between gene expression instances in batches and also their cross-covariance values. Therefore accurate estimation of covariance matrices is essential for the BECCA algorithm. We use the ideas of Chapter 4 to reduce the number of features and consequently have a better estimations of these matrices. Conducting this feature selection step is especially necessary in the high dimensional space of noisy gene expression measurements.

Section 6.1 introduces the notation used in the rest of the chapter. Section 6.2 continues with a brief introduction of direct CCA, the usual way of utilizing CCA,

and some of the CCA fundamental formulations that are essential to understand the mechanism of BECCA. See Appendix for a full review of CCA formulations and different ways to implement it and a discussion of different properties and variant formulations. Section 6.3 uses these formulations to demonstrate the BECCA mechanism and its special way of removing BE from the batches. This section introduces non-standard CCA, which is our modification of CCA for performing BE correction. We will compare the performance of BECCA algorithms against other BE correction algorithms using some empirical studies in Chapter 7.

6.1 Formulation Assumptions

We assume data matrices (batches) contain observations in the rows and features in the columns. So $\mathbf{X}_{1\ [n\times p_1]}$ includes observations vectors of $\mathbf{x}_{1i} \in \mathbb{R}^{p_1}$ for $i = 1 \cdots n$. We can show data matrices using their rows

$$\mathbf{X}_1 = [\mathbf{x}_{11}, \cdots, \mathbf{x}_{1n}]^t$$
$$\mathbf{X}_2 = [\mathbf{x}_{21}, \cdots, \mathbf{x}_{2n}]^t$$

CCA analysis typically assumes $\mathbf{X}_{1\ [n\times p_1]}$ and $\mathbf{X}_{2\ [n\times p_2]}$ are two views of n same objects, expressed using p_1 and p_2 features respectively. We show the column-wise merging of two views by $\mathbf{X} = [\mathbf{X}_1\ \mathbf{X}_2]$ which is a $n \times (p_1 + p_2)$ data matrix. We also assume each of these data matrices is mean centred, *i.e.,* the mean value of each feature is 0 (so the sum is 0 too):

$$\mathbf{X}_1^t \mathbf{1} = \mathbf{0} \tag{6.1}$$
$$\mathbf{X}_2^t \mathbf{1} = \mathbf{0} \tag{6.2}$$

where $\mathbf{1}$ is a column vector of all 1's. This assumption makes the *scatter matrices* $\mathbf{S} = \mathbf{X}^t\mathbf{X}$ proportional to covariance matrices $\mathbf{\Sigma}$

$$\begin{aligned}\mathbf{\Sigma}_{11} = \mathrm{Cov}(\mathbf{X}_1, \mathbf{X}_1) &= \mathrm{E}[(\mathbf{X}_1 - \mathrm{E}[\mathbf{X}_1])^t(\mathbf{X}_1 - \mathrm{E}[\mathbf{X}_1])] \\ &= \mathrm{E}[\mathbf{X}_1^t\mathbf{X}_1] \\ &= \frac{1}{n}\mathbf{X}_1^t\mathbf{X}_1 = \frac{1}{n}\mathbf{S}_{11}\end{aligned}$$

and similarly

$$\mathbf{\Sigma}_{22} = \mathrm{Cov}(\mathbf{X}_2, \mathbf{X}_2) = \frac{1}{n}\mathbf{X}_2^t\mathbf{X}_2 = \frac{1}{n}\mathbf{S}_{22}$$

the same applies to the cross-covariance matrix

$$\mathbf{\Sigma}_{12} = \text{Cov}(\mathbf{X}_1, \mathbf{X}_2) = \frac{1}{n}\mathbf{X}_1^t\mathbf{X}_2 = \frac{1}{n}\mathbf{S}_{12}$$

One might use the following transform to make all features decorrelated and have unit variance.

$$\bar{\mathbf{X}}_1 = \mathbf{X}_1\mathbf{\Sigma}_{11}^{-1/2}$$

This transformation, known as *whitening*, will change the covariance of transformed data into the identity matrix.

$$\bar{\mathbf{\Sigma}}_{11} = \text{Cov}(\bar{\mathbf{X}}_1, \bar{\mathbf{X}}_1) = \frac{1}{n}\bar{\mathbf{X}}_1^t\bar{\mathbf{X}}_1 = \mathbf{\Sigma}_{11}^{-1/2}\left(\frac{1}{n}\mathbf{X}_1^t\mathbf{X}_1\right)\mathbf{\Sigma}_{11}^{-1/2} = \mathbf{I}_{\mathbf{n_1}}$$

where $\mathbf{I}_{\mathbf{n_1}}$ represents the $[n_1 \times n_1]$ identity matrix.

6.2 Canonical Correlation Analysis

The most intuitive way to understand CCA is to study the problem that originally inspired Hotelling [4] to introduce CCA. Given a sample set of n students, we examine their "reading ability" by p_1 tests and their "arithmetic ability" using some other p_2 tests. Now we would like to inquire what is the relation between reading and arithmetic abilities indicated by these tests; CCA produces the answer to this question. Here, the main assumption is that the same cohort of n students were used for both reading and arithmetic tests. Given these two views (size p_1 and p_2) of n subjects, CCA seeks a pair of linear transforms $\mathbf{w}_1 \in \mathbb{R}^{p_1}$ and $\mathbf{w}_2 \in \mathbb{R}^{p_2}$ such that correlation between transformed data is maximized:

$$\begin{aligned}
\lambda &= \max_{\mathbf{w}_1,\mathbf{w}_2} cor(\mathbf{X}_1\mathbf{w}_1, \mathbf{X}_2\mathbf{w}_2) \\
&= \max_{\mathbf{w}_1,\mathbf{w}_2} \frac{cov(\mathbf{X}_1\mathbf{w}_1, \mathbf{X}_2\mathbf{w}_2)}{\sqrt{var(\mathbf{X}_1\mathbf{w}_1)}\sqrt{var(\mathbf{X}_2\mathbf{w}_2)}} \\
&= \max_{\mathbf{w}_1,\mathbf{w}_2} \frac{{\mathbf{w}_1}^t\mathbf{X}_1^t\mathbf{X}_2\mathbf{w}_2}{\sqrt{({\mathbf{w}_1}^t\mathbf{X}_1^t\mathbf{X}_1\mathbf{w}_1)}\sqrt{({\mathbf{w}_2}^t\mathbf{X}_2^t\mathbf{X}_2\mathbf{w}_2)}} \\
&= \max_{\mathbf{w}_1,\mathbf{w}_2} \frac{{\mathbf{w}_1}^t\mathbf{S}_{12}\mathbf{w}_2}{\sqrt{({\mathbf{w}_1}^t\mathbf{S}_{11}\mathbf{w}_1)}\sqrt{({\mathbf{w}_2}^t\mathbf{S}_{22}\mathbf{w}_2)}}
\end{aligned} \tag{6.3}$$

where $\mathbf{S}$ corresponds the scatter matrices as defined in Equation 6.4

$$\text{Cov}(\mathbf{X}_1, \mathbf{X}_2) = \begin{bmatrix}\mathbf{\Sigma}_{11} & \mathbf{\Sigma}_{12} \\ \mathbf{\Sigma}_{21} & \mathbf{\Sigma}_{22}\end{bmatrix} = \frac{1}{n}\begin{bmatrix}\mathbf{X}_1^t\mathbf{X}_1 & \mathbf{X}_1^t\mathbf{X}_2 \\ \mathbf{X}_2^t\mathbf{X}_1 & \mathbf{X}_2^t\mathbf{X}_2\end{bmatrix} = \frac{1}{n}\begin{bmatrix}\mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22}\end{bmatrix} \tag{6.4}$$

Since the norm of the weight vectors does not affect the overall max value, we can fix their value by considering them as constraints

$$\begin{aligned} & \max_{\mathbf{w}_1, \mathbf{w}_2} \ {\mathbf{w}_1}^t \mathbf{S}_{12} \mathbf{w}_2 \\ \text{such that} \quad & || \ \mathbf{X}_1 \mathbf{w}_1 \ ||_2^2 = {\mathbf{w}_1}^t \mathbf{S}_{11} \mathbf{w}_1 = 1 \\ \text{and} \quad & || \ \mathbf{X}_2 \mathbf{w}_2 \ ||_2^2 = {\mathbf{w}_2}^t \mathbf{S}_{22} \mathbf{w}_2 = 1 \end{aligned} \tag{6.5}$$

Using Equation 6.5 one can derive the Lagrangian [5]

$$L(\mathbf{w}_1, \mathbf{w}_2, \lambda_1, \lambda_2) = {\mathbf{w}_1}^t \mathbf{S}_{12} \mathbf{w}_2 - \lambda_1 {\mathbf{w}_1}^t \mathbf{S}_{11} \mathbf{w}_1 - \lambda_2 {\mathbf{w}_2}^t \mathbf{S}_{22} \mathbf{w}_2 \tag{6.6}$$

whose differentiation with respect to $\mathbf{w}_1$ and $\mathbf{w}_2$ results in

$$\begin{aligned} & \begin{cases} \frac{\partial L}{\partial \mathbf{w}_1} = \mathbf{S}_{12} \mathbf{w}_2 - \lambda_1 \mathbf{S}_{11} \mathbf{w}_1 = 0 \\ \frac{\partial L}{\partial \mathbf{w}_2} = \mathbf{S}_{21} \mathbf{w}_1 - \lambda_2 \mathbf{S}_{22} \mathbf{w}_2 = 0 \end{cases} \\ \Longrightarrow & \begin{cases} \mathbf{S}_{12} \mathbf{w}_2 = \lambda_1 \mathbf{S}_{11} \mathbf{w}_1 \\ \mathbf{S}_{21} \mathbf{w}_1 = \lambda_2 \mathbf{S}_{22} \mathbf{w}_2 \end{cases} \end{aligned} \tag{6.7}$$

We can use Equation 6.7 to show that $\lambda_1 = \lambda_2$ by observing

$$\lambda_1 \mathbf{w}_1^t \mathbf{S}_{11} \mathbf{w}_1 = \mathbf{w}_1^t \mathbf{S}_{12} \mathbf{w}_2 = \mathbf{w}_2^t \mathbf{S}_{21} \mathbf{w}_1 = \lambda_2 \mathbf{w}_2^t \mathbf{S}_{22} \mathbf{w}_2$$

and the constraints $\mathbf{w}_1^t \mathbf{S}_{11} \mathbf{w}_1 = \mathbf{w}_2^t \mathbf{S}_{22} \mathbf{w}_2 = 1$. Thus one can rewrite Equation 6.7 as

$$\begin{cases} \mathbf{S}_{12} \mathbf{w}_2 = \lambda \mathbf{S}_{11} \mathbf{w}_1 \\ \mathbf{S}_{21} \mathbf{w}_1 = \lambda \mathbf{S}_{22} \mathbf{w}_2 \end{cases} \tag{6.8}$$

By defining

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{0} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{22} \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix}$$

one can rewrite Equation 6.8 as a *generalized eigenvalue* problem.

$$\mathbf{A}\mathbf{w} = \lambda \mathbf{B}\mathbf{w} \tag{6.9}$$

This problem has $p_1 + p_2$ eigenvalues $\{\lambda_1, -\lambda_1, \lambda_2, -\lambda_2, \cdots, \lambda_p, -\lambda_p, 0, \cdots, 0\}$ where $p = min(p_1, p_2)$. The eigen vectors corresponding to the paired positive and negative eigenvalues differ only in a negative sign. These eigen vectors relate to each other as $[\mathbf{w}_1, \ \mathbf{w}_2]^t$ and $[\mathbf{w}_1, \ -\mathbf{w}_2]^t$. Thus only finding the set of positive eigenvalues is sufficient to fully solve the Equation 6.9.

Similar to other methods formulated as an eigenvalue problem, CCA finds up to $p = min(p_1, p_2)$ paired projection vectors $(\mathbf{w}_{1,i}, \mathbf{w}_{2,i})$ and corresponding correlation

values λ_i. One can construct two transform matrices by concatenating these column vectors:

$$\mathbf{W}_1 = \mathbf{W}_1^p = [\mathbf{w}_{1,1},\ \mathbf{w}_{1,2},\ \ldots,\ \mathbf{w}_{1,p}]$$
$$\mathbf{W}_2 = \mathbf{W}_2^p = [\mathbf{w}_{2,1},\ \mathbf{w}_{2,2},\ \ldots,\ \mathbf{w}_{2,p}]$$

Projecting data using these transform matrices decorrelates data. More specifically, the covariance of the transformed data, $\mathbf{Z}_1 = \mathbf{X}_1\mathbf{W}_1$ and $\mathbf{Z}_2 = \mathbf{X}_2\mathbf{W}_2$, is:

$$\begin{aligned}
\mathrm{Cov}(\mathbf{Z}_1, \mathbf{Z}_2) &= \frac{1}{n}\begin{bmatrix} \mathbf{W}_1^t\mathbf{X}_1^t\mathbf{X}_1\mathbf{W}_1 & \mathbf{W}_1^t\mathbf{X}_1^t\mathbf{X}_2\mathbf{W}_2 \\ \mathbf{W}_2^t\mathbf{X}_2^t\mathbf{X}_1\mathbf{W}_1 & \mathbf{W}_2^t\mathbf{X}_2^t\mathbf{X}_2\mathbf{W}_2 \end{bmatrix} \\
&= \frac{1}{n}\begin{bmatrix} \mathbf{W}_1^t\mathbf{S}_{11}\mathbf{W}_1 & \mathbf{W}_1^t\mathbf{S}_{12}\mathbf{W}_2 \\ \mathbf{W}_2^t\mathbf{S}_{21}\mathbf{W}_1 & \mathbf{W}_2^t\mathbf{S}_{22}\mathbf{W}_2 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{W}_1^t\mathbf{\Sigma}_{11}\mathbf{W}_1 & \mathbf{W}_1^t\mathbf{\Sigma}_{12}\mathbf{W}_2 \\ \mathbf{W}_2^t\mathbf{\Sigma}_{21}\mathbf{W}_1 & \mathbf{W}_2^t\mathbf{\Sigma}_{22}\mathbf{W}_2 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{I}_{p_1} & \mathbf{\Lambda} \\ \mathbf{\Lambda}^t & \mathbf{I}_{p_2} \end{bmatrix}
\end{aligned}$$

where Λ is a $[p_1 \times p_2]$ diagonal matrix with $(\lambda_1, \ldots, \lambda_p)$ as diagonal values, sorted in the descending order $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$. Columns of transformation matrices $\mathbf{W}_1$ $\mathbf{W}_2$ are called **canonical vectors (weights)**. Columns of transformed data, $\mathbf{Z}_1$ and $\mathbf{Z}_2$, are called **canonical variates**. λ_i values are **canonical correlations**, which are the correlations between pairs of canonical variates.

In Appendix A we review many relevant properties of CCA method and several of its formulations. Here we just mention one of the formulations of CCA, which reduces CCA to a SVD decomposition problem. In this formulation we need to construct matrix $\mathcal{C}$ and use the *singular value decomposition (SVD)* [7] method to decompose it into three matrices U, Λ, and V:

$$\mathcal{C} = \mathbf{S}_{11}^{-1/2}\mathbf{S}_{12}\mathbf{S}_{22}^{-1/2} = \mathbf{\Sigma}_{11}^{-1/2}\mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1/2} = \mathbf{U}\Lambda\mathbf{V}^t \tag{6.10}$$

In this setting, the CCA directions are formed using the SVD results:

$$\begin{cases} \mathbf{W}_1 = \mathbf{S}_{11}^{-1/2}\mathbf{U} \\ \mathbf{W}_2 = \mathbf{S}_{22}^{-1/2}\mathbf{V} \end{cases} \tag{6.11}$$

Using this formulation, we can define *reflexive generalized inverse* of CCA transformation matrices:

$$\begin{cases} \mathbf{W}_1^* = \mathbf{U}^t\mathbf{S}_{11}^{1/2} \\ \mathbf{W}_2^* = \mathbf{V}^t\mathbf{S}_{22}^{1/2} \end{cases} \tag{6.12}$$

One can show these two matrices satisfy the two properties of reflexive generalized inverse of matrices, namely

$$\mathbf{W}_i^* \mathbf{W}_i \mathbf{W}_i^* = \mathbf{W}_i^* \qquad \text{and} \qquad \mathbf{W}_i \mathbf{W}_i^* \mathbf{W}_i = \mathbf{W}_i \qquad \text{for} \quad i = 1, 2$$

By comparing Equation 6.11 and Equation 6.12, one can see the direct relationship

$$\begin{cases} \mathbf{W}_1^* = \mathbf{W}_1^t \mathbf{S}_{11} \\ \mathbf{W}_2^* = \mathbf{W}_2^t \mathbf{S}_{22} \end{cases}$$

The $\mathbf{W}_i$ and $\mathbf{W}_i^*$, $i = 1, 2$ form two consecutive transforms for each view $\mathbf{X}_i$ that first project data onto the CCA directions $\mathbf{Z}_i = \mathbf{X}_i \mathbf{W}_i$ and then transform them back to the original space $\tilde{\mathbf{X}}_i = \mathbf{Z}_i \mathbf{W}_i^*$. During these two transformations, all signals in the null space of the CCA transformation will be removed from the data. In the other words, $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$ contain only the correlated signal that CCA was able to match across the two views.

In the next section we will show how we use this idea to extract the common biological signal between two batches and use these two modified versions of data sets to conduct our batch effect correction algorithm.

6.3 BECCA

In this section we introduce our batch correction method, BECCA, based on canonical correlation analysis. Without loss of generality, here we consider merging two batches (note that all results are expandable to more than two batches using multiple CCA ideas; for more details see Appendix A.) Also here we use gene and probeset interchangeably, to mean the features/covariates present in gene expression microarray.

As the main assumption of CCA indicates, we need two views of the same set of objects. Here views are batches and common objects across them are genes. In this setting, we have two gene expression data sets $\mathbf{X}_{1\ [n\times p_1]}$ and $\mathbf{X}_{2\ [n\times p_2]}$ that we want to merge. They share n genes (probesets) while the number of profiles/patients is p_1 and p_2 respectively. In this setting, each gene is considered as one subject that has been observed using two sets of features: its expression for p_1 patients in the first study and p_2 patients in the second study. Note this setting is different from "direct CCA" in which subject are patients, see Parkhomenko et al. [131] which matched the expression intensity of 3554 genes of 194 individuals with their genotypes measured

by 2882 SNP loci to find the relationship between genetic loci and gene expression phenotypes.

We assume these two batches include random samples of one common population, for example both batches are studying breast cancer patients in the age range of 40 to 60. The i^{th} row of these two batches, $\mathbf{x}_{1i} \in \mathbb{R}^{p_1}$ and $\mathbf{x}_{2i} \in \mathbb{R}^{p_2}$, contain p_1 and p_2 realizations of expression value of i^{th} gene for $i = 1 \cdots n$. We are assuming these two sets share some underlying biological signal that is being confounded by their technical differences. Comparing this setting to cross-language document models [135] is very intuitive. Each document is represented by a vector of term frequencies. Matched translated documents are used to make the two views needed for CCA. We assume there is a *semantic* relationship between the documents that is being confounded by unwanted factors such as intrinsic details of each language and/or the specific writing style of individual writers.

Here, we assume the expression matrix of each batch consists of two components: the biological-related component and technical-related component, which is the main confounding factor:

$$\begin{cases} \mathbf{X}_1 = \alpha \mathbf{Y}_1 \ + \ \beta_1 \mathbf{Z}_1 + \epsilon_1 \\ \mathbf{X}_2 = \alpha \mathbf{Y}_2 \ + \ \beta_2 \mathbf{Z}_2 + \epsilon_2 \end{cases} \tag{6.13}$$

Here, the first terms in each equation represent the biological components and the second terms represent the technical confounding components. More specifically, $\mathbf{Y}_i, \ i = 1, 2$ is a $q \times p_i$ design matrix containing q biological annotations for p_i patients in the form of an incidence matrix and α is a $n \times q$ matrix of coefficients that relates these q biological factors to the expression values of n genes. Note that the coefficients of biological factors (α) is the same in the both equations of the two batches, indicating the implicit assumption that both batches are sharing a same biological background. On the other hand, $\mathbf{Z}_{1\ [r_1 \times p_1]}$ and $\mathbf{Z}_{2\ [r_2 \times p_1]}$, which are design matrices containing the technical factors that potentially can influence the gene expression values, are specific to the equation of each batch and different from one another. As $\mathbf{Z}_1$ and $\mathbf{Z}_2$ essentially contain different technical factors that are affecting each batch, in general $r_1 \neq r_2$. The coefficients of technical factors, β_1 and β_2, determine how the technical factors affect the expression value of genes in each batch. Again, we indicate the independence of these two sets of coefficients by assigning them different subscripts. Figure 6.1 schematically represent the Equation 6.13 and uses colors to show the relation between the coefficient vectors across the two batches.

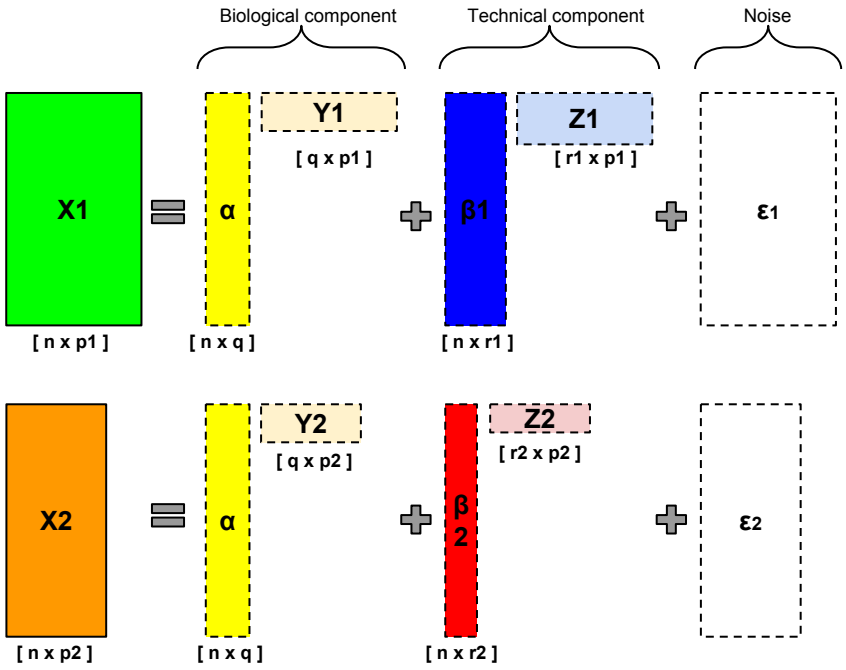


Figure 6.1: Schematic representation of the Equations underlying the BECCA method (see Section 6.13 for details). The dimensions of the matrices are drawn to reflect the expected reality: the larger dimension of $\mathbf{X}_1$ and $\mathbf{X}_2$ (n) correspond to the genes while the smaller directions (p_1 and p_2) are the patients in each study. Also note that β_1 and β_2 have different widths to show that generally their number of columns are different.
The color coding of matrices show the relation between the coefficients across the two batches: while the technical factors that influences each batch are different (**blue**) versus **red**), the common biological factor (**yellow**) is the same. The confounding role of different technical factors results in data matrices ($\mathbf{X}_1$ and $\mathbf{X}_2$) with different color blend (**green** versus **orange**).
Matrices with dashed border lines represent the hidden data while ones with solid border lines represent the observed data.

In a typical setting, all of the factors and coefficients in Equation 6.13 are hidden, as shown in Figure 6.1. The goal of batch effect removal is to remove the second term (confounding component) without modifying the first term (biological component) so the main challenge is to find a way to extract these two components from the expression matrix. Different authors address this problem in different ways. Leek and Storey [70] proposed surrogate variable analysis (SVA), which assumes that $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are given. Thus SVA first estimates the α coefficients and subtracts the biological component ($\alpha\mathbf{Y}_1$ and $\alpha\mathbf{Y}_2$) from the data and then it identifies the surrogate variables by calculating the SVD transform of the residuals matrix and finding top "eigen-genes". Gagnon-Bartsch and Speed [9] proposed *removal*

of unwanted variation (RUV), which utilize a different approach for separating the biological component from the technical component. RUV assumes there exists a set of genes called *negative control genes* that are a priori known to be unrelated to the biological factors $\mathbf{Y}_i$, thus the coefficients α for these genes is zero. RUV uses this assumption to estimate the $\mathbf{Z}_1$ and $\mathbf{Z}_2$ from the negative control genes. Similar to the SVA approach, RUV finds the direction of high variance for the negative control genes by calculating the SVD transform of the expression matrix of the negative control genes. These "eigen-genes" span the negative control gene's directions of variation, which is the "unwanted variation". RUV clears the data from this unwanted variation by projecting the data on to the null span of these eigen-genes.

Our approach, unlike SVA and RUV, does not need any prior knowledge to separate the biological component and confounding technical component; instead, it uses CCA to extract the common signal. BECCA assumes that the additive batch effects of Equation 6.13 are orthogonal to one another as well to the biological signal.

$$\begin{aligned} \alpha^t\ \beta_1 &= 0 \\ \alpha^t\ \beta_2 &= 0 \\ \beta_1^t\ \beta_2 &= 0 \end{aligned} \tag{6.14}$$

Additionally, in concordance of zero mean assumption for the columns of data matrices (6.2), we assume that the biological and technical factors' coefficients are zero mean.

$$\begin{aligned} \alpha^t\ \mathbf{1} &= \mathbf{0} \\ \beta_1^t\ \mathbf{1} &= \mathbf{0} \\ \beta_2^t\ \mathbf{1} &= \mathbf{0} \end{aligned} \tag{6.15}$$

These two sets of assumptions (6.14, 6.15) entail uncorrelated coefficient vectors [137]:

$$\mathrm{Cor}(\alpha, \beta_1) = \mathrm{Cor}(\alpha, \beta_2) = \mathrm{Cor}(\beta_1, \beta_2) = 0 \tag{6.16}$$

One can use this fact to simply prove the following lemma:

Lemma 1. *Assuming the generative model of Equation 6.13 and the two sets of assumptions 6.14 and 6.15, the cross scatter matrix only contains the covariance*

induced by biological coefficients:

$$\begin{aligned}\mathbf{S}_{12} &= \mathbf{X}_1^t\,\mathbf{X}_2 = (\alpha\mathbf{Y}_1 \;+\; \beta_1\mathbf{Z}_1)^t\;(\alpha\mathbf{Y}_2 \;+\; \beta_2\mathbf{Z}_2)\\ &= \mathbf{Y}_1^t\,\mathbf{S}_\alpha\,\mathbf{Y}_2\end{aligned}$$

where $\mathbf{S}_\alpha = \alpha^t\,\alpha$.

Assuming that the confounding factors that influence the expression values of each batch are uncorrelated to one another and also to the biological signal, the correlated signal across two batches found by CCA will be in the column space of α and thus only contain the biological signal. This is the main idea of BECCA, which enables us to separate the biological components of Equation 6.13 from the technical component.

BECCA subtracts the extracted biological components for each batch from the gene expression matrix, the resulting residual in each batch is the signal that CCA was unable to correlate across the two views and potentially is due to the technical factors. We remove these *unwanted variations* from the data using the same approach as the RUV and SVA algorithms [89], by projecting the data in each batch to the null spaces of the high variance directions of the residuals. In summary, the algorithm consists of two steps:

1. It separates the biological component from the confounding technical component by extracting the biological component, that is done by finding the correlated signal across two batches using CCA.

2. It removes the confounding component from the data by finding the top directions of variance in the confounding component and projecting the data onto the null space of these directions.

The main benefit of BECCA is its ability to separate the unwanted variability from the biological signal given only observed batches. BECCA separate them effectively even if the unwanted variability is also caused by hidden factors other than the observed batch. Any signal that is not shared across the batches is assumed to be due to unwanted factors and will be removed from the data.

This benefit will turn into the main disadvantage of BECCA if the data does not follow the BECCA assumptions. For example, if one batch contains one subtype (of the disease under study) and the other batch contains the another subtype then

BECCA will mark the subtype signal as a batch effect, as the subtype signal does not exist in both batches and so CCA can not correlated them across the batches. Note this design is a clear violation of the experimental design rules (3.3) and also against the assumption of BECCA that requires the batches to be random samples of the same population.

In the following section we will examine the details of BECCA and its special way of removing the batch effect. The actual implementation of the method in R is available on my personal website [1] and soon it will be available as a bioconductor package.

6.4 Implementation Details

As we saw earlier in Section 6.2, CCA finds the linear relationships between two views in a special way that results in high correlation across the views. Here two views are two sets of patients, $\mathbf{X}_1$ with p_1 and $\mathbf{X}_2$ with p_2 patients, which are respectively two sets of features for the same n objects –here n genes. More specifically CCA finds a pair of weight vectors, such as $\mathbf{w}_{11} \in \mathbb{R}^{p_1}$ and $\mathbf{w}_{21} \in \mathbb{R}^{p_2}$, that linearly combine the patients in each batch to form two *super-patients*, $\mathbf{a}_{11} = \mathbf{X}_1\mathbf{w}_{11}$ and $\mathbf{a}_{21} = \mathbf{X}_2\mathbf{w}_{21}$. By construction, the arrays of these super-patients are highly correlated with each other $\text{Cor}(\mathbf{a}_{11}, \mathbf{a}_{21}) = \lambda_1$. The next super-patients are formed by the next pair of linear transforms of CCA, producing correlation that typically is high, but is guaranteed to be less than the first pair of super-patients $\text{Cor}(\mathbf{a}_{12}, \mathbf{a}_{22}) = \lambda_2 < \lambda_1$. This new pair is uncorrelated with the previous pair $\text{Cor}(\mathbf{a}_{12}, \mathbf{a}_{11}) = \text{Cor}(\mathbf{a}_{21}, \mathbf{a}_{22}) = \text{Cor}(\mathbf{a}_{11}, \mathbf{a}_{22}) = \text{Cor}(\mathbf{a}_{21}, \mathbf{a}_{12}) = 0$. Ideally, each super-patient pair represents one set of biological conditions that is present in patients in both batches. Considering the cross-language example is again helpful to intuitively understand its mechanism: CCA projections, which are just a weight vector for terms in each language, form pairs of "concepts" that are highly correlated across the two languages.

Stacking all weight vectors as columns of a matrix produces transform matrices for each batch $\mathbf{W}_1 = [\mathbf{w}_{11}, \mathbf{w}_{12}, \cdots \mathbf{w}_{1p}]$ and $\mathbf{W}_2 = [\mathbf{w}_{21}, \mathbf{w}_{22}, \cdots \mathbf{w}_{2p}]$ where $p = \min(p_1, p_2)$. Using these two transform matrices we can rewrite the CCA transformations of $\mathbf{X}_1$ and $\mathbf{X}_2$ into the two sets of p highly correlated super-patients in

[1] https://sites.google.com/site/svaisipour/utilities

matrix format:

$$\begin{cases} \mathbf{A}_1 = \mathbf{X}_1 \, \mathbf{W}_1 \\ \mathbf{A}_2 = \mathbf{X}_2 \, \mathbf{W}_2 \end{cases} \tag{6.17}$$

where $\mathbf{A}_1$ and $\mathbf{A}_2$ are both $n \times p$ matrices containing p super-patients in their columns. Note that the cross correlation of these two sets of super-patients is $\text{Cor}(\mathbf{A}_1, \mathbf{A}_2) = \mathbf{\Lambda}$, where $\mathbf{\Lambda}$ is a $p \times p$ diagonal matrix with $\lambda_1, \lambda_2, \cdots, \lambda_p$ values on its main diagonal. Earlier in Lemma 1 we showed that the only component of cross correlation between $\mathbf{X}_1$ and $\mathbf{X}_2$ is the biological coefficients α in Equation 6.13. Based on this observation we can infer the following:

Lemma 2. $\mathbf{A}_1$ *and* $\mathbf{A}_2$ *are in the column space of* α*.*

Proof. Without loss of generality, we prove this property for the first pairs of weight vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ and the resulting first pair of super-patients $\mathbf{a}_1$ and $\mathbf{a}_2$. Assume the $\mathbf{w}_1$ and $\mathbf{w}_2$ are the *optimal solution* of the Equation 6.6 and the corresponding super-patients are

$$\begin{cases} \mathbf{a}_1 = \mathbf{X}_1 \, \mathbf{w}_1 \\ \mathbf{a}_2 = \mathbf{X}_2 \, \mathbf{w}_2 \end{cases} \tag{6.18}$$

We assume that this pair of super-patients *is not* in the column space of α, therefore, they each have two components: one in the column space of α (denoted by $\mathbf{a}_{.\parallel}$) and one in the null space of α (denoted by $\mathbf{a}_{.\perp}$):

$$\begin{cases} \mathbf{a}_1 = \mathbf{a}_{1\parallel} + \mathbf{a}_{1\perp} = \mathbf{X}_1\mathbf{w}_1 = \alpha\mathbf{w}_{1\parallel} + \beta_1\mathbf{w}_{1\perp} \\ \mathbf{a}_2 = \mathbf{a}_{2\parallel} + \mathbf{a}_{2\perp} = \mathbf{X}_2\mathbf{w}_2 = \alpha\mathbf{w}_{2\parallel} + \beta_2\mathbf{w}_{2\perp} \end{cases} \tag{6.19}$$

where we used the assumed additive batch model in Equation 6.13 to write the $\mathbf{a}_{.\parallel}$ component as a linear combination of columns of α and the $\mathbf{a}_{1\perp}$ and $\mathbf{a}_{2\perp}$ respectively as the linear combination of columns of β_1 and β_2. Note that, according to the Lemma 1, the following properties holds for the two components of super-patient pair:

$$\mathbf{a}_{1\parallel} \, \mathbf{a}_{1\perp} = \mathbf{a}_{1\parallel} \, \mathbf{a}_{2\perp} = \mathbf{a}_{2\parallel} \, \mathbf{a}_{1\perp} = \mathbf{a}_{2\parallel} \, \mathbf{a}_{2\perp} = 0 \tag{6.20}$$

Using the two orthogonal components of super-patient pair, we can rewrite the optimum value of Lagrangian function Equations 6.6, which was achieved by the

optimum answers $\mathbf{w}_1$ and $\mathbf{w}_2$:

$$\begin{aligned}
L^* &= \mathbf{w}_1{}^t\mathbf{S}_{12}\mathbf{w}_2 - \lambda_1\mathbf{w}_1{}^t\mathbf{S}_{11}\mathbf{w}_1 - \lambda_2\mathbf{w}_2{}^t\mathbf{S}_{22}\mathbf{w}_2 \\
&= \mathbf{w}_1{}^t\left(\mathbf{X}_1{}^t\mathbf{X}_2\right)\mathbf{w}_2 - \lambda_1\mathbf{w}_1{}^t\left(\mathbf{X}_1{}^t\mathbf{X}_1\right)\mathbf{w}_1 - \lambda_2\mathbf{w}_2{}^t\left(\mathbf{X}_2{}^t\mathbf{X}_2\right)\mathbf{w}_2 \\
&= \mathbf{a}_1{}^t\mathbf{a}_2 - \lambda_1\mathbf{a}_1{}^t\mathbf{a}_1 - \lambda_2\mathbf{a}_2{}^t\mathbf{a}_2 \\
&= \left(\mathbf{a}_{1\parallel} + \mathbf{a}_{1\perp}\right)^t\left(\mathbf{a}_{2\parallel} + \mathbf{a}_{2\perp}\right) - \lambda_1\left(\mathbf{a}_{1\parallel} + \mathbf{a}_{1\perp}\right)^t\left(\mathbf{a}_{1\parallel} + \mathbf{a}_{1\perp}\right) - \lambda_2\left(\mathbf{a}_{2\parallel} + \mathbf{a}_{2\perp}\right)^t\left(\mathbf{a}_{2\parallel} + \mathbf{a}_{2\perp}\right) \\
&= \mathbf{a}_{1\parallel}{}^t\,\mathbf{a}_{2\parallel} - \lambda_1\left(\mathbf{a}_{1\parallel}{}^t\,\mathbf{a}_{1\parallel} + \mathbf{a}_{1\perp}{}^t\,\mathbf{a}_{1\perp}\right) - \lambda_2\left(\mathbf{a}_{2\parallel}{}^t\,\mathbf{a}_{2\parallel} + \mathbf{a}_{2\perp}{}^t\,\mathbf{a}_{2\perp}\right)
\end{aligned}$$

Evaluating the value of L^* shows that one could achieve higher value for the Lagrangian function if the $\mathbf{a}_1$ and $\mathbf{a}_2$ did not include the component that lies in the null space of the α. Thus the $\mathbf{w}_1$ and $\mathbf{w}_2$ can not be the optimal answers. This contradiction shows that the optimal answers must lie in the column space of α. □

6.4.1 Step1: Separating the biological and technical signals

The main idea of BECCA is to reconstruct $\mathbf{X}_1$ and $\mathbf{X}_2$ using a few top super-patients. In the other words, we project each batch (view) into the space spanned by $k \leq p$ highly cross-correlated vectors discovered by CCA to keep only the correlated signal shared by $\mathbf{X}_1$ and $\mathbf{X}_2$. If we show the first k columns of $\mathbf{W}_1$ and $\mathbf{W}_2$ by $\mathbf{W}_1^k$ and $\mathbf{W}_2^k$ then the batch corrected data $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$ are made by following linear transform

$$\begin{cases} \tilde{\mathbf{X}}_1 = \mathbf{A}_1^k\,\mathbf{W}_1^{k*} = \left(\mathbf{X}_1\,\mathbf{W}_1^k\right)\,\mathbf{W}_1^{k*} \\ \tilde{\mathbf{X}}_2 = \mathbf{A}_2^k\,\mathbf{W}_2^{k*} = \left(\mathbf{X}_2\,\mathbf{W}_2^k\right)\,\mathbf{W}_2^{k*} \end{cases} \tag{6.21}$$

where $\mathbf{W}_1^{k*}$ and $\mathbf{W}_2^{k*}$ are generalized reflexive inverse of projection matrices defined in Equation 6.12. $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$ are the transformed data matrices that are made by projecting the batches onto the top k CCA direction and then projected back into their original space. In other words, transformation 6.21 projects each batch on the top k super-patients and then transform them back to the original space. As a result, this transformation removes all the signal that are orthogonal to the directions of the top k super-patients, meaning that $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$ only contain the signal that is common between the paired batches, *i.e.,* the shared biological signal. This way BECCA extracts the biological component of signal from each batch.

In order to separate the biological signal from the confounding technical signal, BECCA continues with subtracting the transformed batches from the original batches. The resulting residual matrices contain the confounding signal in each

batch:

$$\begin{cases} \mathbf{R}_1 = \mathbf{X}_1 - \tilde{\mathbf{X}}_1 \\ \mathbf{R}_2 = \mathbf{X}_2 - \tilde{\mathbf{X}}_2 \end{cases} \tag{6.22}$$

At this point, the first step of BECCA, which is separating the biological component from the confounding technical component, is done.

6.4.2 Step2: Removing the unwanted variation

The second step of BECCA is simply finding the high variance directions of residual matrices ($\mathbf{R}_1$ and $\mathbf{R}_2$) and removing them from the batches. This step, which is adopted from SVA and RUV, simply calculates the SVD transformations of $\mathbf{R}_1$ and $\mathbf{R}_2$ and finds the high variance directions. These are "unwanted variations" that are caused by factors other than biological ones. BECCA simply removes these unwanted variations by projecting each batch onto the orthogonal space spanned by these high variance directions, *i.e.,* their null space. Thus the second step start with SVD transformations of $\mathbf{R}_1$ and $\mathbf{R}_2$:

$$\begin{cases} \mathbf{R}_1 = \mathbf{U}_1 \, \mathbf{D}_1 \, \mathbf{V}_1^t \\ \mathbf{R}_2 = \mathbf{U}_2 \, \mathbf{D}_2 \, \mathbf{V}_2^t \end{cases} \tag{6.23}$$

According to the singular values, we pick k_1 and k_2 directions as the unwanted ones. In practice, usually the top one (or two) singular value is several orders of magnitude larger than the rest of the singular values; thus by evaluating the distribution of singular values one can simply pick the correct number of k_1 and k_2. After deciding on the values of k_1 and k_2, BECCA forms two *projector matrices*, one for each batch:

$$\begin{cases} \mathbf{Project}_1 = \mathbf{V}_1^{k_1} \, \left(\mathbf{V}_1^{k_1}\right)^t \\ \mathbf{Project}_2 = \mathbf{V}_2^{k_2} \, \left(\mathbf{V}_2^{k_2}\right)^t \end{cases} \tag{6.24}$$

where $\mathbf{V}_i^{k_i}$ denotes the top k_i right singular vectors, *i.e.,* the columns of $\mathbf{V}_i$, for $i = 1, 2$. By applying these two projectors to the original data matrices, $\mathbf{X}_1$ and $\mathbf{X}_2$, one can project the data to the directions of the unwanted variations. Thus the null space of these projectors will be free of the unwanted variations due to confounding technical factors:

$$\begin{cases} \mathbf{X}_1^* = \mathbf{X}_1 \, (\mathbf{I} - \mathbf{Project}_1) \\ \mathbf{X}_2^* = \mathbf{X}_2 \, (\mathbf{I} - \mathbf{Project}_2) \end{cases} \tag{6.25}$$

Finally, $\mathbf{X}_1^*$ and $\mathbf{X}_2^*$ are the corrected data using BECCA.

6.4.3 Parameter Settings

BECCA has three parameters that need to be set by the user, namely k, k_1, and k_2. For easier understanding here is a brief explanation of each one's role:

- k: the number of CCA directions that are used to retain the biological signal between the two batches, $1 \leq k \leq p = min(p_1, p2)$.
- k_1: the number of directions of unwanted variation in batch one, $1 \leq k_1 \leq p_1$.
- k_2: the number of directions of unwanted variation in batch two, $1 \leq k_2 \leq p_2$.

Note that the other two similar algorithms, SVA and RUV, also have the k_1 and k_2 parameters, however, the parameter k only is needed by BECCA. This extra parameter is the price we pay for not using any prior knowledge, as SVA and RUV do.

Setting the values of k_1 and k_2 is very straightforward. By considering the singular values of Equation 6.23 for each batch, one can decide how many of the "eigen-genes" of the residual matrices contain the variations due to the batch effect. In practice, usually, the 1-2 top singular values are several orders of magnitude larger than the rest of them. Thus setting k_1 and k_2 to be equal to this number will result in the best performance, removing the unwanted variations from data, without removing all the variations that are due to intrinsic differences between the subjects in the study.

Setting the value of k is harder, as values that are too big will cause BECCA to include some of the confounding signal as biological signal, but setting it to be too small will cause BECCA to ignore a lot of biological signal and potentially loosing them. In Chapter 7 we will see the effect of setting the value of this parameter on the performance of BECCA.

6.4.4 Properties

We show below that the transformation 6.21 reduces the rank of covariance and cross-covariance matrices to k. We believe this rank reduction is due to the removal of unwanted variations that are caused by technical factors that independently in-

fluenced each batch.

$$\begin{aligned}\mathrm{Cov}(\tilde{\mathbf{X}}_1) = \tilde{\mathbf{\Sigma}}_{11} = \frac{1}{n}\tilde{\mathbf{X}}_1^t\tilde{\mathbf{X}}_1 =& \frac{1}{n}\mathbf{W}_1^{*k^t}\,\mathbf{W}_1^{k^t}\,\mathbf{X}_1{}^t\,\mathbf{X}_1\,\mathbf{W}_1^k\,\mathbf{W}_1^{*k}\\ =& \frac{1}{n}\mathbf{S}_{11}^{1/2}\mathbf{U}^k\,\mathbf{U}^{k^t}\mathbf{S}_{11}^{-1/2}\,\mathbf{S}_{11}\,\mathbf{S}_{11}^{-1/2}\mathbf{U}^k\,\mathbf{U}^{k^t}\mathbf{S}_{11}^{1/2}\\ =& \frac{1}{n}\mathbf{S}_{11}^{1/2}\mathbf{U}^k\,\mathbf{U}^{k^t}\mathbf{S}_{11}^{1/2}\\ =& \mathbf{\Sigma}_{11}^{1/2}\mathbf{U}^k\,\mathbf{U}^{k^t}\mathbf{\Sigma}_{11}^{1/2} = \left(\mathbf{\Sigma}_{11}^{1/2}\mathbf{U}^k\right)\,\left(\mathbf{\Sigma}_{11}^{1/2}\mathbf{U}^k\right)^t\end{aligned}$$

where $\mathbf{U}^k$ is the first k columns of $\mathbf{U}$. One can similarly show that the covariance of the second view is

$$\mathrm{Cov}(\tilde{\mathbf{X}}_2) = \tilde{\mathbf{\Sigma}}_{22} = \mathbf{\Sigma}_{22}^{1/2}\mathbf{V}^k\,\mathbf{V}^{k^t}\mathbf{\Sigma}_{22}^{1/2} = \left(\mathbf{V}^{k^t}\mathbf{\Sigma}_{22}^{1/2}\right)^t\,\left(\mathbf{V}^{k^t}\mathbf{\Sigma}_{22}^{1/2}\right)$$

and the cross covariance is

$$\mathrm{Cov}(\tilde{\mathbf{X}}_1, \tilde{\mathbf{X}}_2) = \tilde{\mathbf{\Sigma}}_{12} = \mathbf{\Sigma}_{12}^{1/2}\,\mathbf{U}^k\mathbf{\Lambda}^k\mathbf{V}^{k^t}\,\mathbf{\Sigma}_{22}^{1/2} = \tilde{\mathbf{\Sigma}}_{11}^{1/2}\,\mathbf{\Lambda}^k\,\tilde{\mathbf{\Sigma}}_{22}^{1/2}$$

Using the covariances of the transformed views, one can reconstruct the SVD based formulation of CCA (Equation 6.10) for the transformed data

$$\tilde{\mathcal{C}} = \tilde{\mathbf{\Sigma}}_{11}^{-1/2}\,\tilde{\mathbf{\Sigma}}_{12}\,\tilde{\mathbf{\Sigma}}_{22}^{-1/2} \tag{6.26}$$

$$= \tilde{\mathbf{\Sigma}}_{11}^{-1/2}\,\left(\tilde{\mathbf{\Sigma}}_{11}^{1/2}\,\mathbf{\Lambda}^k\,\tilde{\mathbf{\Sigma}}_{22}^{1/2}\right)\,\tilde{\mathbf{\Sigma}}_{22}^{-1/2} = \mathbf{\Lambda}^k \tag{6.27}$$

thus the 6.21 transformations modifies each view in a way that changes the relationship between their covariances and cross-covariances in $\tilde{\mathcal{C}}$ (Equation 6.27) to a diagonal matrix.

6.5 Summary

This chapter introduced BECCA, a batch effect correction method using canonical correlation analysis (CCA). First it offers a basic intuitive introduction of CCA along with its fundamental formulations. As CCA has been utilized in many different applications and disciplines, there are several different perspectives to analyze its algorithm and consequently formulate and implement it. Studying all these perspectives is beneficial for better understanding of the method; we summarize most of the available approaches in Appendix A.

After introducing the CCA in Section 6.2, we introduced our batch correction method. Conceptually, our method is very similar to SVA [70] and RUV-2 [9] as it

first separates the biological signals from the non-biological signals and then uses a SVD transform to discover the main directions of variation in the non-biological data. These methods assume that the variations along these discovered directions are only due to the technical factors, and so removes them from the data to correct the batch effects. The only difference between these methods and ours is the way the biological and non-biological signal are separated. SVA relies on a comprehensive biological annotation and RUV-2 relies on the assumption of the existence of "negative control genes" (genes whose expression intensities are assumed to be independent of biological factors).

Our method, however, assumes that the batch effects, which influence the expressions of each batch, are independent from each other and therefore the highly correlated direction discovered by CCA contains only common biological signals across batches. We find the signal orthogonal to these common directions and consider it to be the non-biological signal. At this point our method proceeds in the same way as the other two batch correction algorithms, by finding the main directions of variation in the non-biological part. As Gagnon-Bartsch and Speed [9] mentioned, the orthogonality of technical factors and biological factors plays an important role in the effectiveness of their algorithm. This will apply to our method too. In Chapter 7 we will see how our method's performance stands versus other batch correction algorithms.

Chapter 7

Empirical analysis of BE correction and gene selection methods

This chapter presents the results of our experimental results and showcases all ideas presented in this dissertation. More specifically, the experiments will demonstrate how the feature selection algorithms of Chapter 4 can reduce the influence of batch effects on the performance of downstream analysis. For this purpose, we will utilize the algorithms proposed in Chapter 5 to quantify the influence of the technical factors. We also use the data sets introduced in Section 5.1 to empirically compare the performance of different batch correction algorithms and rank them against our proposed method, BECCA.

This chapter is organized as follows: first we look at the effect of gene selection algorithms on the confounding influence of batch effects in Section 7.1. This section describes experiments that show the importance of selecting a relevant subset of genes before any down-stream analysis, in order to remove the misleading genes. Section 7.2.1 compares the performance of several batch correction methods applied to the mouse study, introduced in Section 5.1.

7.1 Feature selection and batch effects

In this section we look at the effects of feature selection algorithms on the BE. Combining gene expression data sets that are conducted under different technical conditions results in some "evidence" that suggests the existence of batch effects. Chapter 5 introduced some techniques to quantify the existence of batch effects. This section, specifically, studies the effect of running gene selection algorithms on a

data set that has batch effects. In other words, this section studies how limiting the analysis to a subset of genes reduces the confounding influence of technical factors.

In the first experiment we analyze the breast cancer study that was introduced in Section 5.1. This study is a 2×2 factorial study conducted on 16 breast cancer samples that are profiled in 2 labs using 2 types of Affymetrix GeneChips, *i.e.,* $16 \times 2 \times 2 = 64$ profiles that are 4 technical replicates of 16 specimens. We used the unsupervised BE detection method (introduced in Section 5.2) to evaluate the effect of gene selection algorithms on the influence of technical factors, here, the lab that conducted the experiment and the used GeneChip. We use k-mean clustering with number of clusters set to $k = 4$ (as there are 4 batches) to cluster these 64 gene expression profiles and then compare the result of clustering with the batch labeling. If the formed clusters are very similar to the batches then we can infer that unfortunately BE is the dominant signal. On the other hand, if the 4 technical replicates of each of the 16 patients are close together and there is no sign of technical difference between them, then we can infer that the biological signal is dominant and the BE is absent or less strong. We use two metrics to measure the concordance between the formed clusters and the batches, *corrected rand index (CRI)* and *variation of information (VI)*; for more details about these metrics see Section 5.2. Note that the ideal score of CRI is 0 (meaning that there is no relation between the batches and the clusters) while for VI, larger numbers are better.

We repeat this unsupervised analysis five times, once using all 12092 genes during the clustering process and the other four times using only the 1000 genes that were selected by the four algorithms introduced in Chapter 4. The result of this analysis is summarized in Table 7.1, one column for each analysis. Note 3 of these feature selection algorithms (variance based ranking, mean intensity ranking, and integrating correlation) improved these scores, presumably by choosing a subset of genes that are less influenced by technical factors. In the other words, when we use the genes selected by these methods, BE seems to have less influence on the data. The correlation increment method, however, manages to find a subset of genes whose expression values are more confounded by technical factors and as a result are unable to cluster the data according to the biological differences, instead the formed clusters almost perfectly correspond to the batches. Note this method is purposefully designed to choose the most misleading genes (see Section 4.2.3) to demonstrate that there are some genes that are more severely influenced by technical

factors.

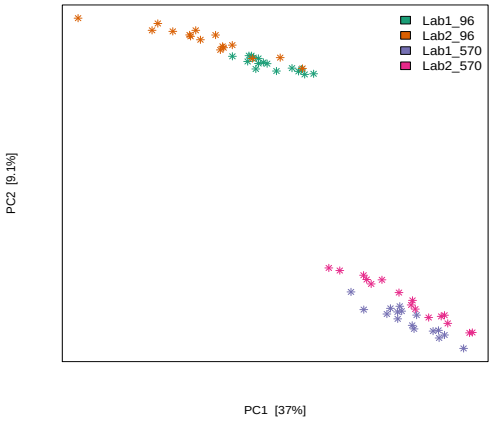
	All genes (12092)	Var rank (1000)	Mean rank (1000)	Intgr Cor (1000)	Cor Inc (1000)
Corrected Rand Inx (CRI)	0.663	**0.026**	0.043	0.119	0.847
Variation of Info (VI)	0.807	**2.535**	2.417	2.280	0.362

Table 7.1: Comparison of the scores of clustering analysis of 64 gene expression profiles of GSE17700 [121] into 4 clusters. This data set contains 16 breast cancer specimens that are profiled on two different Affymetrix GeneChips in two different labs – so there are $16 \times 2 \times 2 = 64$ instances (see Section 5.1 for more details). The scores of using all genes is compared to 1000 genes selected by 4 feature selection algorithms. The best scores for each row (cluster analysis metric) is highlighted in bold, *i.e.,* the smallest CRI score and the largest VI score.

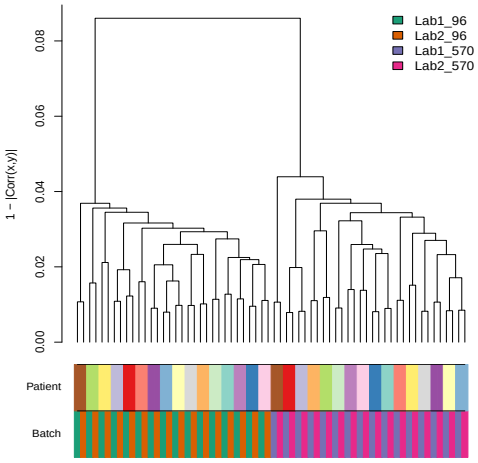
Table 7.1 shows that filtering out some of the genes reveals the biological relations between gene expression profiles that otherwise would be masked by technical factors when all the genes were considered. Also note that, among the three feature selection algorithms that improved the CRI and VI scores, our proposed algorithm (variance based ranking) achieved the best performance. As we will see in the other experiments of this section, this feature selection algorithm consistently achieves the best empirical performance in all conducted experiments. For visual comparison, Figure 7.1 compares the results of the clustering using all genes (first row) to the best (second row) and worst (third row) performing gene selection algorithms.

Figure 7.1(a) shows that, when we use all 12092 genes, the instances of batches that are conducted on the same GeneChip are closely clustered to each other in the 2-dimensional space of the top principal components (lab1-96 and lab2-96 cloud versus lab1-570 and lab2-570 cloud). Panel (b) shows the same behavior: the two main clusters are formed by instances of the same platforms. Note the "Batch" color bar on bottom has two main clusters: lab1-96 and lab2-96 versus lab1-570 and lab2-570. Also note that, in the hierarchical clustering result, pairs of technical factors are clustered together (top color code bar tagged as "patient"). However, the 4 technical replicates of each patients are not all clustered together as they instead formed 2 clusters, each containing a pair of technical replicates.

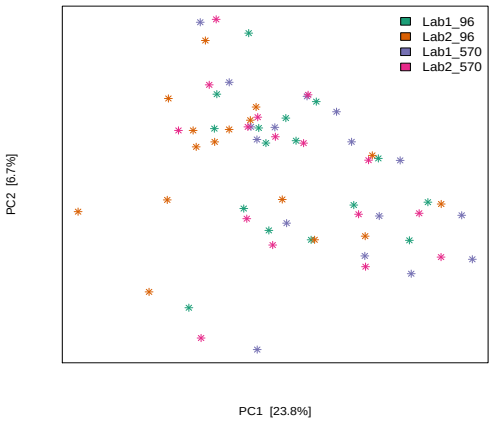
The correct clustering of all 4 replicates together happens in panel (d), which shows the clustering using the 1000 genes selected by variance-based feature selection (see Section 4.2.2). Studying the length of vertical branches in the hierarchical clustering of panel (d) shows that most of the distance is caused by difference from one patient to another while the 4 technical replicates of each patient are very



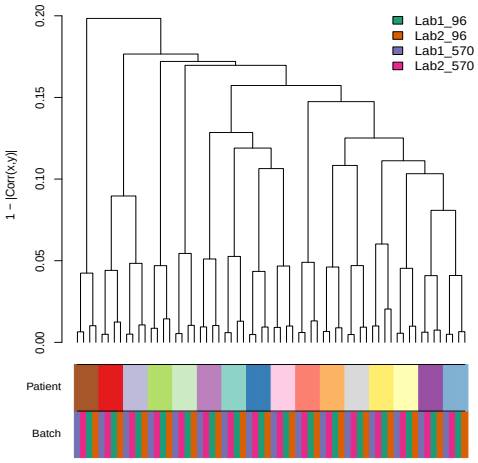
(a) All 12092 genes



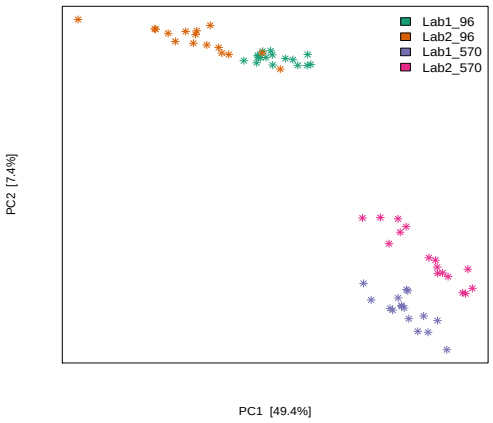
(b) All 12092 genes



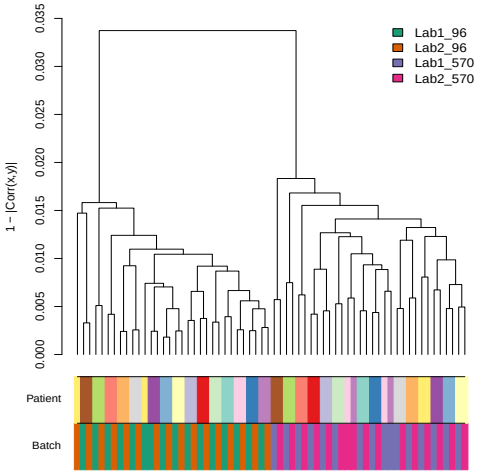
(c) 1000 genes (variance based ranking)



(d) 1000 genes (variance based ranking)



(e) 1000 genes (correlation increment)



(f) 1000 genes (correlation increment)

Figure 7.1: Visual analysis of unsupervised analysis of GSE17700 (breast cancer study): left column depicts the first two principal components plot where each cross is one gene expression profile in this 2-dimensional space colored according to its batch, and the right column depicts the hierarchical clustering results marked with two color coding bars; the upper bar represents the *patient id* and the lower bar represents the batches.

similar. This panel clearly shows how, by limiting the calculations to a meaningful subset of genes, the unsupervised learning algorithm was able to discover the biological relation between the technical replicates. Panel (c), which depicts the principal component plot based on the same set of genes, also shows a very good mixing of the instances of 4 batches and there is no clear bound between the instances of batches, similar to panel (a)

Finally, panels (e) and (f) show the analysis of 64 profiles using the 1000 most *misleading* genes selected by correlation increment gene selection algorithm. Recall this algorithm is designed to degrade the results by finding genes that increase the correlation across batches. Panel (e) shows more clear boundaries between the batches, especially the lab1-570 and lab2-570 cloud are not mixed the way they were in panel (a). Here, the set of genes selected by correlation increment algorithm even further emphasizes the technical differences between the batches. The same interpretation applies to panel (f) where we not only have two big cluster similar to panel (b), but also some of the pairs of technical replicates that are not matched with each other, indicating that, less biological signal is present in this restricted data.

In the second set of experiments in this section, we use the idea of Figure 5.9 to relate gene selection in the severeness of BE. As we saw in Section 5.3, comparing the set of differentially expressed genes that are found *within-batch*, versus *across-batch*, shows that the confounding factors strongly affected the data. More specifically, when the number of significant genes across-batches is an order of magnitude larger than the number of significant genes found within-batches, one can reason that the across-batch significant genes are caused by technical differences rather than the biological differences between case and control groups.

A large gap between the number of significant genes across-batch and the number of significant genes within-batch indicates the stronger confounding influence of BE on gene expression intensities. A good feature selection, however, will filter out the genes that are more influenced by batch effects and, as a result, will shrink the gap between the across-batch and within-batch significant genes. Note that feature selection is not a general cure for batch effects; it only reduces BE's destructive influence by focusing on genes that are less influenced by technical factors.

In the first experiment, we study the two breast cancer studies we introduced in Section 4.1, *GSE2034* [110] and *GSE7390* [111], which seeks the set of genes that are

differentially expressed between ER+ and ER- patients. Table 7.2 summarizes the results of comparing the performance of 5 feature selection algorithms (introduced in Chapter 4): random (with 100 repetitions), variance-based ranking (Section 4.2.2), mean intensity based ranking (Section 4.2.2), integrative correlation (Section 4.2.1), and correlation increment (Section 4.2.3).

No. of selected genes		All	10000	8000	6000	4000	2000	1000
Random	% Within	16.32	16.31±0.15	16.34±0.25	16.36±0.38	16.35±0.42	16.29±0.29	16.51±1.19
	% Across	92.55	92.55±0.13	92.54±0.17	92.54±0.21	92.55±0.31	92.56±0.56	92.64±0.77
Var Rnk	% Within	-	**19.37**	**22.84**	**25.97**	30.03	**36.20**	**42.00**
	% Across	-	**91.07**	**89.00**	**86.12**	**81.67**	**72.50**	**60.00**
Mean Rnk	% Within	-	18.42	20.52	22.10	22.75	23.55	23.50
	% Across	-	91.48	90.52	89.90	88.40	86.45	83.90
Integ Cor	% Within	-	19.35	22.79	26.53	**30.55**	34.75	39.30
	% Across	-	92.34	91.85	91.32	90.18	87.80	85.10
Cor Incr	% Within	-	15.82	15.01	14.50	13.50	12.20	11.10
	% Across	-	92.96	92.76	92.85	92.53	92.20	92.10

Table 7.2: The percentage of significant genes between ER+ and ER- breast cancer patients discovered using two studies: GSE2034(209 ER+, 77 ER-) and GSE7390 (134 ER+, 34 ER-) and by performing t-test at 0.001 significance level. Each column indicates the number of genes used for the experiment, starting with all genes (12092). Each row contains one feature selection algorithm: **Random** (randomly choose genes, reported results are for 100 repetitions), **Var Rnk** (variance based ranking 4.2.2), **Mean Rnk** (mean intensity based ranking 4.2.2), **Integ Cor** (Integrative correlation analysis 4.2.1), and **Cor Incr** (correlation increment gene selection algorithm 4.2.3). For each combination of feature selection algorithm and number of features, two numbers are reported **% Within** (the number of common significant genes found by two within-batch analyses) and **% Across** (the number of common significant genes found two across-batches analyses). The best performance in each column is highlighted: the highest for % Within values and the lowest for % Across values.

In each cell of this table (*i.e.,* the combination of gene number and algorithm) there are two numbers, *% Within* and *% Across*, which are the percentage of genes that are found significant within-batch and the percentage of genes that are found significant across batches, respectively. In the other words, the *% Within* shows how many genes (out of, say 12092 genes) are tagged as significant by both within-batch significant analyses (Figure 5.9, the two bottom ovals) and the *% Across* shows how many genes are tagged significant by two across-batch significant analyses (Figure 5.9, the two top-right ovals). For example, according to Table 7.2, when all 12092 genes are included in the calculations, the *% Within* is 16.32 and *% Across* is 92.55, meaning that, out of 12092 genes, only 1973 genes are tagged significant by both within-batch analyses while 11,191 genes are tagged significant by both across-batch analyses.

Ideally, in the absence of any technical difference between the two batches, we expect these two numbers to be: ***% Within*** $\approx$ ***% Across*** *i.e.,* the set of significant differently expressed genes between case and control instances should not be affected by the batch from which the case and control instance are chosen. In reality,

however, there is a big discrepancy between the *% Within* and *% Across* sets. For example, according to Table 7.2, when all 12092 genes are included in the analysis, within-batch analyses find only 1973 significant genes while across-batch analyses find 11,191 genes. Obviously, this big discrepancy between these two numbers is due to the batch effect, *i.e.,* most of the genes that are tagged significant when we compare case versus control across batches are due to the technical difference between case and control rather than the biological differences between them. What we expect from an effective gene selection algorithm is to shrink the gap between these two sets by filtering out genes whose expression intensity does not reflect the biological relationship between the instances due to the confounding influence of unwanted technical factors. Note that a gene selection algorithm does not remove the batch effect; it only reduces the damaging impact of batch effects by filtering out genes that are more influenced by these unwanted technical factors.

Also note that the *% Within* value without applying any feature selection is 16.32 and it remains approximately the same when we apply random feature selection algorithm. This means that when we choose a random sample of, say size 1000, from 12092 available genes, on average 16.32% of them will be called significant by both within-batch significant analyses. The same applies to *% Across*: without any feature selection its values is 92.55 and for all random samples this values is approximately the same.

An *unsupervised* gene selection algorithm that increases the *% Within* scores is probably filtering out uninformative genes that do not contain important biological information. One can safely infer that this algorithm's criteria for choosing genes is trustworthy and applying it before the downstream analysis will be beneficial. On the other hand, a gene selection algorithm that decreases the *% Within* value is probably choosing genes based on a problematic criterion, which means filtering out genes using this algorithm may lose informative genes. Therefore, a desired gene selection algorithm should increase *% Within* values and decrease *% Across* values and thus shrink the gap between these two scores. In fact, the "oracle" feature selection algorithm will choose a set of features that produced scores 100% for both.

By examining the performance of algorithms in Table 7.2, one can observe three trends: (1) random feature selection algorithm: does not change the scores; (2) variance ranking, mean intensity ranking, and integrative ranking follow the desired pattern (increase the *% Within* values and decrease the *% Across* values); and (3)

correlation increment algorithm follows the inverse of desired pattern (note this algorithm was purposefully designed to harm the data (Section 4.2.3) by selecting the most misleading genes). For each column in this table, we highlighted the best scores (the highest *% Within* and the lowest *% Across* values) and see that almost all best scores are achieve by variance based ranking algorithm. By limiting the number of genes from all 12092 genes to just (a good set of) 1000, this algorithm is able to decrease the gap between *% Within* and *% Across* from $92.55 - 16.32 = 76.23\%$ to $60.00 - 42.00 = 18.00\%$. In this experiment, the integrative correlation based feature selection algorithm's performance is very close to the variance based ranking. However, the computation time of these two methods are not comparable: while the proposed method (variance based ranking) runs seemingly instantly on a machine with 8GB available memory, we were not able to run integrative correlation algorithm on the same machine. The running time of integrative correlation on a machine with 20GB memory was more than 5 minutes.

We performed the same analysis on the mouse study (for more details, refer to Section 5.1 and Section 7.2.1) to find the significant differentially expressed genes between the two strains of mouse used in this study. Table 7.3 summarizes the results of this experiment. Most of the findings of Table 7.2 also occur in Table 7.3: Random feature selection does not change the scores for *% Within* and *% Across* and their values stay the same as when we use all 21187 genes; variance based ranking consistently improves both scores and manages to decrease the gap between *% Within* and *% Across* from $96.51 - 5.98 = 90.53$ to $69.60 - 13.80 = 55.8$. The mean based ranking and integrative correlation analysis manages to increase the *% Within* a little bit by choosing 10,000 and 8,000 genes, but their performance degrades as the number of selection genes become smaller. In fact, integrative correlation algorithm manages to achieve the worst *% Within* score of 0.20% for 1000 selected genes, meaning that both within-batch analyses only agreed that 2 genes out of these 1000 genes were significant. Finally, correlation increment algorithm consistently degrades both *% Within* and *% Across* scores. Again the best scores in each column is highlighted and in this case, for all selected gene numbers, variance-based gene ranking algorithm achieved the best scores.

This section's experiments suggest that conducting an effective gene selection algorithm in order to remove genes that are highly influenced by batch effects is a useful practice. Gene selection does not replace the batch correction algorithms;

No. of selected genes		All	10000	8000	6000	4000	2000	1000
Random	% Within	5.89	5.86±0.16	5.92±0.22	5.85±0.27	5.91±0.36	6.00±0.55	6.01±0.65
	% Across	96.51	96.52±0.13	96.52±0.15	96.52±0.21	96.49±0.25	96.53±0.37	96.58±0.53
Stnd Dev	% Within	-	**9.65**	**10.93**	**11.78**	**12.55**	**12.30**	**13.80**
	% Across	-	**93.71**	**92.51**	**90.68**	**87.67**	**80.50**	**69.60**
Mean	% Within	-	7.70	7.55	7.08	6.55	6.80	6.00
	% Across	-	95.16	94.75	94.53	94.17	94.55	95.00
Integ Cor	% Within	-	9.13	7.61	4.92	2.45	0.85	0.20
	% Across	-	94.62	93.90	93.10	91.47	87.65	82.60
Cor Incr	% Within	-	4.26	3.96	3.92	3.72	3.20	2.10
	% Across	-	95.59	95.31	94.63	93.70	91.55	88.00

Table 7.3: The percentage of significant genes between two mouse strains (*B6* versus *NOD*) discovered using two batches, one conducted using *Mouse Gene 1.0 ST Array* and other one *Mouse Gene 1.1 ST Array*, and by performing t-test at 0.001 significance level. Each column indicates the number of genes used for the experiment, starting with all genes (21187). Each row contains one feature selection algorithm: **Random** (randomly choose genes, reported results are for 100 repetitions), **Stnd Dev** (variance based ranking 4.2.2), **Mean** (mean intensity based ranking 4.2.2), **Integ Cor** (Integrative correlation analysis 4.2.1), and **Cor Incr** (correlation increment gene selection algorithm 4.2.3).
For each combination of feature selection algorithm and number of feature, two numbers are reported **% Within** (the number of common significant genes found by two within-batch analyses) and **% Across** (the number of common significant genes found two across-batches analyses). The best performance in each column is highlighted: the highest for % Within values and the lowest for % Across values.

however, it does improve their performance (and also the performance of all other downstream analyses) by removing genes that seem contain no useful biological information. The next section will look at the actual batch correction algorithms and compare their performance using several experiments.

7.2 Batch effect correction comparison

This section describes several BE correction algorithms and compares their performance using the techniques introduced in Chapter 5. More specifically, using the supervised detection algorithms introduced in Section 5.3, we compare the set of differentially expressed genes before and after applying the BE correction algorithms. For finding the set of significant differentially expressed genes between two groups of biologically different profiles, such as *case* versus *control*, we use the simple t-test statistic and then use a particular significance level to determine whether a gene is significant.

The following two sections compare the performance of our proposed method, BECCA, to four other batch correction algorithms: ComBat [69], RUV2 [9], BMC [86], and DWD [67]. All BE correction methods are conducted in an unsupervised mode, *i.e.,* the biological labeling of gene expression profiles is not observed by BE correction algorithms. The only observed information is the batch grouping, meaning that the algorithms know to which batch each profile belongs.

In addition to this, RUV2 requires a priori known set of negative control genes (see Section 6.3). We used Affymetrix GeneChip documentations to find the housekeeping probesets and used them as negative control genes. Affymetrix's human GeneChip platforms, such as *HGU95A*, *HGU95Av2*, *HGU133A*, or *HGU133Plus2*, annotate the housekeeping probesets with *AFFX* labels. We used these genes as negative control genes for RUV2 method.

We run all algorithms using R version 3.0.1 and bioconductor version 2.12. For ComBat we used the implementation provided by authors, available at `http://www.bu.edu/jlab/wp-assets/ComBat/Download.html`. We used the inSilicoMerging package [138] to run BMC and DWD methods. We implemented the RUV2 method according to the details provided by Gagnon-Bartsch and Speed [9], Jacob et al. [89]. We implemented the *naive unsupervised RUV2* and estimated the unobserved technical covariates $\mathbf{W}$ using SVD and considered the unobserved coefficients as fixed effects.

7.2.1 GSE33822 (mouse brain study)

As Section 5.1 earlier explained, Sun et al. [122] conducted a study that includes $48 = 2{\times}8{\times}3$ mouse brain spicemens: 2 mouse strains *C57BL/6J* versus *NOD/ShiLtJ* (here referred as *B6* and *NOD*), 8 mice per strain, and 3 spicemens from each mouse: *forebrain*, *hindbrain*, and *whole brain*. Each of these 48 specimens was profiled on two Affymetrix GeneChips: *Mouse Gene 1.0 ST Array* and *Mouse Gene 1.1 ST Array*, which share 21187 common probesets. We refer to these two 48×21187 expression matrices as *batch-10* and *batch-11*. Here, the main biological signals that should lead to differential expressed genes are mouse strain and brain region; and the main source of batch effect is the platform.

In this section we use the idea of Figure 7.2 to compare 3 sets of significant differentially expressed genes: (1) within-batch, (2) across-batch, and (3) across-batch false positive. For all significant analysis tests we used the Student t-test with significant level 0.05 applied to p-values. By comparing the number of genes in these three sets, one can evaluate the confounding influence of unwanted technical factors on biological signal. Two *evaluative criteria* indicate the influence of batch effects:

- Number of genes in across-batch false positive $\neq$ zero.
- Number of genes in within-batch $\ll$ number of genes in across-batch.

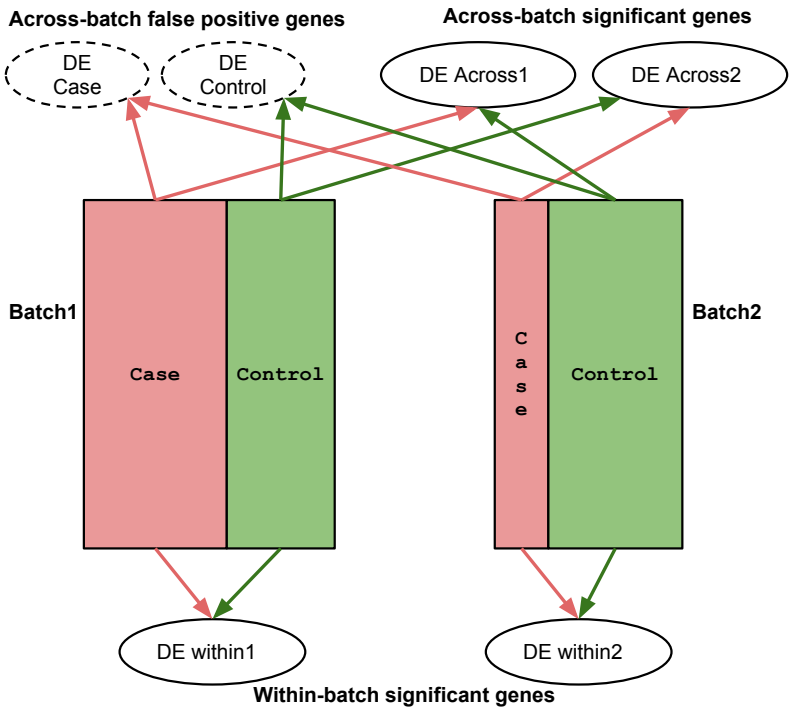


Figure 7.2: Finding the set of significant differently expressed genes between case and control groups, *within-batch* (bottom), *across-batch* (top-right), and *across-batch false positives* (top-left).
This figure is the same as Figure 5.9, just redrawn for easier reference.

The reason why these two criteria indicate BE is very straightforward: comparing two case groups (or two control groups) from different batches should essentially result in no significant genes, as we are comparing two sets of profiles that have similar biological condition. If we find any genes that appears significant, it means the significance is due to the batch factors rather than the biological factor, therefore, across-batch false positive $\neq$ zero is an indication of BE influence. Similarly, comparing the case group versus the control group should result in approximately the same number of significant genes, regardless of the batch to which these groups belong. Again, if the number of across-batch significant genes are much larger than the number of within-batch significant genes, one can infer that the across batch comparison is influenced by the technical differences between the two batches.

We use these two criteria to evaluate how different BE correction algorithms reduce the confounding influence of BE. Moreover, if the two within-batch significant genes have many genes in common with each other, then one can be more certain that these genes are truly associated with the biological factor of interest. Similarly,

if a particular gene is declared significant by both across-batch analyses, there is a higher chance that this gene's expression is truly related to the biological signal. Therefore, if all 4 significant analyses have many genes in common we suggest there is less batch effect influence on the biological signal.

We summarize all these numbers using tabular structure similar to the two tables in Figure 5.11. Figure 7.3 has 12 replicates of this table. As one can see, the within-batch numbers are highlighted by the grey cell colors and the redundant values in these tables are replaced with **"."** as the table is symmetrical. Also cells on the main diagonal contain **"-"** to indicate the absence of any significant genes (comparing a group of gene expression profiles to itself does not result in any significant gene). These tables also include numbers that show the consistency of two within-batch analyses, two across-batch analysis, and all four of them.

Figure 7.3 is organized as 2 columns and 6 rows; each row contains the results of one correction algorithm (the first row is the uncorrected data), whereas the columns are related to the biological annotation used for dividing the profiles into *case* versus *control* groups. The left column contains the significant analysis of two brain regions (*i.e.,* forebrian versus hindbrain) and the right column has the results of mouse strain comparison (*i.e.,* B6 versus NOD).

We start analyzing Figure 7.3 by looking at the top-left table, *i.e.,* the uncorrected results for brain region comparison. The two evaluative criteria for this table are:

- Number of genes in across-batch false positive
  - 21049 (Batch-10/FB vs. Batch11/FB)
  - 21079 (Batch-10/HB vs. Batch11/HB)

  is $\neq$ **zero**.
- Number of genes in within-batch
  - 13980 (Batch-10/FB vs. Batch10/HB)
  - 12247 (Batch-11/FB vs. Batch11/HB)

  is $\ll$ number of genes in across-batch
  - 20755 (Batch-10/FB vs. Batch11/HB)

Uncorrected

Tissue : forebrain (FB) vs. hindbrain (HB)

		Batch-10		Batch-11	
		FB	HB	FB	HB
Batch-10	FB	-	13980	21049	20755
	HB	.	-	20844	21079
Batch-11	FB	.	.	-	12247
	HB	.	.	.	-

Shared Across	20464 (96.83%) out of 21135
Shared Within	10985 (71.13%) out of 15444
Shared All	10406 (49.23%) out of 21139

Strain: B6 vs. NOD

		Batch-10		Batch-11	
		B6	NOD	B6	NOD
Batch-10	B6	-	5370	21071	21024
	NOD	.	-	21039	21072
Batch-11	B6	.	.	-	5725
	NOD	.	.	.	-

Shared Across	20923 (98.97%) out of 21140
Shared Within	4110 (58.82%) out of 6987
Shared All	4009 (18.96%) out of 21145

BECCA

		Batch-10		Batch-11	
		FB	HB	FB	HB
Batch-10	FB	-	13822	115	14625
	HB	.	-	12861	93
Batch-11	FB	.	.	-	14255
	HB	.	.	.	-

Shared Across	12618 (84.85%) out of 14871
Shared Within	12131 (76.07%) out of 15948
Shared All	11712 (72.5%) out of 16154

		Batch-10		Batch-11	
		B6	NOD	B6	NOD
Batch-10	B6	-	3545	93	2803
	NOD	.	-	3052	79
Batch-11	B6	.	.	-	2816
	NOD	.	.	.	-

Shared Across	2564 (77.86%) out of 3293
Shared Within	2389 (60.12%) out of 3974
Shared All	2287 (56.37%) out of 4057

ComBat

		Batch-10		Batch-11	
		FB	HB	FB	HB
Batch-10	FB	-	13978	254	13064
	HB	.	-	13096	372
Batch-11	FB	.	.	-	12452
	HB	.	.	.	-

Shared Across	11968 (84.32%) out of 14194
Shared Within	10989 (71.16%) out of 15443
Shared All	10772 (68.59%) out of 15706

		Batch-10		Batch-11	
		B6	NOD	B6	NOD
Batch-10	B6	-	5373	51	4990
	NOD	.	-	5320	66
Batch-11	B6	.	.	-	5729
	NOD	.	.	.	-

Shared Across	4763 (85.84%) out of 5549
Shared Within	4113 (58.83%) out of 6991
Shared All	4062 (57.45%) out of 7070

RUV2

		Batch-10		Batch-11	
		FB	HB	FB	HB
Batch-10	FB	-	12091	1168	11162
	HB	.	-	11636	925
Batch-11	FB	.	.	-	11426
	HB	.	.	.	-

Shared Across	10196 (80.89%) out of 12604
Shared Within	9882 (72.46%) out of 13637
Shared All	9562 (67.98%) out of 14065

		Batch-10		Batch-11	
		B6	NOD	B6	NOD
Batch-10	B6	-	6466	265	6905
	NOD	.	-	6526	322
Batch-11	B6	.	.	-	7696
	NOD	.	.	.	-

Shared Across	5827 (76.61%) out of 7606
Shared Within	4988 (54.36%) out of 9176
Shared All	4837 (51.69%) out of 9357

BMC

		Batch-10		Batch-11	
		FB	HB	FB	HB
Batch-10	FB	-	13980	904	12966
	HB	.	-	13172	1301
Batch-11	FB	.	.	-	12247
	HB	.	.	.	-

Shared Across	11926 (83.9%) out of 14214
Shared Within	10985 (71.13%) out of 15444
Shared All	10721 (68.26%) out of 15707

		Batch-10		Batch-11	
		B6	NOD	B6	NOD
Batch-10	B6	-	5370	225	4981
	NOD	.	-	5290	265
Batch-11	B6	.	.	-	5725
	NOD	.	.	.	-

Shared Across	4740 (85.67%) out of 5533
Shared Within	4110 (58.82%) out of 6987
Shared All	4043 (57.21%) out of 7067

DWD

		Batch-10		Batch-11	
		FB	HB	FB	HB
Batch-10	FB	-	13980	865	13251
	HB	.	-	12882	1275
Batch-11	FB	.	.	-	12447
	HB	.	.	.	-

Shared Across	11949 (84.23%) out of 14186
Shared Within	10985 (71.13%) out of 15444
Shared All	10733 (68.38%) out of 15695

		Batch-10		Batch-11	
		B6	NOD	B6	NOD
Batch-10	B6	-	5370	245	5033
	NOD	.	-	5267	264
Batch-11	B6	.	.	-	5725
	NOD	.	.	.	-

Shared Across	4567 (79.63%) out of 5735
Shared Within	4110 (58.82%) out of 6987
Shared All	4007 (56.25%) out of 7124

Figure 7.3

– 20844 (Batch-10/HB vs. Batch11/FB)

In comparison, the results of the BECCA (left column, second row) suggest much less influence of BE:

- Number of genes in across-batch false positive
  - 115 (Batch-10/FB vs. Batch11/FB)
  - 93 (Batch-10/HB vs. Batch11/HB)

  is $\approx$ **zero**.
- Number of genes in within-batch
  - 13822 (Batch-10/FB vs. Batch10/HB)
  - 14255 (Batch-11/FB vs. Batch11/HB)

  is $\approx$ number of genes in across-batch
  - 14625 (Batch-10/FB vs. Batch11/HB)
  - 12861 (Batch-10/HB vs. Batch11/FB)

The same trend is observed for the other 4 BE correction methods, however, none reduced the number of across-batch false positive as much as BECCA. The sum of the across-batch false positive for BECCA is $115 + 93 = 208$, while this number for ComBat is 626, for RUV2 is 2093, for DWD is 2140, and for BMC is 2205.

The other interesting consequence of conducting the BE correction algorithms on data is how they change the consistency of significant gene sets. According to the uncorrected results, although 21139 (out of 21187 genes, $\approx 99.8\%$) genes are declared significant by at least one of the 4 sets (two within-batch and two across-batch significant analyses), only 49.23% of them are declared significant by all 4 sets. BECCA increases this number to 72.5%, suggesting that now the 4 significant sets are more consistent with each other. Note that BECCA's number is larger than other 4 correction algorithms and thus, according to this measure, it ranks better than those algorithms.

The right column of Figure 7.3, which shows the number of significant genes between the two strains of mouse used in the study, exhibits the same results. While

in the uncorrected data, there are many across-batch false positive genes (21071 and 21072), utilizing BECCA reduces these numbers to $93 + 79 = 172$. According to this criteria, BECCA ranked second best BE correction algorithm after ComBat with $51 + 66 = 117$ across-batch false positive. Note that utilizing the BE correction algorithms increases the consistency of significant genes from 18.96% in uncorrected data to $\approx 57\%$.

7.2.2 Breast cancer study

In this section we will compare the significant genes between two subtypes of breast cancer, ER+ and ER- . For this purpose, we use instances of two publicly available gene expression data sets: GSE2034 and GSE7390 (for more details see Section 4.1). We process both data sets using the BrainArray [82] custom cdf files version 17.0.0 and the RMA [106] function in R, resulting in 12092 probesets. We use the student t-test to identify the genes that are differentially expressed between the two subgroups of patients, ER+ versus ER-. The number of significant genes (out of 12092 genes) at a 0.001 significant level is reported in each of the 12 tables in Figure 7.4. Highlighted grey cells contain the within-batch numbers and as each table is symmetrical, redundant values are avoided by putting **"."** in corresponding cells. The main diagonal cells are filled with **"-"** indicating there are no significant genes associated with them.

Reading and analyzing the 12 tables of Figure 7.4 is done the same way as we studied Figure 7.3. Here, both of the 2 main columns analyze the ER+ versus ER-; the left column considers all 12092 genes while the right column deals only with the 1000 genes with the highest variance.

Studying the two *evaluative criteria*, introduced in Section 7.2, for the top-left table (all genes in uncorrected data) will suggest the strong confounding influence of BE:

- Number of genes in across-batch false positive
  - 11640 (GSE2034/ER+ vs. GSE7390/ER+)
  - 11847 (GSE2034/ER- vs. GSE7390/ER-)

  is $\neq$ zero.
- Number of genes in within-batch

ER+ versus ER-

All genes (12092)

		GSE2034		GSE7390	
		ER+	ER-	ER+	ER-
Uncorrected					
GSE2034	ER+	-	3323	11640	11630
	ER-	.	-	11532	11847
GSE7390	ER+	.	.	-	2644
	ER-	.	.	.	-
Shared Across		11191 (93.48%) out of 11971			
Shared Within		1974 (49.41%) out of 3995			
Shared All		1585 (13.23%) out of 11982			
BECCA					
GSE2034	ER+	-	2514	2	2120
	ER-	.	-	1311	0
GSE7390	ER+	.	.	-	1276
	ER-	.	.	.	-
Shared Across		1227 (55.62%) out of 2206			
Shared Within		1061 (38.85%) out of 2731			
Shared All		991 (35.81%) out of 2767			
ComBat					
GSE2034	ER+	-	3323	3	3088
	ER-	.	-	2534	0
GSE7390	ER+	.	.	-	2645
	ER-	.	.	.	-
Shared Across		2293 (68.84%) out of 3331			
Shared Within		1974 (49.4%) out of 3996			
Shared All		1947 (48.25%) out of 4035			
RUV2					
GSE2034	ER+	-	957	73	836
	ER-	.	-	660	61
GSE7390	ER+	.	.	-	754
	ER-	.	.	.	-
Shared Across		504 (50.7%) out of 994			
Shared Within		356 (26.23%) out of 1357			
Shared All		330 (22.76%) out of 1450			
BMC					
GSE2034	ER+	-	3325	24	3044
	ER-	.	-	2483	9
GSE7390	ER+	.	.	-	2644
	ER-	.	.	.	-
Shared Across		2237 (67.95%) out of 3292			
Shared Within		1974 (49.39%) out of 3997			
Shared All		1920 (47.49%) out of 4043			
DWD					
GSE2034	ER+	-	3325	22	3104
	ER-	.	-	2461	13
GSE7390	ER+	.	.	-	2644
	ER-	.	.	.	-
Shared Across		2218 (66.23%) out of 3349			
Shared Within		1974 (49.39%) out of 3997			
Shared All		1901 (46.65%) out of 4075			

1000 high variance genes

		GSE2034		GSE7390	
		ER+	ER-	ER+	ER-
Uncorrected					
GSE2034	ER+	-	584	854	803
	ER-	.	-	767	921
GSE7390	ER+	.	.	-	463
	ER-	.	.	.	-
Shared Across		600 (61.79%) out of 971			
Shared Within		420 (66.99%) out of 627			
Shared All		183 (18.83%) out of 972			
BECCA					
GSE2034	ER+	-	504	0	453
	ER-	.	-	365	0
GSE7390	ER+	.	.	-	324
	ER-	.	.	.	-
Shared Across		354 (75.97%) out of 466			
Shared Within		297 (55.72%) out of 533			
Shared All		291 (54.6%) out of 533			
ComBat					
GSE2034	ER+	-	584	1	554
	ER-	.	-	471	0
GSE7390	ER+	.	.	-	463
	ER-	.	.	.	-
Shared Across		445 (76.72%) out of 580			
Shared Within		420 (66.99%) out of 627			
Shared All		404 (63.82%) out of 633			
RUV2					
GSE2034	ER+	-	251	8	239
	ER-	.	-	180	6
GSE7390	ER+	.	.	-	175
	ER-	.	.	.	-
Shared Across		158 (60.08%) out of 263			
Shared Within		113 (35.87%) out of 315			
Shared All		112 (34.04%) out of 329			
BMC					
GSE2034	ER+	-	584	3	555
	ER-	.	-	466	0
GSE7390	ER+	.	.	-	463
	ER-	.	.	.	-
Shared Across		450 (78.81%) out of 571			
Shared Within		420 (66.99%) out of 627			
Shared All		407 (64.4%) out of 632			
DWD					
GSE2034	ER+	-	584	27	596
	ER-	.	-	484	22
GSE7390	ER+	.	.	-	463
	ER-	.	.	.	-
Shared Across		389 (56.3%) out of 691			
Shared Within		420 (66.99%) out of 627			
Shared All		362 (50.42%) out of 718			

Figure 7.4

  - 3323 (GSE2034/ER+ vs. GSE2034/ER-)
  - 2644 (GSE7390/ER+ vs. GSE7390/ER-)

  is $\ll$ number of genes in across-batch

  - 11630 (GSE2034/ER+ vs. GSE7390/ER-)
  - 11532 (GSE2034/ER- vs. GSE7390/ER+)

BECCA removes the BE from the data successfully and improves both evaluative criteria:

- Number of genes in across-batch false positive
  - 2 (GSE2034/ER+ vs. GSE7390/ER+)
  - 0 (GSE2034/ER- vs. GSE7390/ER-)

  is $\approx$ **zero**.

- Number of genes in within-batch
  - 2514 (GSE2034/ER+ vs. GSE2034/ER-)
  - 1276 (GSE7390/ER+ vs. GSE7390/ER-)

  is $\approx$ number of genes in across-batch
  - 2120 (GSE2034/ER+ vs. GSE7390/ER-)
  - 1311 (GSE2034/ER- vs. GSE7390/ER+)

BECCA removes the BE signal more conservatively than ComBat. Thus the number of *true* significant genes within-batch and across-batch is lower. Further analysis of other BE correction algorithms shows that RUV2 is even more conservative and its number of significant genes is less than BECCA. Better performance of ComBat is also manifested by the consistency of significant genes: ComBat finds 1947 genes significant by all 4 analyses (two within-batch and 2 across-batch), which is 48.25% of declared significant genes. In comparison, BECCA only finds 35.81% of genes consistently significant by all 4 analysis and RUV2 22.76%.

We can understand the reason of this lower performance by comparing BMC results to ComBat results. As one can see, BMC results are very similar to ComBat results, especially the across-batch and within-batch numbers. This means that the

BE influence in this data set is less complicated and by subtracting means one can remove the confounding influence of BE. ComBat, being a *location-scale method* [2], manages to perform accordingly and remove BE without distorting the biological signal. On the other hand, BECCA and RUV2, being *matrix-factorization based*, fail to produce the same result. Therefore, we anticipate that BECCA will perform better than ComBat when the batch effect is more complex.

Comparing the left column to the right column, one can see another evidence supporting the claim of Section 7.1 regarding the better performance of high variance genes. In all 6 rows of Figure 7.4, the consistency of significant genes in better in the right column, *i.e.,* when using only the 1000 genes with high variance. For example, 13.23% of the genes are declared significant by all 4 analyses for uncorrected data, but when we only consider 1000 high variance genes, this number increases to 18.83%. The same improvement is observed for all other rows: BECCA from 35.81% to 54.6%, ComBat from 48.25% to 63.82%, RUV2 from 22.76% to 34.03%, BMC from 47.49% to 64.40%, and DWD from 46.65% to 50.42%,

Our classifier-based analysis of batch effects (see Section 5.3), confirms the findings of the DE based analysis. Figure 7.5 summarizes the results of applying KNN ($k = 3$) classification algorithm to the task of distinguishing ER+ breast cancer samples from ER- samples. The classifier is trained within each batch (grey cells in each table) as well as across the batches. Here, the same two datasets of Figure 7.4 (GSE2034 and GSE7390) are used as the two batches. The left column contains the results using all available genes while the right column displays the results based on the top 1000 high variance genes.

Both tables in the top row contain a block of four 100.0 scores, which means four *across-batch* classification tasks perfectly distinguished samples of the two classes, even when these samples are biologically the same (ER+ versus ER+ and ER- versus ER-). This is caused by the same phenomena that we observed earlier in Figure 7.4 and discussed it in Section 3.3. These four classifiers perfectly separate samples because of the technical bias caused by the BE, rather than the biological differences between samples.

The bottom row shows the affect of removing the technical bias between the batches using the BECCA method. We like to emphasize three important observations about comparing the top row and the bottom row:

1. Within-batch accuracies are almost constant after the batch correction.

ER+ versus ER-

All genes (12092)

			GSE2034		GSE7093	
			ER+	ER-	ER+	ER-
Uncorrected	GSE2034	ER+	-	81.12	**100.0**	100.0
		ER-	74.74	-	100.0	**100.0**
	GSE7093	ER+			-	
		ER-				-
BECCA	GSE2034	ER+	-	80.07	**58.87**	85.31
		ER-	74.24	-	85.71	**66.76**
	GSE7093	ER+			-	
		ER-				-

1000 high variance genes

		GSE2034		GSE7093	
		ER+	ER-	ER+	ER-
GSE2034	ER+	-	83.92	**100.0**	100.0
	ER-	81.31	-	100.0	**100.0**
GSE7093	ER+			-	
	ER-				-
GSE2034	ER+	-	84.62	**57.45**	84.83
	ER-	80.81	-	83.52	**60.93**
GSE7093	ER+			-	
	ER-				-

Figure 7.5: Leave one out classification accuracy of ER status prediction using KNN algorithm ($k = 3$).
Comparing within-batch performance (shaded area) to across-batch perfromance in top row shows a significant difference between them which indicates the existance of BE in the uncorrected data.
The **highlighted values** (ER+/ER+ and ER-/ER- *across batches*) are related to the classification of sample of the same biological group. There is no biological distinction between these paitients and only the technical bias makes them different. Note that after batch correction, these classification accuracies reduces from 100.0 to ≈ 50.00. This indicates that, after correction, there is no evident difference between biologically similar patients, as desired.

2. After correction, across-batch accuracies are almost equal to within-batch numbers.

3. After correction, ER+/ER+ and ER-/ER- accuracies are ≈ 50.00.

These three criteria confirm that, in this example, the BE correction algorithm was able to successfully remove the technical differences without distorting the biological signal of interest.

By comparing the left column of Figure 7.5 to its right column, one can see the affect of removing low variance genes on the severity of BE. Especially in the bottom row, using high variance genes improves all three criteria. This is consistent with our previous observations regarding the benefits of using variance-based feature selection to reduce the negative affect of technical bias.

7.3 Summary

This chapter examines how the ideas and algorithms proposed throughout this thesis perform when using real gene expression data sets. These experiments are designed to explore two goals: (1) to demonstrate the influence of feature selection on batch effects, and (2) to compare the performance of batch effect correction algorithms.

In all experiments of this chapter to measure the amount of batch effect influence, we used the techniques introduced in Chapter 5.

Section 7.1 looked at the effect of feature selection on reducing the influence of confounding factors. According to our results, in the big pool of measured gene expressions in microarray data sets, there exist some genes that are less influenced by unwanted technical factors and contain more useful biological signal for learning algorithms; and there are some genes that appear to contain very little, or no biological information. If an unsupervised gene selection algorithm (one that does not use any gene or instance labeling) is able to utilize some heuristics to choose genes from the former group, then it can effectively reduce the influence of unwanted technical factors and emphasize the biological relation between the instances across the batches. Our experiments show that the variance based ranking feature selection, introduced in Section 4.2.2, consistently chooses the best subset of genes comparing to other feature selection algorithms introduced in Chapter 4. In contrast, the correlation increment gene selection algorithm, introduced in Section 4.2.3, which was purposefully designed to choose the most misleading genes, consistently chooses a subset of genes that are more influenced by batch effects and thus limiting the analysis to these genes will result in worse results. In summary, these experiments show that it is essential to remove these genes before conducting the batch correction algorithms or down stream analysis.

Section 7.2 studies the effect of conducting different BE correction algorithms and compares their performance to our proposed method BECCA (see Chpater 6). These empirical results show that BECCA performs as effectively as state-of-the-art algorithms such as ComBat and RUV. Our experiments show BECCA transforms data in a way that the set of significant genes found within-batch and across-batch agree with each other and that the number of false-positives across-batches is significantly reduced. This behavior was consistently observed in all experiments. That section also looked at the performance of BE correction algorithms after applying variance-based ranking gene selection; the performance of all BE correction algorithms improved when they were applied to the selected features. This experiment shows that using variance-based feature selection not only is effective on its own (as experiments of Section 7.1 showed) but also it improves the performance of BE correction algorithms by removing severely confounded genes.

Chapter 8

Conclusion and Future Work

As the number of features (p) is much smaller than the number of samples (n) in gene expression microarrays, applying many of the standard leaning algorithms to gene expression studies will not produce effective classifiers. This thesis deals with the *"large p, small n"* challenge in the gene expression microarrays and compares different ways to reduce the gap between these two numbers. Solutions for this challenge can be divided into two main groups: (1) methods that reduce p by applying feature selection algorithms and (2) methods that increase n by combining several gene expression studies together.

We considered several *unsupervised* gene selection algorithms in Chapter 4 and compared their performance experimentally in Section 7.1. Our results show that the best performing gene selection algorithm is variance-based gene ranking; it basically ranks genes by aggregating their ranked variance in each batch and then removes genes with lower rankings. We proposed an algorithm that efficiently aggregates the variance ranking of genes in several batches and produces one final ranking for them, *i.e.,* we extend the single-study variance-based feature selection into a multi-study gene selection algorithms that computationally is linear in the number of studies. The feature selection algorithm based on this idea consistently achieved better results compared to other gene selection algorithms in our experiments. In fact, in Section 4.5 we observed that similarity between the variance of genes across two gene expression studies, is a strong indicator of the similarity of the biological phenomenons in each study [**Contribution 4**, Section 1.1].

The other main finding related to the feature reduction was achieved by proposing a novel algorithm that ranks genes based on a criterion that increases the correlation between gene expression profiles across studies. This feature selection algorithm

is able to identify a set of "less informative" or misleading genes that do not represent the biological differences between samples – using just this subset of genes, the mean correlation between the profiles of any two studies is around 0.95, regardless of the biological properties of the profiles such as their tissue of origin. For example, in Section 4.3 we observed that using these genes the mean correlation between ovarian cancer instances and prostate cancer instances was more than 0.95, while using another subset of genes the mean correlation was less than 0.6, which is a more reasonable correlation score for cancers from different tissues. Experimental results of Section 7.1 confirmed that these "less informative" genes are significantly more influenced by technical factors comparing to all other genes selected by other algorithms as well as random selection of genes. These results suggest that there are some genes in gene expression studies that do not contain any useful biological information and are heavily influenced by technical factors; failing to remove them will degrade the performance of downstream analysis. Therefore, we highly recommend applying a feature selection algorithm to every gene expression data set before conducting any analysis on it [**Contribution 3**, Section 1.1].

The alternative way to deal with "large p, small n" problem is to increase n by combining publicly available gene expression data sets containing profiles with similar biological properties. The main obstacle to use this solution is the *batch effects*, the unwanted influence of technical factors on gene expression intensities that confounds the effect of biological factors in each data set differently and therefore makes the intensities across data sets not comparable. Therefore, in order to combine several gene expression studies together, first one needs to remove the batch effects from each study. Several algorithms have been proposed for this purpose; in this thesis we introduce a novel batch correction algorithm based on canonical correlation analysis (CCA) called BECCA.

Our method's main assumption is that the common signal across studies is the biological signal while the signal caused by the technical factors in each study is independent from the others and also from the biological signal. BECCA first deploys CCA to find the common signal across data sets and then subtracts this component from the expression values. The variance in the resulting residual matrix is believed to be due to the unwanted technical factors, which one needs to remove from the data. We remove their influence by finding the main directions of variance in the residual matrix of each batch and subtract it from the associated expression matrix

[**Contribution 1**, Section 1.1].

We experimentally compared the performance of BECCA to other popular batch correction algorithms using several real-world gene expression data sets in Section 7.2. In order to evaluate their performance, we used the ideas proposed in Chapter 5. According to these measures, BECCA performs competitively in all experiments and successfully removes the influence of technical factors from the gene expression data sets. The performance of BECCA in Chapter 7 suggests that our method's assumptions are valid in real-world gene expression microarrays [**Contribution 2**, Section 1.1].

To compare the performance of different batch correction algorithms and also to measure the amount of confounding influence of technical factors as a function of selected genes, we used the evaluation techniques introduced in the Chapter 5. These techniques provide a wide range of visual and qualitative measures to evaluate the amount of batch effects present in a dataset. In the experiments of Chapter 7, we mostly used the performance measure based on the number of differentially expressed genes within and across batches (see Section 5.3) which is an extension to the analyses conducted by Sims et al. [86]. We believe this evaluation method, with its minimal assumptions and simple computations, is a very effective way to evaluate the existence of batch effects [**Contribution 5**, Section 1.1].

One of the main challenges of using BECCA is setting the correct value of its parameters: the number of canonical directions preserved to separate the shared biological signal, and the number of high-variance dimensions in residual matrices chosen to subtract from the data. Currently, we are using a simple heuristic for this purpose, as running exhaustive search to find the optimal values of all 3 parameters is computationally very expensive. In the future, we plan to explore better ways to set the BECCA parameters to isolate and subtract the batch effect more effectively.

Another challenge faced by batch effects correction algorithms is the composition of the batches, *i.e.,* the biological properties, such as cancer sub-types and clinical outcomes, of instances in each batch. Many batch correction algorithms, including BECCA, assume the biological composition of batches are similar, *i.e.,* instances of all batches are random samples of one common population and there is no major biological difference between batches. Therefore, batches are assumed to have proportionally equal number of instances from each biological group. This is an assumption that is hard to evaluate as in some cases the key biological properties

may not even be known. Sims et al. [86] conducted some experiments on two main sub-types of breast cancer (ER+ versus ER-) and concluded that the accuracy of predictions and batch corrections is highly dependent upon the composition of the data sets and patient characteristics. Finding a way to distinguish the differences in biological composition from the technical differences across batches is a very important task to decide whether two batches are good candidates to be merged together. Dealing with the *covariate shift* [139] in the presence of batch effects is another challenge we would like to explore in the future.

As BECCA does not depend on the biological labeling of instances or any special properties of genes and because it does not make any strong assumptions about the generating source of the batch effects and their distribution, we believe it will perform effectively in other domains where the signal of interest is confounded by the influence of irrelevant technical factors – including other biological high-throughput technologies such as proteomics, bead chips, mass spectrometers, and next-generation DNA sequencing [3]. Applying BECCA to data from other domains and evaluating its performance is another direction we would like to explore. For this purpose we have implemented BECCA in an R script that is freely from `https://sites.google.com/site/svaisipour/utilities` so other researchers can easily apply it to their data sets.

Appendix A

Canonical Correlation Analysis Formulations

This chapter provides a detailed review of canonical correlation (CCA) analysis and its different formulations. It shares some materials with Chapter 6, as in that chapter we provided the basic idea behind CCA. This appendix provides a full overview of different ways to solve CCA problem.

Formulation Assumptions

We assume data matrices (batches) contain observations in the rows and features in the columns. So $\mathbf{X}_1$ $_{[n\times p_1]}$ includes observations vectors of $\mathbf{x}_{1i} \in \mathbb{R}^{p_1}$ for $i = 1 \cdots n$. We can show data matrices using their rows

$$\mathbf{X}_1 = [\mathbf{x}_{11}, \cdots, \mathbf{x}_{1n}]^t$$
$$\mathbf{X}_2 = [\mathbf{x}_{21}, \cdots, \mathbf{x}_{2n}]^t$$

CCA analysis typically assumes $\mathbf{X}_1$ $_{[n\times p_1]}$ and $\mathbf{X}_2$ $_{[n\times p_2]}$ are two views of n same objects that study them by p_1 and p_2 features respectively. We show the column-wise merging of two views by $\mathbf{X} = [\mathbf{X}_1\ \mathbf{X}_2]$ which is a $n \times (p_1 + p_2)$ data matrix. We also assume each of these data matrices is mean centered.

$$\mathbf{X}_1^t \mathbf{1} = \mathbf{0}$$
$$\mathbf{X}_2^t \mathbf{1} = \mathbf{0}$$

where $\mathbf{1}$ is a column vector of all 1's. This assumption makes the *scatter matrices* $\mathbf{S}$ proportional to covariance matrices $\mathbf{\Sigma}$

$$\begin{aligned}\mathbf{\Sigma}_{11} = \text{Cov}(\mathbf{X}_1, \mathbf{X}_1) &= \text{E}[(\mathbf{X}_1 - \text{E}[\mathbf{X}_1])^t(\mathbf{X}_1 - \text{E}[\mathbf{X}_1])] \\ &= \text{E}[\mathbf{X}_1^t\mathbf{X}_1] \\ &= \frac{1}{n}\mathbf{X}_1^t\mathbf{X}_1 = \frac{1}{n}\mathbf{S}_{11}\end{aligned}$$

and similarly

$$\mathbf{\Sigma}_{22} = \text{Cov}(\mathbf{X}_2, \mathbf{X}_2) = \frac{1}{n}\mathbf{X}_2^t\mathbf{X}_2 = \frac{1}{n}\mathbf{S}_{22}$$

the same applies to the cross-covariance matrix

$$\mathbf{\Sigma}_{12} = \text{Cov}(\mathbf{X}_1, \mathbf{X}_2) = \frac{1}{n}\mathbf{X}_1^t\mathbf{X}_2 = \frac{1}{n}\mathbf{S}_{12}$$

One might use the following transform to make all features decorrelated and have unit variance.

$$\bar{\mathbf{X}}_1 = \mathbf{X}_1\mathbf{\Sigma}_{11}^{-1/2}$$

This transformation is known as *whitening* and it will change the covariance of transformed data into identity matrix.

$$\bar{\mathbf{\Sigma}}_{11} = \text{Cov}(\bar{\mathbf{X}}_1, \bar{\mathbf{X}}_1) = \frac{1}{n}\bar{\mathbf{X}}_1^t\bar{\mathbf{X}}_1 = \mathbf{\Sigma}_{11}^{-1/2}\left(\frac{1}{n}\mathbf{X}_1^t\mathbf{X}_1\right)\mathbf{\Sigma}_{11}^{-1/2} = \mathbf{I}_{\mathbf{n_1}}$$

where $\mathbf{I}_{\mathbf{n_1}}$ represents $[n_1 \times n_1]$ identity matrix.

CCA

Given two views of size p_1 and p_2 of n subjects, CCA seeks a pair of transforms $\mathbf{w}_1 \in \mathbb{R}^{p_1}$ and $\mathbf{w}_2 \in \mathbb{R}^{p_2}$ such that correlation between transformed data is maximized

$$\begin{aligned}\lambda &= \max_{\mathbf{w}_1,\mathbf{w}_2} cor(\mathbf{X}_1\mathbf{w}_1, \mathbf{X}_2\mathbf{w}_2) \\ &= \max_{\mathbf{w}_1^1,\mathbf{w}_2} \frac{cov(\mathbf{X}_1\mathbf{w}_1, \mathbf{X}_2\mathbf{w}_2)}{\sqrt{var(\mathbf{X}_1\mathbf{w}_1)}\sqrt{var(\mathbf{X}_2\mathbf{w}_2)}} \\ &= \max_{\mathbf{w}_1,\mathbf{w}_2} \frac{{\mathbf{w}_1}^t\mathbf{X}_1^t\mathbf{X}_2\mathbf{w}_2}{\sqrt{({\mathbf{w}_1}^t\mathbf{X}_1^t\mathbf{X}_1\mathbf{w}_1)}\sqrt{({\mathbf{w}_2}^t\mathbf{X}_2^t\mathbf{X}_2\mathbf{w}_2)}} \\ &= \max_{\mathbf{w}_1,\mathbf{w}_2} \frac{{\mathbf{w}_1}^t\mathbf{S}_{12}\mathbf{w}_2}{\sqrt{({\mathbf{w}_1}^t\mathbf{S}_{11}\mathbf{w}_1)}\sqrt{({\mathbf{w}_2}^t\mathbf{S}_{22}\mathbf{w}_2)}}\end{aligned} \tag{A.1}$$

where $\mathbf{S}$ corresponds the scatter matrices as defined in Equation A.2

$$\text{Cov}(\mathbf{X}_1, \mathbf{X}_2) = \begin{bmatrix}\mathbf{\Sigma}_{11} & \mathbf{\Sigma}_{12} \\ \mathbf{\Sigma}_{21} & \mathbf{\Sigma}_{22}\end{bmatrix} = \frac{1}{n}\begin{bmatrix}\mathbf{X}_1^t\mathbf{X}_1 & \mathbf{X}_1^t\mathbf{X}_2 \\ \mathbf{X}_2^t\mathbf{X}_1 & \mathbf{X}_2^t\mathbf{X}_2\end{bmatrix} = \frac{1}{n}\begin{bmatrix}\mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22}\end{bmatrix} \tag{A.2}$$

Since the norm of the weight vectors does not affect the overall max value, we can fix their value by considering them as constraints

$$\begin{aligned} & \max_{\mathbf{w}_1,\mathbf{w}_2} \mathbf{w}_1{}^t\mathbf{S}_{12}\mathbf{w}_2 \\ \text{such that} \quad & || \mathbf{X}_1\mathbf{w}_1 ||_2^2 = \mathbf{w}_1{}^t\mathbf{S}_{11}\mathbf{w}_1 = 1 \\ \text{and} \quad & || \mathbf{X}_2\mathbf{w}_2 ||_2^2 = \mathbf{w}_2{}^t\mathbf{S}_{22}\mathbf{w}_2 = 1 \end{aligned} \tag{A.3}$$

one can easily show that optimization problem in Equation A.3 is equal to minimizing the distance between transformed $\mathbf{X}_1$ and transformed $\mathbf{X}_2$

$$\begin{aligned} & \min_{\mathbf{w}_1,\mathbf{w}_2} || \mathbf{X}_1\mathbf{w}_1 - \mathbf{X}_2\mathbf{w}_2 ||_F \\ \text{such that} \quad & || \mathbf{X}_1\mathbf{w}_1 ||^2 = \mathbf{w}_1{}^t\mathbf{S}_{11}\mathbf{w}_1 = 1 \\ \text{and} \quad & || \mathbf{X}_2\mathbf{w}_2 ||^2 = \mathbf{w}_2{}^t\mathbf{S}_{22}\mathbf{w}_2 = 1 \end{aligned} \tag{A.4}$$

where $|| \cdot ||_F$ is the Frobenius norm, defined as $|| A ||_F = \sqrt{trace(A^tA)}$.

Using Equation A.3 we can derive the Lagrangian

$$L(\mathbf{w}_1, \mathbf{w}_2, \lambda_1, \lambda_2) = \mathbf{w}_1{}^t\mathbf{S}_{12}\mathbf{w}_2 - \lambda_1\mathbf{w}_1{}^t\mathbf{S}_{11}\mathbf{w}_1 - \lambda_2\mathbf{w}_2{}^t\mathbf{S}_{22}\mathbf{w}_2 \tag{A.5}$$

whose differentiation with respect to $\mathbf{w}_1$ and $\mathbf{w}_2$results in

$$\begin{aligned} & \begin{cases} \frac{\partial L}{\partial \mathbf{w}_1} = \mathbf{S}_{12}\mathbf{w}_2 - \lambda_1\mathbf{S}_{11}\mathbf{w}_1 = 0 \\ \frac{\partial L}{\partial \mathbf{w}_2} = \mathbf{S}_{21}\mathbf{w}_1 - \lambda_2\mathbf{S}_{22}\mathbf{w}_2 = 0 \end{cases} \\ \Longrightarrow & \begin{cases} \mathbf{S}_{12}\mathbf{w}_2 = \lambda_1\mathbf{S}_{11}\mathbf{w}_1 \\ \mathbf{S}_{21}\mathbf{w}_1 = \lambda_2\mathbf{S}_{22}\mathbf{w}_2 \end{cases} \end{aligned} \tag{A.6}$$

We can use Equation A.6 to show that $\lambda_1 = \lambda_2$ by observing

$$\lambda_1\mathbf{w}_1^t\mathbf{S}_{11}\mathbf{w}_1 = \mathbf{w}_1^t\mathbf{S}_{12}\mathbf{w}_2 = \mathbf{w}_2^t\mathbf{S}_{21}\mathbf{w}_1 = \lambda_2\mathbf{w}_2^t\mathbf{S}_{22}\mathbf{w}_2$$

and the constraints $\mathbf{w}_1^t\mathbf{S}_{11}\mathbf{w}_1 = \mathbf{w}_2^t\mathbf{S}_{22}\mathbf{w}_2 = 1$. Thus one can rewrite Equation A.6 as

$$\begin{cases} \mathbf{S}_{12}\mathbf{w}_2 = \lambda\mathbf{S}_{11}\mathbf{w}_1 \\ \mathbf{S}_{21}\mathbf{w}_1 = \lambda\mathbf{S}_{22}\mathbf{w}_2 \end{cases} \tag{A.7}$$

By defining

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{0} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{22} \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix}$$

one can rewrite A.7 as a *generalized eigenvalue* problem.

$$\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w} \tag{A.8}$$

This problem has $p_1 + p_2$ eigenvalues $\{\lambda_1, -\lambda_1, \lambda_2, -\lambda_2, \cdots, \lambda_p, -\lambda_p, 0, \cdots, 0\}$ where $p = min(p_1, p_2)$. The eigen vectors corresponding to the paired positive and negative eigenvalues differ only in a negative sign. These eigen vectors relate to each other as $[\mathbf{w}_1,\ \mathbf{w}_2]^t$ and $[\mathbf{w}_1,\ -\mathbf{w}_2]^t$. Thus only finding the set of positive eigenvalues is sufficient to fully solve the Equation A.8. Bach and Jordan [140] used a modified version of generalized eigenvalue problem A.8 that is a more suitable formulation to extend CCA for more than two views, *i.e.,* multiCCA. By adding $\mathbf{Bw}$ to both sides of A.8 they constructed a new generalized eigenvalue problem

$$\begin{aligned} \mathbf{Aw} + \mathbf{Bw} &= \lambda\mathbf{Bw} + \mathbf{Bw} \\ \implies (\mathbf{A} + \mathbf{B})\mathbf{w} &= (1 + \lambda)\mathbf{Bw} \\ \implies \mathbf{Cw} &= (1 + \lambda)\mathbf{Bw} \end{aligned} \tag{A.9}$$

where

$$\mathbf{C} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22} \end{bmatrix}$$

whose eigenvalues of this problem are$\{1 + \lambda_1, 1 - \lambda_1, 1 + \lambda_2, 1 - \lambda_2, \cdots, 1 + \lambda_p, 1 - \lambda_p, 1, \cdots, 1\}$. Based on the special relationship between $\pm\lambda_i$, Bach and Jordan [140] showed that finding the maximum eigenvalues of A.9, $1 + \lambda_{max}$, is equivalent to finding its minimum eigenvalue, $1 - \lambda_{max}$.

Properties

- Similar to other methods formulated as eigenvalue problems, CCA finds up to $p = min(p_1, p_2)$ paired projection vectors $(\mathbf{w}_{1,i}, \mathbf{w}_{2,i})$ and corresponding correlation values λ_i. One can construct two transform matrices such as

  $$\mathbf{W}_1 = [\mathbf{w}_{1,1}\mathbf{w}_{1,2} \ldots \mathbf{w}_{1,p}]$$
  $$\mathbf{W}_2 = [\mathbf{w}_{2,2}\mathbf{w}_{2,2} \ldots \mathbf{w}_{2,p}]$$

  Projecting data using these transform matrices decorrelates data. More specifically, transformed data, $\mathbf{Z}_1 = \mathbf{X}_1\mathbf{W}_1$ and $\mathbf{Z}_2 = \mathbf{X}_2\mathbf{W}_2$, covariance is

  $$\begin{aligned} \text{Cov}(\mathbf{Z}_1, \mathbf{Z}_2) &= \frac{1}{n}\begin{bmatrix} \mathbf{W}_1^t\mathbf{X}_1^t\mathbf{X}_1\mathbf{W}_1 & \mathbf{W}_1^t\mathbf{X}_1^t\mathbf{X}_2\mathbf{W}_2 \\ \mathbf{W}_2^t\mathbf{X}_2^t\mathbf{X}_1\mathbf{W}_1 & \mathbf{W}_2^t\mathbf{X}_2^t\mathbf{X}_2\mathbf{W}_2 \end{bmatrix} \\ &= \frac{1}{n}\begin{bmatrix} \mathbf{W}_1^t\mathbf{S}_{11}\mathbf{W}_1 & \mathbf{W}_1^t\mathbf{S}_{12}\mathbf{W}_2 \\ \mathbf{W}_2^t\mathbf{S}_{21}\mathbf{W}_1 & \mathbf{W}_2^t\mathbf{S}_{22}\mathbf{W}_2 \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{W}_1^t\mathbf{\Sigma}_{11}\mathbf{W}_1 & \mathbf{W}_1^t\mathbf{\Sigma}_{12}\mathbf{W}_2 \\ \mathbf{W}_2^t\mathbf{\Sigma}_{21}\mathbf{W}_1 & \mathbf{W}_2^t\mathbf{\Sigma}_{22}\mathbf{W}_2 \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I}_{p_1} & \mathbf{\Lambda} \\ \mathbf{\Lambda}^t & \mathbf{I}_{p_2} \end{bmatrix} \end{aligned}$$

where Λ is a $[p_1 \times p_2]$ diagonal matrix with $(\lambda_1, \ldots, \lambda_p)$ as diagonal values, sorted in the descending order $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$. Columns of transformation matrices $\mathbf{W}_1$ $\mathbf{W}_2$ are called **canonical vectors (weights)**. Columns of transformed data, $\mathbf{Z}_1$ and $\mathbf{Z}_2$, are called **canonical variates**. λ_i values are **canonical correlations**, correlation between pairs of canonical variates.

Note that transformation matrices $\mathbf{W}_i$ enforce the within view covariances to identity matrix $\mathbf{W}_i^t \mathbf{\Sigma}_{ii} \mathbf{W}_i = \mathbf{I}_{p_i}$, for $i = 1, 2$, while they make the cross covariance matrices to become diagonal $\mathbf{W}_1^t \mathbf{\Sigma}_{12} \mathbf{W}_2 = \mathbf{\Lambda}$.

- One can reduce principal component analysis (PCA), partial least square (PLS), and multiple linear regression (MLR) as a generalized eigenvalue problem too [5]. The following table summarizes the values of two matrices of $\mathbf{Aw} = \lambda \mathbf{Bw}$ for each algorithm

	A	B
PCA	$\mathbf{S}_{11}$	I
PLS	$\begin{pmatrix} \mathbf{0} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} I & \mathbf{0} \\ \mathbf{0} & I \end{pmatrix}$
CCA	$\begin{pmatrix} \mathbf{0} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{S}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{22} \end{pmatrix}$
MLR	$\begin{pmatrix} \mathbf{0} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{S}_{11} & \mathbf{0} \\ \mathbf{0} & I \end{pmatrix}$
$OPCA$	S	N

(A.10)

  Oriented principal component analysis (OPCA) is solving $\frac{w^t S w}{w^t N w}$ where S and N are signal and noise covariance matrices [135].

- Recall *least square regression (LSR)* answer is $\beta = (\mathbf{X}^t \mathbf{X})^{-1} \mathbf{X}^t \mathbf{y}$. Here if we assume that $\mathbf{X}_2 = \mathbf{y}$ and $w_1 = 1$ we can rewrite the first line of A.7 as

$$\begin{aligned} \mathbf{X}_1 \mathbf{y} &= \mathbf{X}_1^t \mathbf{X}_1 \mathbf{w}_1 \\ \implies \mathbf{w}_1 &= (\mathbf{X}_1^t \mathbf{X}_1)^{-1} \mathbf{X}_1^t \mathbf{y} \end{aligned}$$

  which shows that CCA reduced to least square regression when one of the data spaces is one dimensional.

- CCA correlation values are related to the mutual information between $\mathbf{X}_1$ and $\mathbf{X}_2$. Assuming Normal distribution for features in these two views, Tripathi

et al. [141] showed

$$
\begin{aligned}
I(\mathbf{X}_1, \mathbf{X}_2) &= -\frac{1}{2} log \left(\frac{\det(\mathbf{\Sigma})}{\det(\mathbf{\Sigma}_{11}) \det(\mathbf{\Sigma}_{22})} \right) \\
&= -\frac{1}{2} log \left(\frac{\det(\mathbf{S})}{\det(\mathbf{S}_{11}) \det(\mathbf{S}_{22})} \right) \\
&= -\frac{1}{2} log \left(\prod_{i=1}^{p} (1-\lambda_i)(1+\lambda_i) \right) \\
&= -\frac{1}{2} \sum_{i=1}^{p} log(1-\lambda_i^2)
\end{aligned} \tag{A.11}
$$

The ratio $\frac{\det(\mathbf{S})}{\det(\mathbf{S}_{11}) \det(\mathbf{S}_{22})}$ is called *generalized variance*. Note that maximum mutual information between a pair of one dimension linear projections of two views is equal to the largest term of the above sum $-\frac{1}{2} log(1-\lambda_1^2)$, which corresponds to the first canonical direction, *i.e.,* the highest canonical correlation value λ_1.

Other formulations of CCA

Here we show some other reformulations of CCA beside the generalized eigenvalue problem A.8. We start by assuming that $\mathbf{S}_{11}$ and $\mathbf{S}_{22}$ are nonsingular. One can reformulate the generalized eigenvalue problem A.8 as an ordinary eigenvalue problem

$$
\mathbf{B}^{-1}\mathbf{A}\mathbf{w} = \lambda \mathbf{w} \qquad \Longrightarrow \qquad \begin{bmatrix} \mathbf{0} & \mathbf{S}_{11}^{-1}\mathbf{S}_{12} \\ \mathbf{S}_{22}^{-1}\mathbf{S}_{21} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix} \tag{A.12}
$$

We can turn this into a symmetric eigenvalue problem using the substitution $\mathbf{u} = \mathbf{S}_{11}^{1/2}\mathbf{w}_1$ and $\mathbf{v} = \mathbf{S}_{22}^{1/2}\mathbf{w}_2$.

$$
\begin{bmatrix} \mathbf{0} & \mathbf{S}_{11}^{-1/2}\mathbf{S}_{12}\mathbf{S}_{22}^{-1/2} \\ \mathbf{S}_{22}^{-1/2}\mathbf{S}_{21}\mathbf{S}_{11}^{-1/2} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \tag{A.13}
$$

by using the relation between SVD transforms of a matrix, say A, and the eigenvectors of the matrix $\begin{bmatrix} 0 & A \\ A^t & 0 \end{bmatrix}$, one can verify A.13 is the PCA reformulation of SVD decomposition of off-diagonal component $\mathbf{S}_{11}^{-1/2}\mathbf{S}_{12}\mathbf{S}_{22}^{-1/2}$. Here $\mathbf{u}$ and $\mathbf{v}$ are right and left singular vectors. Thus one can reduce CCA problem to a SVD decomposition problem

$$
\mathcal{C} = \mathbf{S}_{11}^{-1/2}\mathbf{S}_{12}\mathbf{S}_{22}^{-1/2} = \mathbf{\Sigma}_{11}^{-1/2}\mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1/2} = \mathbf{U}\Lambda\mathbf{V}^t \tag{A.14}
$$

In this setting the CCA directions are

$$
\begin{cases} \mathbf{W}_1 = \mathbf{S}_{11}^{-1/2}\mathbf{U} \\ \mathbf{W}_2 = \mathbf{S}_{22}^{-1/2}\mathbf{V} \end{cases} \tag{A.15}
$$

using this formulation, we can define *reflexive generalized inverse* of CCA transformations.

$$\begin{cases} \mathbf{W}_1^* = \mathbf{U}^t \mathbf{S}_{11}^{1/2} \\ \mathbf{W}_2^* = \mathbf{V}^t \mathbf{S}_{22}^{1/2} \end{cases} \tag{A.16}$$

One can show these two matrices satisfy the two properties of reflexive generalized inverse of matrices, namely

$$\mathbf{W}_i^* \mathbf{W}_i \mathbf{W}_i^* = \mathbf{W}_i^* \qquad \text{and} \qquad \mathbf{W}_i \mathbf{W}_i^* \mathbf{W}_i = \mathbf{W}_i \quad \text{for} \quad i = 1, 2$$

By comparing A.15 and A.16, one can see the direct relationship

$$\begin{cases} \mathbf{W}_1^* = \mathbf{W}_1^t \mathbf{S}_{11} \\ \mathbf{W}_2^* = \mathbf{W}_2^t \mathbf{S}_{22} \end{cases}$$

The other way to solve the SVD problem in A.14 is to solve the PCA solution for $\mathcal{C}\mathcal{C}^t$ and $\mathcal{C}^t\mathcal{C}$ (based on Equation A.14. This will lead into a new formulation of CCA in which one can find pairs of canonical vectors as two independent eigenvalue problems

$$\mathcal{C}\mathcal{C}^t = \mathbf{S}_{11}^{-1/2} \mathbf{S}_{12} \mathbf{S}_{22}^{-1} \mathbf{S}_{21} \mathbf{S}_{11}^{-1/2} = \mathbf{U} \Lambda^2 \mathbf{U}^t$$

This PCA problem is solved by an ordinary eigenvalue problem

$$\mathbf{S}_{11}^{-1/2} \mathbf{S}_{12} \mathbf{S}_{22}^{-1} \mathbf{S}_{21} \mathbf{S}_{11}^{-1/2} \mathbf{u} = \lambda^2 \mathbf{u}$$

with the familiar change of variable $\mathbf{u} = \mathbf{S}_{11}^{1/2} \mathbf{w}_1$, we then have

$$\mathbf{S}_{11}^{-1/2} \mathbf{S}_{12} \mathbf{S}_{22}^{-1} \mathbf{S}_{21} \mathbf{w}_1 = \lambda^2 \mathbf{S}_{11}^{1/2} \mathbf{w}_1$$

We can then left multiply both side with $\mathbf{S}_{11}^{-1/2}$ to change this into an ordinary eigenvalue problem. One achieves a similar result for $\mathbf{w}_2$ by PCA analysis of $\mathcal{C}^t\mathcal{C}$.

$$\begin{cases} \mathbf{S}_{11}^{-1} \mathbf{S}_{12} \mathbf{S}_{22}^{-1} \mathbf{S}_{21} \ \mathbf{w}_1 = \lambda^2 \mathbf{w}_1 \\ \mathbf{S}_{22}^{-1} \mathbf{S}_{21} \mathbf{S}_{11}^{-1} \mathbf{S}_{12} \ \mathbf{w}_2 = \lambda^2 \mathbf{w}_2 \end{cases} \tag{A.17}$$

This is a new formulation to solve CCA. Note that in this case the eigenvalues are the square of the canonical correlation values.

Another way to look at CCA is by applying whitening transformation on both views $\mathbf{X}_1$ and $\mathbf{X}_2$ and applying PCA to the merged whitened SAMAN data set. First we show that whitening transformation does not affect the final transformed data under CCA. In the other words, if we represent the whiten views by $\bar{\mathbf{X}}_1 = \mathbf{X}_1 \mathbf{\Sigma}_{11}^{-1/2}$ and $\bar{\mathbf{X}}_2 = \mathbf{X}_2 \mathbf{\Sigma}_2^{-1/2}$ then we can show

$$\begin{cases} \bar{\mathbf{X}}_1 \bar{\mathbf{W}}_1 = \mathbf{X}_1 \mathbf{W}_1 \\ \bar{\mathbf{X}}_2 \bar{\mathbf{W}}_2 = \mathbf{X}_2 \mathbf{W}_2 \\ \bar{\mathbf{\Lambda}} = \mathbf{\Lambda} \end{cases} \tag{A.18}$$

To proof this we assume $\bar{\mathbf{X}} = [\bar{\mathbf{X}}_1\ \bar{\mathbf{X}}_2]$. Observe that

$$\text{Cov}(\bar{\mathbf{X}}) = \bar{\mathbf{\Sigma}} = \begin{bmatrix} \bar{\mathbf{\Sigma}}_{11} & \bar{\mathbf{\Sigma}}_{12} \\ \bar{\mathbf{\Sigma}}_{21} & \bar{\mathbf{\Sigma}}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{\Sigma}_{11}^{-1/2}\mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1/2} \\ \mathbf{\Sigma}_{22}^{-1/2}\mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1/2} & \mathbf{I} \end{bmatrix}$$

we use the A.14 formulation to solve the CCA problem for $\bar{\mathbf{X}}_1$ and $\bar{\mathbf{X}}_2$

$$\begin{aligned} \bar{\mathbf{\Sigma}}_{11}^{-1/2}\bar{\mathbf{\Sigma}}_{12}\bar{\mathbf{\Sigma}}_{22}^{-1/2} = &\quad \bar{\mathbf{U}}\bar{\Lambda}\mathbf{I}\bar{\mathbf{V}}^t = \\ \mathbf{I}\ \bar{\mathbf{\Sigma}}_{12}\ \mathbf{I} = & \\ \mathbf{\Sigma}_{11}^{-1/2}\mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1/2} = &\quad \mathbf{U}\Lambda\mathbf{V}^t \end{aligned} \tag{A.19}$$

thus we show $\bar{\Lambda} = \Lambda$. Using the left and right singular vectors of A.19 we can drive the CCA weights for whiten data sets

$$\begin{cases} \bar{\mathbf{W}}_1 = \bar{\mathbf{\Sigma}}_{11}^{-1/2}\bar{\mathbf{U}} = \mathbf{I}\ \bar{\mathbf{U}} = \mathbf{U} \\ \bar{\mathbf{W}}_2 = \bar{\mathbf{\Sigma}}_{22}^{-1/2}\bar{\mathbf{V}} = \mathbf{I}\ \bar{\mathbf{V}} = \mathbf{V} \end{cases}$$

and finally we can show

$$\begin{cases} \bar{\mathbf{X}}_1\bar{\mathbf{W}}_1 = \bar{\mathbf{X}}_1\mathbf{U} = \mathbf{X}_1\mathbf{\Sigma}_{11}^{-1/2}\mathbf{U} = \mathbf{X}_1\mathbf{W}_1 \\ \bar{\mathbf{X}}_2\bar{\mathbf{W}}_2 = \bar{\mathbf{X}}_2\mathbf{V} = \mathbf{X}_2\mathbf{\Sigma}_{22}^{-1/2}\mathbf{V} = \mathbf{X}_2\mathbf{W}_2 \end{cases}$$

There is also an interesting relation between the PCA transformation of whiten merged data $\bar{\mathbf{X}}$ and the CCA transformations of $\mathbf{X}_1$ and $\mathbf{X}_2$. Here we represent top $d \leq p_1 + p_2$ principal components of $\bar{\mathbf{X}}$ by $\bar{\mathbf{T}}_d$ and their corresponding eigenvalues by $\bar{\mathbf{D}}_d = diag(\alpha_1, \cdots, \alpha_d)$. By solving the following eigenvalue problem one can find these directions

$$\bar{\mathbf{\Sigma}}\ \bar{\mathbf{T}}_d = \bar{\mathbf{T}}_d\ \bar{\mathbf{D}} \tag{A.20}$$

be using the partitioned values of $\bar{\mathbf{\Sigma}}$ one can show

$$\begin{aligned} &\begin{bmatrix} \mathbf{I} & \mathbf{\Sigma}_{11}^{-1/2}\mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1/2} \\ \mathbf{\Sigma}_{22}^{-1/2}\mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1/2} & \mathbf{I} \end{bmatrix} \bar{\mathbf{T}}_d = \bar{\mathbf{T}}_d\ \bar{\mathbf{D}} \\ \Longrightarrow &\begin{bmatrix} \mathbf{0} & \mathbf{\Sigma}_{11}^{-1/2}\mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1/2} \\ \mathbf{\Sigma}_{22}^{-1/2}\mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1/2} & \mathbf{0} \end{bmatrix} \bar{\mathbf{T}}_d = \bar{\mathbf{T}}_d\ (\bar{\mathbf{D}} - \mathbf{I}) \end{aligned}$$

earlier in A.19 we looked at the SVD transform of off diagonal components. ,we can show that columns of $\bar{\mathbf{T}}_d$ are composed of left and right singular vectors of $\mathbf{\Sigma}_{11}^{-1/2}\mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1/2}$

$$\bar{\mathbf{T}}_d = \begin{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{v}_1 \end{bmatrix} & \cdots & \begin{bmatrix} \mathbf{u}_d \\ \mathbf{v}_d \end{bmatrix} \end{bmatrix}$$

and $\bar{\mathbf{D}} = \mathbf{\Lambda} + 1$. One might use the $\bar{\mathbf{T}}_d$ to reduce the dimensions of $\bar{\mathbf{X}}$ from $p_1 + p_2$ to d

$$
\begin{aligned}
\bar{\mathbf{X}}_d = \bar{\mathbf{X}}\,\bar{\mathbf{T}}_d &= [\bar{\mathbf{X}}_1\ \bar{\mathbf{X}}_2]\,\bar{\mathbf{T}}_d \\
&= [\mathbf{X}_1\mathbf{\Sigma}_{11}^{-1/2}\ \mathbf{X}_2\mathbf{\Sigma}_{22}^{-1/2}]\,\bar{\mathbf{T}}_d \\
&= [\mathbf{X}_1\mathbf{\Sigma}_{11}^{-1/2}\ \mathbf{X}_2\mathbf{\Sigma}_{22}^{-1/2}]\,\begin{bmatrix}\begin{bmatrix}\mathbf{u}_1\\ \mathbf{v}_1\end{bmatrix} & \cdots & \begin{bmatrix}\mathbf{u}_d\\ \mathbf{v}_d\end{bmatrix}\end{bmatrix} \\
&= [\mathbf{X}_1\ \mathbf{X}_2]\,\begin{bmatrix}\begin{bmatrix}\mathbf{\Sigma}_{11}^{-1/2}\mathbf{u}_1\\ \mathbf{\Sigma}_{22}^{-1/2}\mathbf{v}_1\end{bmatrix} & \cdots & \begin{bmatrix}\mathbf{\Sigma}_{11}^{-1/2}\mathbf{u}_d\\ \mathbf{\Sigma}_{22}^{-1/2}\mathbf{v}_d\end{bmatrix}\end{bmatrix} \\
&= [\mathbf{X}_1\ \mathbf{X}_2]\,\begin{bmatrix}\begin{bmatrix}\mathbf{w}_{11}\\ \mathbf{w}_{21}\end{bmatrix} & \cdots & \begin{bmatrix}\mathbf{w}_{1d}\\ \mathbf{w}_{2d}\end{bmatrix}\end{bmatrix} \\
&= \mathbf{X}_1\,\mathbf{W}_{1d}\ +\ \mathbf{X}_2\,\mathbf{W}_{2d}
\end{aligned}
$$

this means that if one whitens the two views of objects and then apply PCA to it to reduce the dimensionality to d the result will be the same as applying CCA to each view and adding the transformed views together. So far we have formulated CCA as

- Generalized eigenvalue problem A.8 and A.9
- Symmetric eigenvalue problem A.13
- SVD problem A.14
- Two independent ordinary eigenvalue problems A.17
- PCA problem of whiten views A.20

Note all these formulations are in primal space as opposed to the dual space. Also note all of them are based on within and across scatter matrix components A.2.

Probabilistic interpretation of CCA

All formulations of CCA so far were based on linear algebraic formulations. Bach and Jordan [142] proposed a **latent variable** interpretation of CCA problem in a probabilistic framework. In this setting CCA is studied under a generative model assumption. Here instead of studying $\mathbf{X}_1$ and $\mathbf{X}_1$ as two data sets with p_1 and p_2 features that are observerd for n objects, we look at them as two random vectors $\mathbf{x}_1 \in \mathbb{R}^{p_1}$ and $\mathbf{x}_2 \in \mathbb{R}^{p_2}$ each with n realizations as in the rows of $\mathbf{X}_1$ and $\mathbf{X}_2$. Note each of these $i = 1, \cdots, n$ observations are two views of a same object, *i.e.,* paired

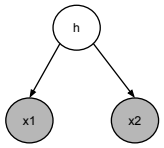


Figure A.1: Graphical model of relationship between random variables in CCA model.

observation $(\mathbf{x}_{1i}, \mathbf{x}_{2i})$. Bach and Jordan [142] proposed a generative model for CCA by assuming that $\mathbf{x}_1$ and $\mathbf{x}_2$ are independent given a latent random vector $\mathbf{h} \in \mathbb{R}^k$ where $1 \leq k \leq min(p_1, p_2)$. This relationship is shown in the simple graphical model of Figure A.1. They also assumed multi-variate normal distributions for all these three random variable.

$$\begin{cases} \mathbf{h} & \sim \mathcal{N}(0, I_k) \\ \mathbf{x}_1|\mathbf{h} & \sim \mathcal{N}(\mathbf{T}_1\mathbf{h} + \mu_1, \mathbf{\Phi}_1) \\ \mathbf{x}_2|\mathbf{h} & \sim \mathcal{N}(\mathbf{T}_2\mathbf{h} + \mu_2, \mathbf{\Phi}_2) \end{cases} \tag{A.21}$$

where $\mathbf{T}_1$ $_{[p_1 \times k]}$ and $\mathbf{T}_2$ $_{[p_2 \times k]}$ are transform matrices and $\Phi_1 \succeq 0$ and $\Phi_2 \succeq 0$. This conditional independence relationship between $\mathbf{x}_1$ and $\mathbf{x}_2$ imposes a special partitioned formate on marginal covariance of concatenated random variable $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix}$

$$\text{Cov}(\mathbf{x}) = \begin{bmatrix} \mathbf{T}_1\mathbf{T}_1^t + \mathbf{\Phi}_1 & \mathbf{T}_1\mathbf{T}_2^t \\ \mathbf{T}_2\mathbf{T}_1^t & \mathbf{T}_2\mathbf{T}_2^t + \mathbf{\Phi}_1 \end{bmatrix}$$

The maximum likelihood estimation for these parameters are:

$$\begin{aligned} \hat{\mathbf{T}}_1 &= \tilde{\mathbf{\Sigma}}_{11}\mathbf{W}_1\mathbf{\Lambda}^{1/2} \\ \hat{\mathbf{T}}_2 &= \tilde{\mathbf{\Sigma}}_{22}\mathbf{W}_2\mathbf{\Lambda}^{1/2} \\ \hat{\mathbf{\Phi}}_1 &= \tilde{\mathbf{\Sigma}}_{11} - \hat{\mathbf{T}}_1\hat{\mathbf{T}}_1^t \\ \hat{\mathbf{\Phi}}_2 &= \tilde{\mathbf{\Sigma}}_{22} - \hat{\mathbf{T}}_2\hat{\mathbf{T}}_2^t \\ \hat{\mu}_1 &= \tilde{\mu}_1 \\ \hat{\mu}_2 &= \tilde{\mu}_2 \end{aligned}$$

as one can see the transformation matrices $\hat{\mathbf{T}}_1$ and $\hat{\mathbf{T}}_2$ are directly related to the CCA transformations $\mathbf{W}_1$ and $\mathbf{W}_1$.

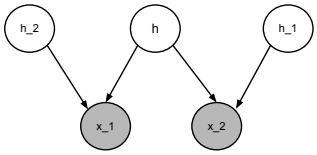


Figure A.2: Graphical model of relationship between random variables in CCA model with three latent variables.

under this model the posterior expectations of $\mathbf{h}$ given $\mathbf{x}_1$ and $\mathbf{x}_2$ is

$$\begin{aligned}
\mathrm{E}[\mathbf{h}|\mathbf{x}_1] &= \mathbf{\Lambda}^{1/2^t}\mathbf{W}_1^t(\mathbf{x}_1 - \mu_1) \\
\mathrm{E}[\mathbf{h}|\mathbf{x}_2] &= \mathbf{\Lambda}^{1/2^t}\mathbf{W}_2^t(\mathbf{x}_2 - \mu_2) \\
\mathrm{E}[\mathbf{h}|\mathbf{x}_1, \mathbf{x}_2] &= \begin{bmatrix} \mathbf{\Lambda}^{1/2}{}_t \\ \mathbf{\Lambda}^{1/2} \end{bmatrix} \begin{bmatrix} (\mathbf{I} - \mathbf{\Lambda}^2)^{-1} & (\mathbf{I} - \mathbf{\Lambda}^2)^{-1}\mathbf{\Lambda} \\ (\mathbf{I} - \mathbf{\Lambda}^2)^{-1}\mathbf{\Lambda} & (\mathbf{I} - \mathbf{\Lambda}^2)^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{W}_1^t(\mathbf{x}_1 - \mu_1) \\ \mathbf{W}_2^t(\mathbf{x}_2 - \mu_2) \end{bmatrix}
\end{aligned}$$

Klami and Kaski [6] showed that as long as we have a model with identical marginal covariance structure we will achieve the same relationship between maximum likelihood results and CCA projections. They formulated the maximum likelihood for the generative model in Figure A.2.

Multiple CCA

There are several ways to expand CCA formulation to more than two views. Here we assume $\mathbf{X}_i$ is a $[n \times p_i]$ matrix contains observations $\mathbf{x}_{ij} \in \mathbb{R}^{p_i}$ for $j = 1, \cdots, n$ same objects that are being observed in all $i = 1, 2, \cdots d$ views. Bach and Jordan [140] used the generalized eigenvalue problem A.9 to extend binary CCA to multiple CCA:

$$\begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \cdots & \mathbf{S}_{1d} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \cdots & \mathbf{S}_{2d} \\ \vdots & \vdots & & \vdots \\ \mathbf{S}_{d1} & \mathbf{S}_{d2} & \cdots & \mathbf{S}_{dd} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_d \end{bmatrix} = \alpha \begin{bmatrix} \mathbf{S}_{11} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{22} & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{S}_{dd} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_d \end{bmatrix} \tag{A.22}$$

they showed the relationship between mutual information and eigenvalues of A.22 still exist:

$$\begin{aligned}
I(\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_d) &= -\frac{1}{2} log \left(\frac{\det(\mathbf{\Sigma})}{\det(\mathbf{\Sigma}_{11}) \det(\mathbf{\Sigma}_{22}) \cdots \det(\mathbf{\Sigma}_{dd})} \right) \\
&= -\frac{1}{2} log \left(\frac{\det(\mathbf{S})}{\det(\mathbf{S}_{11}) \det(\mathbf{S}_{22}) \cdots \det(\mathbf{S}_{dd})} \right) \\
&= -\frac{1}{2} \sum_{i=1}^{P} log(\alpha_i)
\end{aligned} \tag{A.23}$$

where $P = \sum_{i=1}^{d} p_i$ and α_i are eigenvalues of generalized eigenvalue problem A.22. Similar to two view CCA, if we want the maximum canonical correlation, we should find the largest positive term in sum A.23 which corresponds to the smallest eigenvalue of A.22, here we represent as α_1.

There are some differences between multiple CCA and binary CCA. In the case of multiple CCA, eigenvalues are not paired as in two views case A.9. Also in two views case the canonical correlation values are related to eigenvalues of A.9 $\lambda_i = \alpha_i - 1$ which is equal to the correlation of transformed views $\lambda_i = \text{Cor}(\mathbf{X}_1\mathbf{w}_{1i}, \mathbf{X}_2\mathbf{w}_{2i})$. In the case of multiple CCA, instead of one correlation value, we have a $d \times d$ correlation matrix for each one-dimensional projection of views $\mathbf{X}_1\mathbf{w}_{1i}, \mathbf{X}_2\mathbf{w}_{2i}, \cdots, \mathbf{X}_d\mathbf{w}_{di}$. Here we represent the correlation of one dimensional projections by $\tilde{C}_i = \text{Cor}(\mathbf{X}_1\mathbf{w}_{1i}, \mathbf{X}_2\mathbf{w}_{2i}, \cdots, \mathbf{X}_d\mathbf{w}_{di})$. Note that $\tilde{C}_i$ being the correlation matrix, is a symmetric positive semidefinite with trace of d. This means its eigenvalues are non-negative and sum up to d. The minimum eigenvalue is in $[0, 1]$ range. If the minimum eigenvalue is 1 then all eigenvalues are 1 and $\tilde{C}_i = \mathbf{I}_d$. This means one dimensional projections are uncorrelated. Here we represent the minimum eigenvalue of correlation matrix of one dimensional projections by $\nu(\mathbf{X}_1\mathbf{w}_{1i}, \mathbf{X}_2\mathbf{w}_{2i}, \cdots, \mathbf{X}_d\mathbf{w}_{di})$. Bach and Jordan [140] proved the minimum eigenvalue of generalized problem A.22 is equal to the minimum eigenvalue of correlation matrix of all possible one-dimensional projection of views

$$\alpha_1 = \min_{\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_d} \nu(\mathbf{X}_1\mathbf{t}_1, \mathbf{X}_2\mathbf{t}_2, \cdots, \mathbf{X}_d\mathbf{t}_d) \tag{A.24}$$

they also proved that $\alpha_1 = 1$ if and only if $\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_d$ are uncorrelated.

List of Abbreviations

AGC	Array Generation gene Centering
BECCA	Batch Effects Correction using CCA
BFRM	Bayesian Factor Regression Modeling
CAT	Correspondence At The Top
CCA	Canonical Correlation Analysis
CFD	Chip Description Files
CNV	copy-number variations
ComBat	Combining Batches
CV	Cross Validation
DC	Detection call
DE	Differentially Expressed
DWD	Distance Weighted Discrimination
EB	Empirical Bayes
FA	Factor Analysis
FLEO	Feature Level Extraction Output
GAGE	Generally Applicable Gene set Enrichment
GEDM	Gene Expression Data Matrix
GEO	Gene Expression Omnibus
GLAPA	Gene List Analysis with Prediction Accuracy

GSA	Gene Set Analysis
GSEA	Gene Set Enrichment Analysis
GSFLD	Gene Shaving based on Fisher Linear Discrimination
GSRF	Gene Shaving based on Random Forest
i.i.d.	independent and identically distributed
IQR	Inter Quantile Range
k-TSP	k Top Scored Pairs
LOO	Leave One Out
LTR	Linear Transformation of Replicates
MAD	Median Absolute Deviation
mDEDS	Meta differential expression via distance synthesis
MDS	Multidimensional Scaling
METRADISC	METa-analysis of RAnked DISCovery
miRNA	microRNA
mRNA	messenger RNA
MRS	Median Rank Score
NORDI	Normal Discretization
PAGE	Parametric Analysis of Gene Set Enrichment
PAM	Predictive Analysis of Microarrays
PCA	Principal Component Analysis
POE	Probability of Expression
QD	Quantile Discretization
RUV	Remove Unwanted Variation, 2-step
SAFE	Significance Analysis of Function and Expression

SAM	Significance Analysis of Microarrays
SAM-GS	Significance Analysis of Microarray (SAM) to gene-set
SAM-GSR	Significance Analysis of Microarray for Gene-Set Reduction
SNP	single-nucleotide polymorphism
SVA	Surrogate Variable Analysis
XPN	Cross Platform Normalization

Bibliography

[1] Adaikalavan Ramasamy, Adrian Mondry, Chris C Holmes, and Douglas G Altman. Key issues in conducting a meta-analysis of gene expression microarray datasets. *PLoS medicine*, 5(9):e184, 2008.

[2] Cosmin Lazar, Stijn Meganck, Jonatan Taminau, David Steenhoff, Alain Coletta, Colin Molter, David Y Weiss-Solís, Robin Duque, Hugues Bersini, and Ann Nowé. Batch effect removal methods for microarray gene expression data integration: a survey. *Briefings in Bioinformatics*, 2012.

[3] Jeffrey T Leek, Robert B Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews Genetics*, 11(10):733–739, 2010.

[4] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4): 321–377, 1936.

[5] Magnus Borga. Canonical correlation: a tutorial. *On line tutorial http://people. imt. liu. se/magnus/cca*, 2001.

[6] Arto Klami and Samuel Kaski. Generative models that discover dependencies between data sets. pages 123–128, 2006.

[7] Tijl De Bie, Nello Cristianini, and Roman Rosipal. Eigenproblems in pattern recognition. In *Handbook of Geometric Computing*, pages 129–167. Springer, 2005.

[8] Cosmin Lazar, Jonatan Taminau, Stijn Meganck, David Steenhoff, Alain Coletta, Colin Molter, Virginie de Schaetzen, Robin Duque, Hugues Bersini, and Ann Nowé. A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 9(4):1106–1119, 2012.

[9] Johann A Gagnon-Bartsch and Terence P Speed. Using control genes to correct for unwanted variation in microarray data. *Biostatistics*, 13(3):539–552, 2012.

[10] Ahmedin Jemal, Rebecca Siegel, Jiaquan Xu, and Elizabeth Ward. Cancer statistics, 2010. *CA: a cancer journal for clinicians*, 60(5):277–300, 2010.

[11] Charles M Perou, Therese Sørlie, Michael B Eisen, Matt van de Rijn, Stefanie S Jeffrey, Christian A Rees, Jonathan R Pollack, Douglas T Ross, Hilde Johnsen, Lars A Akslen, et al. Molecular portraits of human breast tumours. *Nature*, 406(6797):747–752, 2000.

[12] Martin H van Vliet, Christiaan N Klijn, Lodewyk FA Wessels, and Marcel JT Reinders. Module-based outcome prediction using breast cancer compendia. *PLoS One*, 2(10):e1047, 2007.

[13] Michael L Gatza, Joseph E Lucas, William T Barry, Jong Wook Kim, Quanli Wang, Matthew D Crawford, Michael B Datto, Michael Kelley, Bernard Mathey-Prevot, Anil Potti, et al. A pathway-based classification of human breast cancer. *Proceedings of the National Academy of Sciences*, 107(15): 6994–6999, 2010.

[14] S.M. Tu. Heterogeneity of cancer. *Origin of Cancers*, pages 129–136, 2010.

[15] Adel Tabchy, Bryan T Hennessy, Gabriel Hortobagyi, and Gordon B Mills. Systems biology of breast cancer. *Current Breast Cancer Reports*, 1(4):238–245, 2009.

[16] Sofia K Gruvberger-Saal, Heather E Cunliffe, Kristen M Carr, and Ingrid A Hedenfalk. Microarrays in breast cancer research and clinical practice–the future lies ahead. *Endocrine-related cancer*, 13(4):1017–1031, 2006.

[17] Carole L Yauk, M Lynn Berndt, Andrew Williams, and George R Douglas. Comprehensive comparison of six microarray technologies. *Nucleic Acids Research*, 32(15):e124–e124, 2004.

[18] Nancy Mah, Anders Thelin, Tim Lu, Susanna Nikolaus, Tanja Kühbacher, Yesim Gurbuz, Holger Eickhoff, Günther Klöppel, Hans Lehrach, Björn Mellgård, et al. A comparison of oligonucleotide and cdna-based microarray systems. *Physiological genomics*, 16(3):361–370, 2004.

[19] Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin. *The elements of statistical learning: data mining, inference and prediction*, volume 27. Springer, 2005.

[20] Liangjiang Wang, Anand Srivastava, and Charles Schwartz. Microarray data integration for genome-wide analysis of human tissue-selective gene expression. *BMC genomics*, 11(Suppl 2):S15, 2010.

[21] Ron Edgar, Michael Domrachev, and Alex E Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic acids research*, 30(1):207–210, 2002.

[22] Helen Parkinson, Misha Kapushesky, Nikolay Kolesnikov, Gabriella Rustici, Mohammad Shojatalab, Niran Abeygunawardena, Hugo Berube, Miroslaw Dylag, Ibrahim Emam, Anna Farne, et al. Arrayexpress updatefrom an archive of functional genomics experiments to the atlas of gene expression. *Nucleic acids research*, 37(suppl 1):D868–D872, 2009.

[23] Jack Zhu and Sean Davis. *GEOmetadb: A compilation of metadata from NCBI GEO*, 2011. URL http://gbnci.abcc.ncifcrf.gov/geo/. R package version 1.20.0.

[24] R.A. Irizarry, B. Hobbs, F. Collin, Y.D. Beazer-Barclay, K.J. Antonellis, U. Scherf, T.P. Speed, et al. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, 2003.

[25] F. Hahne, W. Huber, R. Gentleman, and S. Falcon. *Bioconductor case studies*. Springer Verlag, 2008.

[26] Yosef Prat, Menachem Fromer, Nathan Linial, and Michal Linial. Recovering key biological constituents through sparse representation of gene expression. *Bioinformatics*, 27(5):655–661, 2011.

[27] Daniel Marbach, Robert J Prill, Thomas Schaffter, Claudio Mattiussi, Dario Floreano, and Gustavo Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286–6291, 2010.

[28] Elisa T Lee and John Wang. *Statistical methods for survival data analysis*, volume 476. Wiley. com, 2003.

[29] Marit Ackermann and Korbinian Strimmer. A general modular framework for gene set enrichment analysis. *BMC bioinformatics*, 10(1):47, 2009.

[30] Luca Abatangelo, Rosalia Maglietta, Angela Distaso, Annarita D'Addabbo, Teresa M Creanza, Sayan Mukherjee, and Nicola Ancona. Comparative study of gene set enrichment methods. *BMC bioinformatics*, 10(1):275, 2009.

[31] Weijun Luo, Michael Friedman, Kerby Shedden, Kurt Hankenson, and Peter Woolf. Gage: generally applicable gene set enrichment for pathway analysis. *BMC bioinformatics*, 10(1):161, 2009.

[32] Fátima Al-Shahrour, Leonardo Arbiza, Hernán Dopazo, Jaime Huerta-Cepas, Pablo Mínguez, David Montaner, and Joaquín Dopazo. From genes to functional classes in the study of biological systems. *BMC bioinformatics*, 8(1): 114, 2007.

[33] John D Storey and Robert Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences*, 100(16):9440–9445, 2003.

[34] Virginia Goss Tusher, Robert Tibshirani, and Gilbert Chu. Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences*, 98(9):5116–5121, 2001.

[35] Hongying Jiang, Youping Deng, Huann-Sheng Chen, Lin Tao, Qiuying Sha, Jun Chen, Chung-Jui Tsai, and Shuanglin Zhang. Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes. *BMC bioinformatics*, 5(1):81, 2004.

[36] Adeniyi J Adewale, Irina Dinu, John D Potter, Qi Liu, and Yutaka Yasui. Pathway analysis of microarray data via regression. *Journal of Computational Biology*, 15(3):269–277, 2008.

[37] Aravind Subramanian, Pablo Tamayo, Vamsi K Mootha, Sayan Mukherjee, Benjamin L Ebert, Michael A Gillette, Amanda Paulovich, Scott L Pomeroy, Todd R Golub, Eric S Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, 2005.

[38] Robert Tibshirani, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences*, 99(10):6567–6572, 2002.

[39] Lei Xu, Aik C Tan, Raimond L Winslow, and Donald Geman. Merging microarray data from separate breast cancer studies provides a robust prognostic test. *BMC bioinformatics*, 9(1):125, 2008.

[40] Larry Wasserman. *All of statistics: a concise course in statistical inference*. Springer, 2004.

[41] Jelle J Goeman and Peter Bühlmann. Analyzing gene expression data in terms of gene sets: methodological issues. *Bioinformatics*, 23(8):980–987, 2007.

[42] Jelle J Goeman, Sara A Van De Geer, Floor De Kort, and Hans C Van Houwelingen. A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, 20(1):93–99, 2004.

[43] Jelle J Goeman, Jan Oosting, Anne-Marie Cleton-Jansen, Jakob K Anninga, and Hans C Van Houwelingen. Testing association of a pathway with survival using gene expression data. *Bioinformatics*, 21(9):1950–1957, 2005.

[44] Irina Dinu, John D Potter, Thomas Mueller, Qi Liu, Adeniyi J Adewale, Gian S Jhangri, Gunilla Einecke, Konrad S Famulski, Philip Halloran, and Yutaka Yasui. Improving gene set analysis of microarray data by sam-gs. *BMC bioinformatics*, 8(1):242, 2007.

[45] Irina Dinu, John D Potter, Thomas Mueller, Qi Liu, Adeniyi J Adewale, Gian S Jhangri, Gunilla Einecke, Konrad S Famulski, Philip Halloran, and Yutaka Yasui. Gene-set analysis and reduction. *Briefings in bioinformatics*, 10(1):24–34, 2009.

[46] Liat Ein-Dor, Itai Kela, Gad Getz, David Givol, and Eytan Domany. Outcome signature genes in breast cancer: is there a unique set? *Bioinformatics*, 21 (2):171–178, 2005.

[47] Laura J van't Veer, Hongyue Dai, Marc J Van De Vijver, Yudong D He, Augustinus AM Hart, Mao Mao, Hans L Peterse, Karin van der Kooy, Matthew J Marton, Anke T Witteveen, et al. Gene expression profiling predicts clinical outcome of breast cancer. *nature*, 415(6871):530–536, 2002.

[48] Andrew H Sims, Kai Ren Ong, Robert B Clarke, and Anthony Howell. Exploiting the potential of gene expression profiling: Is it ready for the clinic. *Breast Cancer Res*, 8(5):214–220, 2006.

[49] Liat Ein-Dor, Or Zuk, and Eytan Domany. Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *Proceedings of the National Academy of Sciences*, 103(15):5923–5928, 2006.

[50] Therese Sørlie, Robert Tibshirani, Joel Parker, Trevor Hastie, JS Marron, Andrew Nobel, Shibing Deng, Hilde Johnsen, Robert Pesich, Stephanie Geisler, et al. Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proceedings of the National Academy of Sciences*, 100 (14):8418–8423, 2003.

[51] Anja von Heydebreck, Wolfgang Huber, and Robert Gentleman. Differential expression with the bioconductor project. 2004.

[52] Sofia Gruvberger, Markus Ringnér, Yidong Chen, Sujatha Panavally, Lao H Saal, Åke Borg, Mårten Fernö, Carsten Peterson, and Paul S Meltzer. Estrogen receptor status in breast cancer is associated with remarkably distinct gene expression patterns. *Cancer research*, 61(16):5979–5984, 2001.

[53] Alain Dupuy and Richard M Simon. Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. *Journal of the National Cancer Institute*, 99(2):147–157, 2007.

[54] Sami Kilpinen, Reija Autio, Kalle Ojala, Kristiina Iljin, Elmar Bucher, Henri Sara, Tommi Pisto, Matti Saarela, Rolf I Skotheim, Mari Bjorkman, et al. Systematic bioinformatic analysis of expression levels of 17,330 human genes across 9,783 samples from 175 types of healthy and pathological tissues. *Genome Biol*, 9(9):R139, 2008.

[55] Greg Finak, Nicholas Bertos, Francois Pepin, Svetlana Sadekova, Margarita Souleimanova, Hong Zhao, Haiying Chen, Gulbeyaz Omeroglu, Sarkis Meterissian, Atilla Omeroglu, et al. Stromal gene expression predicts clinical outcome in breast cancer. *Nature medicine*, 14(5):518–527, 2008.

[56] Vlad Popovici, Weijie Chen, Brandon G Gallas, Christos Hatzis, Weiwei Shi, Frank W Samuelson, Yuri Nikolsky, Marina Tsyganova, Alex Ishkin, Tatiana Nikolskaya, et al. Effect of training-sample size and classification difficulty on the accuracy of genomic predictors. *Breast Cancer Res*, 12(1):R5, 2010.

[57] Elizabeth Garrett-Mayer, Giovanni Parmigiani, Xiaogang Zhong, Leslie Cope, and Edward Gabrielson. Cross-study validation and combined analysis of gene expression microarray data. *Biostatistics*, 9(2):333–354, 2008.

[58] Andreas Scherer. *Batch effects and noise in microarray experiments: sources and solutions*, volume 868. Wiley. com, 2009.

[59] Shuangge Ma. Integrative analysis of cancer genomic data. 2009.

[60] Andrey A Shabalin, Håkon Tjelmeland, Cheng Fan, Charles M Perou, and Andrew B Nobel. Merging two gene-expression studies via cross-platform normalization. *Bioinformatics*, 24(9):1154–1160, 2008.

[61] Anna Campain and Yee H Yang. Comparison study of microarray meta-analysis methods. *BMC bioinformatics*, 11(1):408, 2010.

[62] Reija Autio, Sami Kilpinen, Matti Saarela, Olli Kallioniemi, Sampsa Hautaniemi, and Jaakko Astola. Comparison of affymetrix data normalization methods using 6,926 experiments across five array generations. *BMC bioinformatics*, 10(Suppl 1):S24, 2009.

[63] Patrick Warnat, Roland Eils, and Benedikt Brors. Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes. *BMC bioinformatics*, 6(1):265, 2005.

[64] Jung Kyoon Choi, Ungsik Yu, Sangsoo Kim, and Ook Joon Yoo. Combining multiple microarray studies and modeling interstudy variation. *Bioinformatics*, 19(suppl 1):i84–i90, 2003.

[65] Giovanni Parmigiani, Elizabeth S Garrett, Ramaswamy Anbazhagan, and Edward Gabrielson. A statistical framework for expression-based molecular classification in cancer. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):717–736, 2002.

[66] Ricardo Martinez, Claude Pasquier, and Nicolas Pasquier. Genminer: mining informative association rules from genomic data. pages 15–22, 2007.

[67] JS Marron, Michael J Todd, and Jeongyoun Ahn. Distance-weighted discrimination. *Journal of the American Statistical Association*, 102(480):1267–1271, 2007.

[68] Paul C Boutros. Ltr: Linear cross-platform integration of microarray data. *Cancer informatics*, 9:197, 2010.

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

[70] Jeffrey T Leek and John D Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, 3(9):e161, 2007.

[71] Carlos M Carvalho, Jeffrey Chang, Joseph E Lucas, Joseph R Nevins, Quanli Wang, and Mike West. High-dimensional sparse factor modeling: applications in gene expression genomics. *Journal of the American Statistical Association*, 103(484), 2008.

[72] Xin Victoria Wang, Roel GW Verhaak, Elizabeth Purdom, Paul T Spellman, and Terence P Speed. Unifying gene expression measures from multiple platforms using factor analysis. *PloS one*, 6(3):e17691, 2011.

[73] Daniel R Rhodes, Jianjun Yu, K Shanker, Nandan Deshpande, Radhika Varambally, Debashis Ghosh, Terrence Barrette, Akhilesh Pandey, and Arul M Chinnaiyan. Large-scale meta-analysis of cancer microarray data identifies common transcriptional profiles of neoplastic transformation and progression. *Proceedings of the National Academy of Sciences of the United States of America*, 101(25):9309–9314, 2004.

[74] Babak Damavandi, Chun-Nam Yu, Sambasivarao Damaraju, and Russell Greiner. Explaining the gene signature anomaly: Estimating the overlap of two ranked lists. *Breast Cancer*, 44(45):70–76, 2012.

[75] Donna Maglott, Jim Ostell, Kim D Pruitt, and Tatiana Tatusova. Entrez gene: gene-centered information at ncbi. *Nucleic acids research*, 39(suppl 1): D52–D57, 2011.

[76] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[77] Lei Xu, Aik Choon Tan, Daniel Q Naiman, Donald Geman, and Raimond L Winslow. Robust prostate cancer marker genes emerge from direct integration of inter-study microarray data. *Bioinformatics*, 21(20):3905–3911, 2005.

[78] Jonatan Taminau, Stijn Meganck, Cosmin Lazar, David Steenhoff, Alain Coletta, Colin Molter, Robin Duque, Virginie de Schaetzen, David Y Weiss Solís, Hugues Bersini, et al. Unlocking the potential of publicly available microarray data using insilicodb and insilicomerging r/bioconductor packages. *BMC bioinformatics*, 13(1):335, 2012.

[79] Wei-Chung Cheng, Min-Lung Tsai, Cheng-Wei Chang, Ching-Lung Huang, Chaang-Ray Chen, Wun-Yi Shu, Yun-Shien Lee, Tzu-Hao Wang, Ji-Hong Hong, Chia-Yang Li, et al. Microarray meta-analysis database (m2db): a uniformly pre-processed, quality controlled, and manually curated human clinical microarray database. *BMC bioinformatics*, 11(1):421, 2010.

[80] Enrico Glaab, Jonathan Garibaldi, and Natalio Krasnogor. Arraymining: a modular web-application for microarray analysis combining ensemble and consensus methods with cross-study normalization. *BMC Bioinformatics*, 10(1): 358, 2009.

[81] Ramil N Nurtdinov, Mikhail O Vasiliev, Anna S Ershova, Ilia S Lossev, and Anna S Karyagina. Plandbaffy: probe-level annotation database for affymetrix expression microarrays. *Nucleic acids research*, 38(suppl 1):D726–D730, 2010.

[82] Manhong Dai, Pinglang Wang, Andrew D Boyd, Georgi Kostov, Brian Athey, Edward G Jones, William E Bunney, Richard M Myers, Terry P Speed, Huda Akil, et al. Evolving gene/transcript definitions significantly alter the interpretation of genechip data. *Nucleic acids research*, 33(20):e175–e175, 2005.

[83] Rickard Sandberg and Ola Larsson. Improved precision and accuracy for microarrays using updated probe set definitions. *BMC bioinformatics*, 8(1): 48, 2007.

[84] Stefan Michiels, Serge Koscielny, and Catherine Hill. Prediction of cancer outcome with microarrays: a multiple random validation strategy. *The Lancet*, 365(9458):488–492, 2005.

[85] Daniel R Rhodes and Arul M Chinnaiyan. Integrative analysis of the cancer transcriptome. *Nature genetics*, 37:S31–S37, 2005.

[86] Andrew H Sims, Graeme J Smethurst, Yvonne Hey, Michal J Okoniewski, Stuart D Pepper, Anthony Howell, Crispin J Miller, and Robert B Clarke. The removal of multiplicative, systematic bias allows integration of breast cancer gene expression datasets–improving meta-analysis and prediction of prognosis. *BMC medical genomics*, 1(1):42, 2008.

[87] Robert Kitchen, Vicky Sabine, Andrew Sims, E Jane Macaskill, Lorna Renshaw, Jeremy Thomas, Jano van Hemert, J Michael Dixon, and John Bartlett. Correcting for intra-experiment variation in illumina beadchip data is necessary to generate robust gene-expression profiles. *BMC Genomics*, 11(1):134, 2010.

[88] Robert R Kitchen, Vicky S Sabine, Arthur A Simen, J Michael Dixon, John MS Bartlett, and Andrew H Sims. Relative impact of key sources of systematic noise in affymetrix and illumina gene-expression microarray experiments. *BMC genomics*, 12(1):589, 2011.

[89] Laurent Jacob, Johann Gagnon-Bartsch, and Terence P Speed. Correcting gene expression data when neither the unwanted variation nor the factor of interest are observed. *arXiv preprint arXiv:1211.4259*, 2012.

[90] Rafael A Irizarry, Daniel Warren, Forrest Spencer, Irene F Kim, Shyam Biswal, Bryan C Frank, Edward Gabrielson, Joe GN Garcia, Joel Geoghegan, Gregory Germino, et al. Multiple-laboratory comparison of microarray platforms. *Nature methods*, 2(5):345–350, 2005.

[91] Matthew N McCall and Rafael A Irizarry. Consolidated strategy for the analysis of microarray spike-in data. *Nucleic acids research*, 36(17):e108–e108, 2008.

[92] BK Weis. Standardizing global gene expression analysis between laboratories and across platforms. *Nature methods*, 2(5):351–356, 2005.

[93] Hyuna Yang, Christina A Harrington, Kristina Vartanian, Christopher D Coldren, Rob Hall, and Gary A Churchill. Randomization in laboratory procedure is key to obtaining reproducible microarray results. *PLoS One*, 3(11):e3724, 2008.

[94] Earl Hubbell, Wei-Min Liu, and Rui Mei. Robust estimators for expression analysis. *Bioinformatics*, 18(12):1585–1592, 2002.

[95] Pablo Tamayo, Daniel Scanfeld, Benjamin L Ebert, Michael A Gillette, Charles WM Roberts, and Jill P Mesirov. Metagene projection for cross-platform, cross-species characterization of global transcriptional states. volume 104, pages 5959–5964. National Acad Sciences, 2007.

[96] Dan Nettleton. A discussion of statistical methods for design and analysis of microarray experiments for plant scientists. *The Plant Cell Online*, 18(9): 2112–2121, 2006.

[97] Christophe G Lambert and Laura J Black. Learning from our gwas mistakes: from experimental design to scientific method. *Biostatistics*, 13(2):195–203, 2012.

[98] Hyuna Yang, Christina A Harrington, Kristina Vartanian, Christopher D Coldren, Rob Hall, and Gary A Churchill. Randomization in laboratory procedure is key to obtaining reproducible microarray results. *PLoS One*, 3(11):e3724, 2008.

[99] C Kendziorski, RA Irizarry, K-S Chen, JD Haag, and MN Gould. On the utility of pooling biological samples in microarray experiments. *Proceedings of the National Academy of Sciences of the United States of America*, 102(12): 4252–4257, 2005.

[100] William S Branham, Cathy D Melvin, Tao Han, Varsha G Desai, Carrie L Moland, Adam T Scully, and James C Fuscoe. Elimination of laboratory ozone leads to a dramatic improvement in the reproducibility of microarray gene expression measurements. *BMC biotechnology*, 7(1):8, 2007.

[101] Douglas C Montgomery, Douglas C Montgomery, and Douglas C Montgomery. *Design and analysis of experiments*, volume 7. Wiley New York, 1984.

[102] Kun Yang, Jianzhong Li, and Hong Gao. The impact of sample imbalance on identifying differentially expressed genes. *BMC Bioinformatics*, 7(Suppl 4): S8, 2006.

[103] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.

[104] Andrey A Shabalin, Håkon Tjelmeland, Cheng Fan, Charles M Perou, and Andrew B Nobel. Merging two gene-expression studies via cross-platform normalization. *Bioinformatics*, 24(9):1154–1160, 2008.

[105] Leslie M Cope, Liz Garrett-Mayer, Edward Gabrielson, and Giovanni Parmigiani. The integrative correlation coefficient: A measure of cross-study reproducibility for gene expressionea array data. 2007.

[106] Rafael A Irizarry, Bridget Hobbs, Francois Collin, Yasmin D Beazer-Barclay, Kristen J Antonellis, Uwe Scherf, and Terence P Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, 2003.

[107] Tomas Bonome, Douglas A Levine, Joanna Shih, Mike Randonovich, Cindy A Pise-Masison, Faina Bogomolniy, Laurent Ozbun, John Brady, J Carl Barrett, Jeff Boyd, et al. A gene signature predicting for survival in suboptimally debulked patients with ovarian cancer. *Cancer research*, 68(13):5478–5486, 2008.

[108] Zhenyu Jia, Yipeng Wang, Anne Sawyers, Huazhen Yao, Farahnaz Rahmatpanah, Xiao-Qin Xia, Qiang Xu, Rebecca Pio, Tolga Turan, James A Koziol, et al. Diagnosis of prostate cancer using differentially expressed genes in stroma. *Cancer research*, 71(7):2476–2487, 2011.

[109] Maria Teresa Landi, Tatiana Dracheva, Melissa Rotunno, Jonine D Figueroa, Huaitian Liu, Abhijit Dasgupta, Felecia E Mann, Junya Fukuoka, Megan Hames, Andrew W Bergen, et al. Gene expression signature of cigarette smoking and its role in lung adenocarcinoma development and survival. *PLoS One*, 3(2):e1651, 2008.

[110] Yixin Wang, Jan GM Klijn, Yi Zhang, Anieta M Sieuwerts, Maxime P Look, Fei Yang, Dmitri Talantov, Mieke Timmermans, Marion E Meijer-van Gelder, Jack Yu, et al. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *The Lancet*, 365(9460):671–679, 2005.

[111] Christine Desmedt, Fanny Piette, Sherene Loi, Yixin Wang, Françoise Lallemand, Benjamin Haibe-Kains, Giuseppe Viale, Mauro Delorenzi, Yi Zhang, Mahasti Saghatchian d'Assignies, et al. Strong time dependence of the 76-gene prognostic signature for node-negative breast cancer patients in the transbig multicenter independent validation series. *Clinical cancer research*, 13(11): 3207–3214, 2007.

[112] Stuart D Pepper, Emma K Saunders, Laura E Edwards, Claire L Wilson, and Crispin J Miller. The utility of mas5 expression summary and detection call algorithms. *BMC bioinformatics*, 8(1):273, 2007.

[113] Kellie J Archer and Sarah E Reese. Detection call algorithms for high-throughput gene expression microarray data. *Briefings in bioinformatics*, 11 (2):244–252, 2010.

[114] Jeanette N McClintick and Howard J Edenberg. Effects of filtering by present call on analysis of microarray experiments. *BMC bioinformatics*, 7(1):49, 2006.

[115] Richard Shippy, Timothy Sendera, Randall Lockner, Chockalingam Palaniappan, Tamma Kaysser-Kranich, George Watts, and John Alsobrook. Performance evaluation of commercial short-oligonucleotide microarrays and the impact of noise in making cross-platform correlations. *BMC genomics*, 5(1):61, 2004.

[116] Elizabeth Garrett-Mayer, Giovanni Parmigiani, Xiaogang Zhong, Leslie Cope, and Edward Gabrielson. Cross-study validation and combined analysis of gene expression microarray data. *Biostatistics*, 9(2):333–354, 2008.

[117] Roberta Bosotti, Giuseppe Locatelli, Sandra Healy, Emanuela Scacheri, Luca Sartori, Ciro Mercurio, Raffaele Calogero, and Antonella Isacchi. Cross platform microarray analysis for robust identification of differentially expressed genes. *BMC bioinformatics*, 8(Suppl 1):S5, 2007.

[118] Giovanni Parmigiani, Elizabeth S Garrett-Mayer, Ramaswamy Anbazhagan, and Edward Gabrielson. A cross-study comparison of gene expression studies for the molecular classification of lung cancer. *Clinical cancer research*, 10(9): 2922–2927, 2004.

[119] Xiaogang Zhong, Luigi Marchionni, Leslie Cope, Edwin S Iversen, ELIZABETH S GARRETTMAYER, Edward Gabrielson, and Giovanni Parmigiani. Optimized cross-study analysis of microarray-based predictors. *Advances in Statistical Bioinformatics: Models and Integrative Inference for High-Throughput Data*, page 398, 2013.

[120] Giovanni Parmigiani, Elizabeth S Garrett-Mayer, Ramaswamy Anbazhagan, and Edward Gabrielson. A cross-study comparison of gene expression studies for the molecular classification of lung cancer. *Clinical cancer research*, 10(9): 2922–2927, 2004.

[121] W Fraser Symmans, Christos Hatzis, Christos Sotiriou, Fabrice Andre, Florentia Peintinger, Peter Regitnig, Guenter Daxenbichler, Christine Desmedt, Julien Domont, Christian Marth, et al. Genomic index of sensitivity to endocrine therapy for breast cancer. *Journal of clinical oncology*, 28(27):4111–4119, 2010.

[122] Wei Sun, Seunggeun Lee, Vasyl Zhabotynsky, Fei Zou, Fred A Wright, James J Crowley, Zaining Yun, Ryan J Buus, Darla R Miller, Jeremy Wang, et al. Transcriptome atlases of mouse brain reveals differential expression across brain regions and genetic backgrounds. *G3: Genes— Genomes— Genetics*, 2 (2):203–211, 2012.

[123] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1073–1080. ACM, 2009.

[124] Douglas Steinley. Properties of the hubert-arable adjusted rand index. *Psychological methods*, 9(3):386, 2004.

[125] Marina Meilă. Comparing clusterings by the variation of information. In *Learning theory and kernel machines*, pages 173–187. Springer, 2003.

[126] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. Wiley. com, 2009.

[127] Martin Maechler, Peter Rousseeuw, Anja Struyf, Mia Hubert, and Kurt Hornik. *cluster: Cluster Analysis Basics and Extensions*, 2013.

[128] Hilary S Parker and Jeffrey T Leek. The practical effect of batch on genomic prediction. *Statistical applications in genetics and molecular biology*, 11(3), 2012.

[129] David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004.

[130] Genevera I Allen. *Transposable Regularized Covariance Models with Applications to High-Dimensional Data*. PhD thesis, Stanford University, 2010.

[131] Elena Parkhomenko, David Tritchler, and Joseph Beyene. Genome-wide sparse canonical correlation of gene expression with genotypes. In *BMC proceedings*, volume 1, page S119. BioMed Central Ltd, 2007.

[132] Sandra Waaijenborg and Aeilko H Zwinderman. Penalized canonical correlation analysis to quantify the association between gene expression and dna markers. In *BMC proceedings*, volume 1, page S122. BioMed Central Ltd, 2007.

[133] Waaijenborg Sandra and Zwinderman Aeilko. Sparse canonical correlation analysis for identifying, connecting and completing gene-expression networks. *BMC Bioinformatics*, 10.

[134] Morine Melissa, McMonagle Jolene, Reynolds Clare, Moloney Aidan, Gormley Isobel, Gaora Peadar, and Roche Helen. Bi-directional gene set enrichment and canonical correlation analysis identify key diet-sensitive pathways and biomarkers of metabolic syndrome. *BMC Bioinformatics*, 11.

[135] John C Platt, Kristina Toutanova, and Wen-tau Yih. Translingual document representations from discriminative projections. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 251–261. Association for Computational Linguistics, 2010.

[136] Alexei Vinokourov, Nello Cristianini, and John S Shawe-taylor. Inferring a semantic representation of text via cross-language correlation analysis. In *Advances in neural information processing systems*, pages 1473–1480, 2002.

[137] Joseph Lee Rodgers, W Alan Nicewander, and Larry Toothaker. Linearly independent, orthogonal, and uncorrelated variables. *The American Statistician*, 38(2):133–134, 1984.

[138] Jonatan Taminau. *inSilicoMerging: Collection of Merging Techniques for Gene Expression Data*, 2013. R package version 1.4.1.

[139] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

[140] Francis R Bach and Michael I Jordan. Kernel independent component analysis. *The Journal of Machine Learning Research*, 3:1–48, 2003.

[141] Abhishek Tripathi et al. Data fusion and matching by maximizing statistical dependencies. 2011.

[142] Francis R Bach and Michael I Jordan. A probabilistic interpretation of canonical correlation analysis. 2005.