Vrije Universiteit Brussel

Faculty of Science and Bio-engineering Sciences
Department of Computer Science
Computational Modeling Lab

Unlocking the potential of public available gene expression data for large-scale analysis

Jonatan Taminau

Dissertation submitted for the degree of Doctor of Philosophy in Sciences

Supervisor: Prof. Dr. Ann Nowé

Print: Silhouet, Maldegem

2012 Uitgeverij VUBPRESS Brussels University Press
VUBPRESS is an imprint of ASP nv (Academic and Scientific Publishers nv)
Ravensteingalerij 28
B-1000 Brussels
Tel. +32 (0)2 289 26 50
Fax +32 (0)2 289 26 59
E-mail: info@vubpress.be
www.vubpress.be

ISBN 978 90 5487 993 0
NUR 910
Legal Deposit D/2012/11.161/011

Voor Maaike en Hanne

Committee members:

Internal members:

Prof. Dr. Ann Nowé
Vrije Universiteit Brussel

Prof. Dr. Bernard Manderick
Vrije Universiteit Brussel

Prof. Dr. Dominique Maes
Vrije Universiteit Brussel

External members:

Prof. Dr. Hugues Bersini
Université Libre de Bruxelles

Prof. Dr. Jacques De Grève
Universitair Ziekenhuis Brussel

Dr. Willem Talloen
Janssen Pharmaceutica NV, Beerse

Dr. Benjamin Haibe-Kains
Institut de Recherches Cliniques de Montréal

Abstract

After more than a decade of microarray gene expression research there is a vast amount of data publicly available through online repositories. It is clear that for the future the new challenges for this technology lie in the integration of this plethora of different data sets in order to obtain more robust, accurate and generalizable results.

A first hurdle for this large-scale integration of studies coming from different labs, using different experimental protocols and even hybridized on different platforms, is the retrieval of the data sets in a uniformed standard. Nowadays it is unfortunately still not possible to retrieve gene expression data in a completely consistent and trackable way and many manual interventions are needed before the actual analysis can be performed. This step is error-prone, leading to obscure errors and reproducibility issues. In this thesis we present the InSilico DB, a tool that provides consistently preprocessed and manually curated genomics data, thereby overcoming many of the current issues related to data acquisition.

In a second hurdle towards the integration of multiple data sets, information from individual gene expression data sets has to be combined and we extensively describe and compare the two main approaches in order to do so: meta-analysis, an approach that retrieves results from individual data sets and then combines the results; and merging, an approach that first combines the actual expression values and then retrieves results on this new data set. Both approaches are described in detail with special

attention for their limitations, issues and advantages.

Both for the consistent retrieval of the data and for the integration of multiple data sets we developed two freely available R/Bioconductor packages providing the necessary tools. These two packages seamlessly integrate with each other and we illustrate their power in a final application where we empirically compare both meta-analysis and merging approaches for the identification of differentially expressed genes in lung cancer.

Samenvatting

Na meer dan een decennium of microarray onderzoek is er een grote hoeveelheid data publiek beschikbaar via online repositories. Het is alsmaar duidelijk dat de nieuwe uitdagingen in de nabije toekomst liggen in het combineren van verschillende bestaande data sets om zo meer robuuste, accurate en generaliseerbare resultaten te bekomen.

Een eerste obstakel voor deze grootschalige integratie van studies, komende van verschillende labs en gebruik makend van verschillende experimentele protocollen en technologieën, is het bekomen van de data in een uniform en gestandaardiseerd formaat. Het is vandaag de dag helaas nog niet mogelijk om op een volledig consistente en traceerbare manier data uit deze repositories te verkrijgen en vele manuele interventies zijn nodig vooraleer de effectieve analyse kan uitgevoerd worden. Deze interventies kunnen leiden tot fouten die niet reproduceerbaar zijn. In deze thesis presenteren we de InSilico DB, een online tool die consistent gegenereerde en manueel gecureerde data aanbiedt en zo de huidige problematiek van data acquisitie probeert te verhelpen.

In deze thesis is ook een tweede obstakel geïdentificeerd: het effectief samenvoegen van de informatie van verschillende data sets. We beschrijven uitvoerig de twee gangbare methoden. In *meta-analysis* worden eerst resultaten bekomen van de individuele studies en dan worden die resultaten gecombineerd. In *merging* gaat men eerst de numerieke gene expressie waarden samenvoegen om dan resultaten te bekomen op deze grote gecombineerde data set. Beide methoden worden in detail bespro-

ken met speciale aandacht voor hun limitaties en sterkten.

Zowel voor de consistente acquisitie van de individuele data sets als voor het uiteindelijke integreren, hebben we twee vrij beschikbare en open software pakketten ontwikkeld die de nodige functionaliteit bevatten. Deze twee pakketten zijn reeds opgenomen in het R/Bioconductor framework en werken naadloos met elkaar samen. We illustreren hun mogelijkheden in een finale applicatie waar we meta-analysis en merging met elkaar vergelijken in de context van het vinden van biomarkers in verschillende bestaande long kanker studies.

Acknowledgments

This dissertation was realized thanks to the encouragement and support of many people. I am indebted to following people for their contributions, both scientific and otherwise:

I am very grateful to my supervisor and promotor Ann Nowé who provided me with the opportunity to start this PhD. I would also like to thank the members of the examination committee who took the time to read this dissertation and provided many helpful suggestions and constructive criticisms.

Further, I enjoyed to work in the inSilico team and many thanks goes to Alain, Ann, Colin, Cosmin, Hugues, David S., David W., Robin and Stijn for this fruitful collaboration. A special thanks goes to Stijn and Cosmin for the very nice and close interaction we had the past few years.

The members of the Computational Modeling Lab (COMO) whom I had the honor and pleasure to work with also deserve a special word of gratitude for creating such a nice and inspiring environment: Abdel, Allan, Ann, Bernard, Bert, Chiqui, Cosmin, David C., David S., Kevin, Kristof, Maarten, Madalina, Marjon, Matteo, Mike, Pasquale, Peter, Ruben, Saba, Steven, Stijn, Sven, Tim, Yailen and Yann-Aël.

Tenslotte wil ik mijn ouders bedanken voor de opvoeding die het mogelijk maakte om een doctoraat te starten. Samen met mijn schoonouders en familie hebben ze mij en mijn gezin ook steeds geholpen op momenten dat het soms moeilijk was.

Mijn grootste dank en liefde gaat uiteraard uit naar mijn vrouw en dochter voor alle steun en plezier in mijn leven. Zij hebben me steeds een reden gegeven om dit doctoraat enerzijds te starten en anderzijds ook zo snel mogelijk tot een succesvol einde te brengen.

Contents

1
Introduction

Bioinformatics is an interesting and currently very challenging research area which is, as the name already suggests, connecting the well established fields of biology and informatics. Constantly improving this field is necessary to cope with the exponential increase of the quantity (and quality) of various kinds of biological data. From this huge amount of biological and biomedical data we want to retrieve relevant information which we then can transform into useful knowledge. This data ⇒ information ⇒ knowledge workflow requires a multi-disciplinary interaction between different domains.

In this thesis and the work we present, we are traveling on top, beyond and hopefully across this bridge between biology and informatics. On the one hand we present tools and solutions mainly inspired from computer science to cope with this large amount of biological data and to optimize the knowledge that can be extracted from it. On the other hand we hope this work can serve as a roadmap for people on both sides, eagerly wanting to cross this bridge and face the many challenges.

A central tool or framework throughout this thesis is the Bioconductor toolkit, which provides tools for the analysis and comprehension of high-throughput genomic data [Gentleman et al. (2004)]. This framework consists of a huge collection of public available and open source packages for the statistical language R [R Development Core Team (2005)]. The R/Bioconductor project's focus is on reproducibility and in the same ideology we made many of the tools developed during this thesis public available. Two new packages were added to the Bioconductor repository and many key code fragments are provided to the reader in Appendix A.

This dissertation was performed in the context of the *InSilico project*, a five-years project funded by the Brussels Institute for Research and Innovation (INNOVIRIS)[1]. This project consisted of more than eight pre- and post doctoral researchers divided over the two participated labs: the COMO lab from the Vrije Universiteit Brussel (VUB) and the IRIDIA lab from the Université libre de Bruxelles (ULB). Many of the decisions and directions in this dissertation were taken with respect to the common goals of this project.

In this introductory chapter we will first outline the specific research area we will focus on and clarify the actual aim of this thesis. We will provide a per-chapter overview of the entire manuscript and end with a detailed overview of my own contributions in each part.

1.1 Situation and Aim of this Thesis

Microarrays are a high-throughput technology to measure the abundance of gene transcripts in a particular sample. Gene transcripts can be seen as intermediate biochemical molecules that transfer the information captured in genes to the corresponding proteins. This transfer of information is essential for every cell since genes are fixed in the DNA sequence that

[1] http://www.innoviris.be/

is stored in the nucleus, while proteins can travel across the whole cell and organism to fulfill their specific roles. The abundance of transcripts is actually an approximation of the intensities of the genes that are expressed in a given cell. Therefore, microarrays are mostly described as measuring *gene intensities* or *gene expression*.

One of the big advantages with respect to other techniques measuring gene transcripts, and what has lead to the major breakthrough of microarrays, was its ability to measure the expression of thousands of genes at the same time. This immediately leads to a massive amount of data per experiment.

Since the costs of microarray technology were constantly decreasing and its use as discovery tool was proven, it gained a lot of popularity in the last decade. Nowadays there is a vast amount of gene expression microarray data sets publicly available through several online repositories. It is clear that in the near future the new challenges for this technology lie in the integration of this plethora of different data sets in order to obtain more robust, accurate and generalizable results.

In order to obtain a gene expression matrix, where the rows contain all the genes measured on the array and the columns corresponds to the different samples, many complicated steps have to be performed. Roughly speaking the major steps can be categorized as: tissue collection, mRNA extraction, probe hybridization, fluorescence detection, image processing and numerical preprocessing. It is clear that at each step many uncontrollable factors or parameters can influence the resulting gene expression matrix. This leads to many of the reproducibility and compatibility issues generally associated with microarray data.

Unfortunately, it is currently still not possible to retrieve gene expression data in a completely consistent and trackable way and many manual interventions are needed before the actual analysis can be performed. This step is error-prone, leading to obscure errors and reproducibility issues and severely hindering any comparative analysis.

Even when the data is unambiguously described and provided in a trackable and reproducible manner it is still not possible to simply combine several studies and conduct large scale integrative analysis. The many uncontrollable factors during the different steps of the microarray analysis (e.g. temperature and light intensity during fluorescence detection, different protocols and reagents for tissue extraction, different design of arrays, etc.) lead to a situation were the data between different studies or experiments are not comparable. The combination of all factors leading to this incompatibility are called *batch effects*.

It is only recently that the research community is becoming aware of this undesirable phenomenon [Leek et al. (2010)], currently blocking the integration of existing microarray data. The aim of this thesis is to further investigate this problem. Moreover, we will propose new approaches to deal with the issues currently hindering large-scale analysis and develop and provide the necessary tools in order to do so.

1.2 Overview

We start this thesis by explaining the basic concepts related to microarray technology and the type of data it produces in Chapter 2. By providing a brief introduction in basic cell biology, the underlying principles of microarrays can be explained. Having at least a notion of the technical details of gene expression microarray technology is important to understand many of the issues encountered in the next chapters.

The first hurdle for conducting integrative microarray analysis is the problem of consistent data acquisition. The problems related to the retrieval of consistent, reproducible and trackable data from public repositories will be explained, with examples from literature, in Chapter 3.

In the next chapter we present a tool that was developed to overcome many of the problems related to data acquisition: the InSilico database. This tool was one of the main objectives of the InSilico project. An overview of the implementation details is provided with special focus on the genomic pipelines and the underlying backbone it uses in order to provide expert-curated and consistently preprocessed data. A detailed discussion of the functionalities of the tool and the specific choices that were made help us to solve many of the issues identified in Chapter 3. We end Chapter 4 by presenting our first publicly available R/Bioconductor package `inSilicoDb`, which can be used to programmatically query the InSilico database.

In the second part of this thesis we start by introducing the second hurdle towards integrative microarray analysis: the problem of combining different studies. In Chapter 5 we discuss the many benefits of the integration of microarray data, together with the possible pitfalls. Two main approaches are identified, each being the subject of the two following chapters.

In Chapter 6 the meta-analysis approach is discussed and a concrete application illustrating its possible use is presented. In this application we use the massive amount of public available studies from the InSilico database to screen for stable genes, which can be used as reference genes for normalization purposes. We extracted a compact and diverse list of 12 genes, all with a stable expression profile across all biological conditions present in the InSilico database.

The second approach, merging, is the focus of Chapter 7. First the concept and the main sources of batch effects are discussed, followed by an extensive overview of the current methods to remove this batch effect during merging. A complete list of both quantitative and visual validation methods for the batch effect removal process is provided as well. We end Chapter 7 by presenting our second public available R/Bioconductor package `inSilicoMerging`, which bundles most of the existing batch effect removal and validation methods in a unified framework.

Finally, a concluding application is presented in which we compare both meta-analysis and merging approaches for integrative gene expression microarray analysis in the context of the identification of differentially expressed genes for lung cancer. Without looking at the different subtypes of lung cancer, we obtain with both approaches a list of potential biomarkers, which are believed to play a role in the most basic principles or mechanisms of lung cancer, and cancer in general.

1.3 Contributions

My first contribution was in the development of the InSilico database (described in Chapter 4) as part of the InSilico team[2]. I played an active role in the design, development, implementation and maintenance of this tool. An article presenting InSilico database is currently under review for *Genome Biology*, one of the leading journals for high-quality and innovative tools for computational biology ([Coletta et al. (2012)], impact factor: 9.04). My main contribution was the development of the genomic pipelines (see Section 4.3.1) and the internal backbone (see Section 4.3.2), both essential for the consistent data preprocessing and precomputing. A spin-off is currently set-up to commercialize this tool.

In addition I also developed a R/Bioconductor package to programmatically access and query InSilico database, called `inSilicoDb` (see Section 4.4), which was published in *Bioinformatics*, one of the top ranked journals in Mathematical and Computational Biology ([Taminau et al. (2011)b], impact factor: 4.88). As of July 2012, the `inSilicoDb` package is downloaded more than 1500 times[3] and is already used in large-scale gene expression analysis published in other high-quality journals [Tomás et al. (2012),Tamayo et al. (2011)].

[2] https://insilico.ulb.ac.be/

[3] http://www.bioconductor.org/packages/stats/bioc/inSilicoDb.html

The application to identify stable genes through meta-analysis (see Section 6.3) is novel and own work. This study is yet unpublished but a manuscript is in preparation. Currently, wet-lab validation is ongoing in collaboration with Dr Bram de Craene from the VIB Departement Moleculair Biomedisch Onderzoek (UGent), where the stable genes will be validated in the context of qPCR normalization.

For the merging of gene expression microarray data my earliest contribution was an empirical comparison of different existing merging methods in three different cases with increasing complexity: NCI60 cell lines, thyroid cancer and breast cancer [Taminau et al. (2009)c, Taminau et al. (2010)c]. Next I was co-first-author of an extensive survey of batch effect removal methods which was published in *Briefings in Bioinformatics* ([Lazar et al. (2012)a], impact factor: 9.28). This is still, to our knowledge, the most thorough review of this topic that is gaining more and more attention from the field.

In addition I also developed a R/Bioconductor package, called `inSilicoMerging` (see Section 7.4) which bundles most existing batch effect removal methods for merging, together with a wide variety of validation measures. This tool seamlessly integrates with the `inSilicoDb` package. As of July 2012, the `inSilicoMerging` package is downloaded more than 500 times[4], although not even published. An article presenting the tool is currently under submission for *BMC Bioinformatics* ([Taminau et al. (Subm)], impact factor: 2.75).

The application in which I conducted integrative analysis of microarrays through both meta-analysis and merging (see Chapter 8) is also novel and own work. Further analysis of the results still needs to be done prior to publication and a collaboration with experts in lung cancer is already set up.

[4] `http://www.bioconductor.org/packages/stats/bioc/inSilicoMerging.html`

1.3.1 Contributions Outside the Scope of this Thesis

During this thesis we traveled a few side tracks not covered in this final manuscript. We will briefly list the different topics in this section. A complete list of publications can be found in Appendix B.

At the beginning of my scholarship we investigated the use of *Subgroup Discovery*, a rule-based machine learning technique lying on the intersection of predictive and descriptive induction, on symbolic music data. The initial aim was to use this technique in a later stadia on biological data. In a first preliminary study we applied subgroup discovery on a cohort of symbolic folk music pieces to explain their geographical origin [Taminau et al. (2009)b, Taminau et al. (2009)a].

In a second follow-up study we applied the same technique to a data set composed of 112 string quartet movements from Haydn and 95 string quartet movements from Mozart to find subgroups that are characterized by one of the two composers [Taminau et al. (2010)a, Taminau et al. (2011)a]. Although we still are convinced this type of techniques may be very suitable for genomic data, we decided to concentrate on the two core aspects of this thesis listed above, and we left this interesting topic to possible future research.

In two other publications that are not fully described in this thesis I had a major contribution: an extensive overview of filter techniques for feature selection in gene expression microarray analysis [Lazar et al. (2012)b] and the combined application of feature selection and feature extraction methods in the context of identifying gene signatures for breast cancer aggressiveness [Taminau et al. (2010)b]. A concise continuation of this research was also presented as a poster spotlight talk at the Cancer Bioinformatics Workshop 2010, Cambridge (UK)[5].

Finally, I also had minor contributions in the application of computa-

[5] `http://www.enm.bris.ac.uk/cig/cb/canbioprog.pdf`

tional modeling techniques for distributed feature selection problems [Gómez et al. (2009)] and for computer-aided ligand-based drug design [Pérez-Castillo et al. (2012)].

2 Preliminaries

In this chapter some context and basic information to understand gene expression analysis is provided. We start by briefly summarizing the essential biological background in Section 2.1. Within this context the working of two high-throughput gene expression profiling techniques will be detailed: DNA microarrays in Section 2.2.1 and next-generation sequencing (NGS) in Section 2.2.2. Finally, the complete analysis workflow for microarray gene expression data, from preprocessing to visualization, is described in Section 2.3.

2.1 Biological Context

To understand and interpret the results of a microarray experiment, the so-called gene expression values, it is important to be aware of the biological concepts behind this technology. In order to do so for any kind of reader we have to go back to the *Central Dogma of Biology* [Crick (1970)], proposed by Francis Crick, co-discoverer of the structure of the DNA molecule. In its simplest form, this framework, illustrated in Figure 2.1, states that genes are transformed into proteins; but not vice versa. Genes, which are parts of the DNA and located in the nucleus of every cell, carry all information needed to grow and maintain every organism. This information needs to be transformed into proteins, responsible for the proper working (and sometimes failure) of the cell. This transfer of information has however an intermediate step. First a gene is *transcribed* into a mRNA molecule, containing the same information but able to leave the nucleus through the rest of the cell. This mRNA, or messenger RNA, is a less stable molecule but can be *translated* into a protein molecule, through a complex process in which RNA base pairs are mapped onto amino acids, the building blocks of a protein. The newly created protein can then perform his specific jobs in the cell.

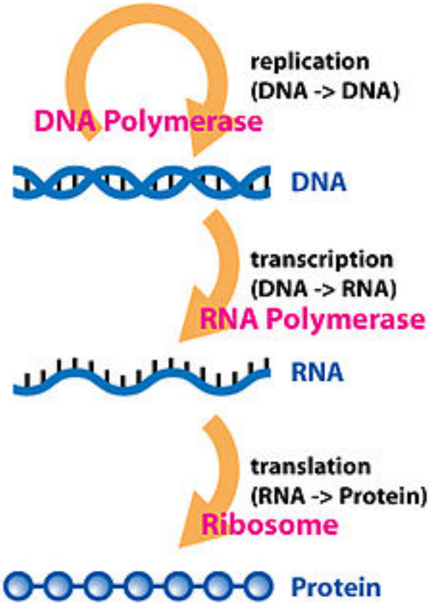


Figure 2.1: An overview of the central dogma of biology. Enzymes facilitating each step are labeled in pink. The orange arrows describe the transfer of knowledge.

After the whole human genome was sequenced in 2001 [Venter et al. (2001)], estimates of the number of genes in humans lie between 20.000

and 30.000 [Pennisi (2003)], but not all these genes are transformed into proteins all the time. Only part of the total number of genes are *expressed* at a certain point in time, for example because the cell always needs specific basic proteins in order to stay alive, or in response to external triggers of the cell. This whole process of expression is controlled by so called *transcription factors*, proteins binding to specific places in the DNA and thereby enhancing or inhibiting the expression of certain genes. These transcription factors are again part of complex protein-protein networks, containing multiple feedback loops.

The behavior or status of a cell can thus be defined by the composition of its proteins. Healthy cells will have for example other proteins present (and thus other genes expressed) than malignant or cancer cells. Many drugs are based on a specific interaction with a protein which is known to be (partially) responsible for that disease. It is therefore crucial to detect those proteins as potential targets in early drug discovery [Lindsay (2003)]. In general, the ability to quantify the level at which one or more genes/proteins are expressed can provide a huge amount of valuable information. Ideally this measurement is performed by detecting the final product, the proteins, but in reality it is often easier to detect the intermediate product: mRNA[1]. The complete set of (mRNA) transcripts in a cell, and their quantity, is called the transcriptome.

Methods to quantify the levels of mRNA are northern blotting, a technique which detects size and sequence information of mRNA molecules [Alwine et al. (1977)], and quantitative reverse transcription polymerase chain reaction (qRT-PCR), a technique which amplifies copied DNA templates (cDNA) from mRNA molecules [Heid et al. (1996)]. Both techniques are well suited to detect the expression of single genes with high accuracy. With the invention of DNA microarrays on the other hand,

[1] We have to note however that there is no exact one-to-one mapping from mRNA to gene/protein due to an intermediate process called *splicing* in which a mRNA molecule can give rise to different proteins through alternative splicing. This is however beyond the scope of this thesis but the reader should be aware of the assumption made.

transcript levels for thousands of genes can be measured simultaneously [Schena et al. (1995), Lockhart et al. (1996), Brown & Botstein (1999)]. The remainder of this thesis will handle with the output of these high-throughput gene expression microarrays and we will briefly explain its working in the next section.

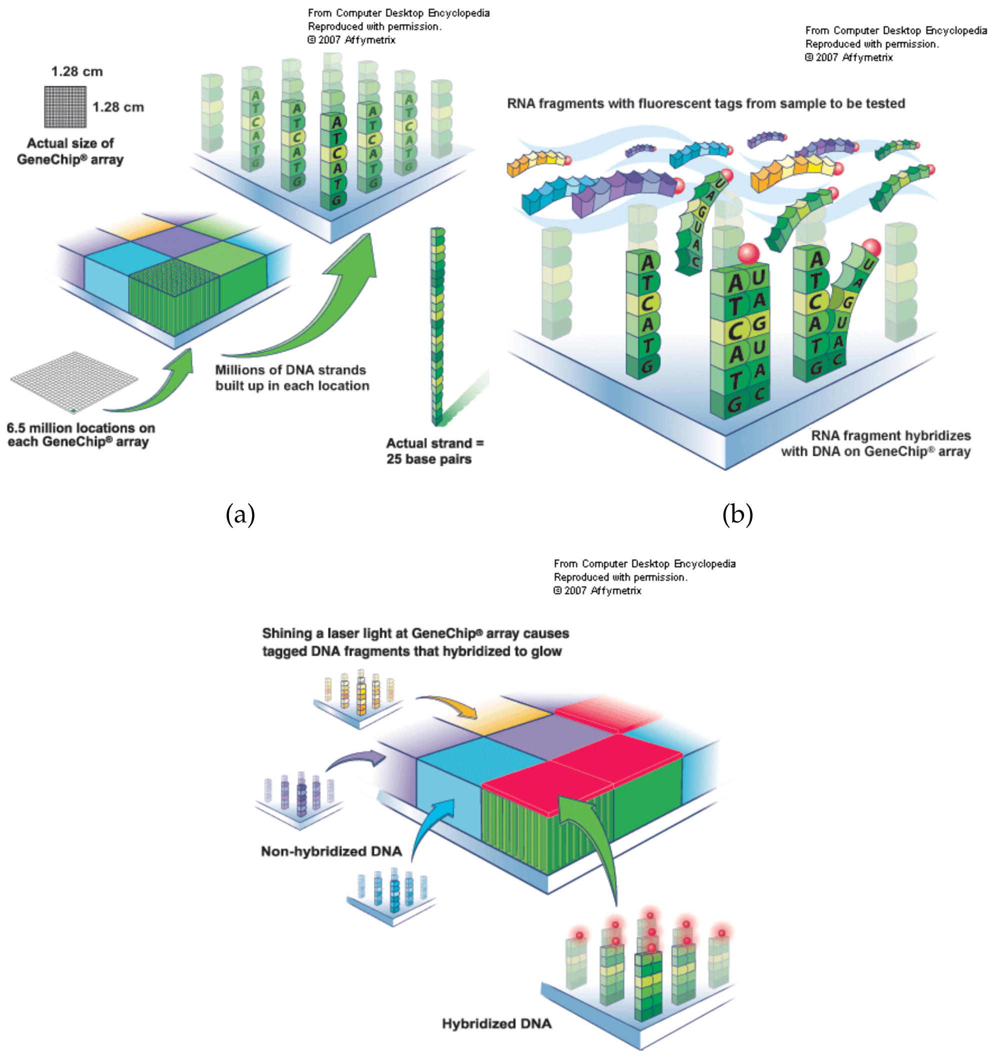
2.2 High-Throughput Gene Expression Technology

Microarrays made the analysis of the transcriptome possible more then a decade ago, and have produced much important information. Today, researchers are increasingly turning to direct high-throughput sequencing (RNA-Seq) which has considerable advantages for examining transcriptome fine structure. Both technologies show however similar performance and complement each other [Malone & Oliver (2011)]. In this section we explain both techniques and motivate our choice for focussing on microarray technology in the remainder of this thesis.

2.2.1 DNA Microarray Technology

A DNA microarray is a 2D array on a solid substrate, usually glass or silicon, composed of DNA fragments, called probes, positioned on its surface. Other biological material can be assayed as well, resulting in a wide variety of different microarrays (peptide, protein, microRNA, tissue, etc.). For DNA microarrays each probe(set) represent a specific gene coding region, see Figure 2.2 (a). In a next step, purified mRNA fragments from a biological sample of interest are then fluorescently labeled and hybridized to the chip, see Figure 2.2 (b). Finally, non-hybridized fragments are removed by washing the array and laser based scanners can detect the areas on the chip where hybridization occurred, see Figure 2.2 (c).

Currently there are four dominant microarray vendors offering DNA microarrays: Affymetrix (Santa Clara, CA, USA), Agilent Technologies (Santa


From Computer Desktop Encyclopedia
Reproduced with permission.
© 2007 Affymetrix
1.28 cm
1.28 cm
Actual size of GeneChip® array
Millions of DNA strands built up in each location
6.5 million locations on each GeneChip® array
Actual strand = 25 base pairs
From Computer Desktop Encyclopedia
Reproduced with permission.
© 2007 Affymetrix
RNA fragments with fluorescent tags from sample to be tested
RNA fragment hybridizes with DNA on GeneChip® array
From Computer Desktop Encyclopedia
Reproduced with permission.
© 2007 Affymetrix
Shining a laser light at GeneChip® array causes tagged DNA fragments that hybridized to glow
Non-hybridized DNA
Hybridized DNA

(a) (b)

(c)

Figure 2.2: Visual overview of the working of DNA microarrays. (a) Zoomed and detailed view of a microarray chip. (b) Hybridization of the RNA fragments of the biological sample of interest (floating) and the probes of the array (fixed). The red dot attached to the fragments represents the fluorescent labeling. (c) Quantify hybridization by laser based scanner for each area of the array. Figures taken from [Affymetrix (2002)].

Clara, CA, USA), Illumina (San Diego, CA, USA) and Roche Nimblegen (Madison, WI, USA). Although each vendor manufactures his array in a different way, the underlying mechanism is always the same: obtain RNA or DNA fragments from a biological source of interest, hybridize it to the probes on the array and measure the amount of hybridized fragments.

In the remainder of this thesis we will focus on Affymetrix microarrays. Affymetrix typically synthesizes 25-mer oligonucleotide probes on their arrays, see Figure 2.2 (a). All probes on an Affymetrix array occur in pairs, consisting of two different sequences: one that is complementary to the transcript it is supposed to hybridize with, leading to a perfect match (PM) and one that has a central mismatch in its sequence (MM). The eventual expression value for that probe is a combination of both values, where the mismatch probe can be used for initial normalization and background correction. On more recent arrays we speak of *probe sets* instead of *probe pairs* since more than two different sequences are used per probe.

2.2.2 Next-Generation Sequencing (NGS)

Another more recent technique to investigate the transcriptome can be found in one of the applications of *next-generation sequencing* (NGS). NGS is an improved technology for sequencing large numbers of human genomes in a fast, inexpensive and accurate manner [Metzker (2010)]. This production of large volumes of sequence data is the primary advantages over conventional or *first generation* methods, used for example to sequence the first human genome [Venter et al. (2001)].

In contrast to hybridization-based methods (e.g. DNA microarrays, Section 2.2.1), sequence-based methods directly determine the cDNA sequence, enabling more accurate results and overcoming several limitations of hybridization methods. For gene expression analysis the method *RNA-Seq* (RNA sequencing) was developed and its working is illustrated in Figure 2.3. Note that for the sequencing step any high-throughput sequencing technology can be used.

Although RNA-Seq and other NGS technologies clearly have a number of advantages over microarrays, there still are some issues or *childhood diseases*: the necessity of an amplification step prior to sequencing, cDNA library construction, management and costs of large amounts of data, etc. [Wang et al. (2009)].

Since measuring transcripts using NGS technology is very recent, analysis software is still continually appearing and it is yet unclear how each of those methods perform on different genomes (most studies are only performed on mouse transcripts). Currently there are however two main approaches for reconstructing transcripts from RNA-Seq reads [Haas & Zody (2010)]. The *assemble-then-align* approach first assembles transcript sequences before aligning. Examples methods are ABySS [Birol et al. (2009)] and Oasis [Schulz et al. (2012)]. A second approach, the *align-then-assemble* approach is more common and first aligns short RNA-Seq reads to the genome, accounting for possible splicing events, and then reconstructs transcripts from the spliced alignments. Example methods are Cufflinks [Trapnell et al. (2010)] and Scripture [Guttman et al. (2010)], both programs use the TopHat aligner [Trapnell et al. (2009)] to generate spliced alignments.

In Section 4.3.1 we briefly detail the pipeline we used in InSilico DB. The remainder of this thesis will however focus on microarray technology for gene expression analysis.

2.2.3 Discussion

Both sequencing and hybridizing mRNA to arrays are high-throughput ways to profile the transcriptome and for problems that can be addressed by both, they show similar performance and complement each other [Malone & Oliver (2011)]. Despite their recent fallback in popularity, microar-

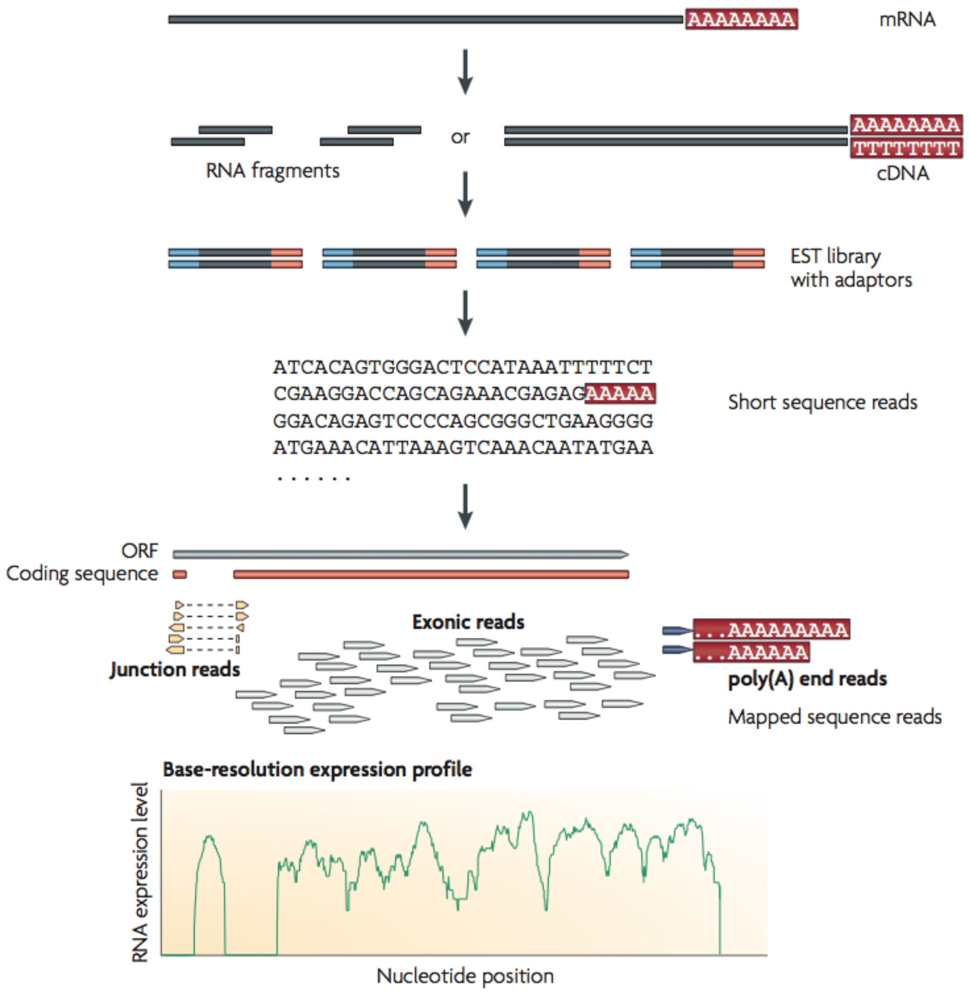


Figure 2.3: Overview of RNA-Seq technology. A transcript is first converted into a library of cDNA fragments. Each fragment is then sequenced using adaptors added to one (blue) or both (blue & orange) ends. The resulting sequence reads are aligned with a reference genome and classified as three types which are all used to generate a expression profile for each gene. Figure taken from [Wang et al. (2009)].

rays remain useful and accurate tools for measuring expression levels, and RNA-Seq complements and extends microarray measurements.

The choice to focus on microarray data in this thesis was mainly influenced by the global goal when the InSilico project was initiated five years ago. The inSilico DB was developed to cope with the increasing amount of unmanaged microarray studies (see Chapter 4) already present at that time. Although we adapted this tool to store, preprocess and manage NGS data as well, there are different needs for this new technology.

In the context of this thesis, the accumulated data and studies of a decade of microarray research provides an unmatched variety of information present in public repositories. Since it will take NGS technologies for gene expression analysis several years to reach the same level of maturity, both technology- and content-wise, it is in our opinion still worthwhile to concentrate further on microarray technology.

2.3 Microarray Gene Expression Analysis

In this section we focus on the analysis of microarray gene expression data. Many of the concepts, from pre-processing to analysis, are however also applicable for RNA-Seq data.

2.3.1 Caveats of Microarray Technology

Although the general mechanism of microarrays described in the previous section looks straightforward, it is still a very complex method with typical characteristics that make microarray analysis challenging. In the context of this thesis it is essential to be aware of the possible limitations and caveats of microarray data. We will do this by enumerating the most important issues:

High level of noise

Since the early years of microarrays, noise and its impact on microarray analysis has been recognized [Kerr et al. (2000), Tu et al. (2002), Ioannidis (2005)] as an undesirable consequence of the technique, influencing downstream analysis. The purity of the samples and the many technical steps in the method lead to variability in the experimental outcomes. Especially the measurements of low abundance genes are difficult to detect accurately [Draghici et al. (2006)].

Poor reproducibility

Another open issue is the poor reproducibility of microarray results across studies. Analyzing the same tissues using the same technology but by different labs, can produce different analysis results, questioning both reproducibility and *reliability* of microarrays [Marshall (2004)]. This issue was clearly demonstrated by different studies showing the inability of researchers to replicate differentially expressed gene lists (see Section 2.3.3) across highly similar experiments [Tan et al. (2003),Michiels et al. (2005)]. However, it turned out that the choice of data analysis procedures addresses and circumvents many of these problems [Shi et al. (2005)].

Cross-platform Consistency

Another related issue is the robustness or consistency of results across different platforms [Kitchen et al. (2011)]. Different platforms contain different probes and therefore one of the main difficulties in the cross-platform comparison of microarray data is to ascertain that probes on the various platforms aimed at the same gene do in fact quantify the same mRNA transcript [Draghici et al. (2006)]. Due to this and other factors, consistency between different platforms suffers from similar but more prominent problems as consistency within a platform.

Low sample size

The number of samples in microarray studies typically lie between 10-100[2]. In order to generate robust gene signatures for predicting outcome of disease, actually thousands of samples are needed [Ein-Dor et al.

[2] The average number of samples per study for all 2267 studies assayed on the Affymetrix Human Genome U133 Plus 2.0 platform found in InSilico DB is **33** samples per study. Code to obtain this value can be found in Appendix A.1.

(2006)]. The current low sample sizes might actually hinder the identification of truly important genes [Ioannidis (2005), Ein-Dor et al. (2005)]. Moreover, since in microarray studies each sample is characterized by thousands of genes[3] the low sample sizes make this data extra vulnerable to the *curse of dimensionality* [Somorjai et al. (2003)] and this high dimension, low sample size (HDLSS) data requires specific statistical analysis [Benito et al. (2004)].

As already explained before, combining gene expression data from different studies can circumvent some of these issues but at the same time can be highly influenced as well, as we will see in Section 7.2.

2.3.2 Microarray Data Preprocessing

The *raw* data resulting from a microarray experiment can be seen as an image file capturing the observed fluorescent intensities for all probes. A first step is to transform this image into numerical values. In the Affymetrix system, the raw image data is stored in *DAT files* and Affymetrix has its own image analysis software to estimate probe intensity values, resulting in a so-called *CEL file*, containing all probe-level data. Bioconductor [Gentleman et al. (2004)], offers a lot of tools to import, examine and normalize CEL files. At the end of this section we will provide a typical example workflow.

The actual preprocessing of microarray arrays starts with the CEL files and usually involves three steps: background adjustment, normalization and summarization. We will briefly explain each step:

Background adjustment

The major reason for background noise in microarray data is non-specific binding, i.e. in reality the sample consists of a complex mixture of nucleotide molecules and non-complementory sequences also bind to the

[3] For the same Affymetrix Human Genome U133 Plus 2.0 platform, the number of probes is **54675**.

probes. This first step consists of an adjustment for hybridization effects that are not associated with the interaction of the probes with target DNA. Measurements on *mismatch probes* (MM) are designed to measure this array specific background and several methods simply subtract the *perfect match probes* (PM) intensities with the MM intensities to adjust for background noise.

Normalization

In addition to background noise there are often other sources of variation affecting the observed measurements that are not of biological interest. These can be technical variations due to the scanner for example or variation introduced by the preparation of the samples. Most normalization methods equalize a summary statistic of the distribution of the measurements across arrays. This normalization implicitly assumes that biological variations of interest may affect a number of probe intensities, but should not change the mean or mode of the distribution of all intensities on each array.

Summarization

In a last step the different probe sets are combined into one expression value per probe. Most methods choose a robust summary that is resilient to outliers. A difference can be made between single array summaries and multiple array summaries, where the latter uses information across arrays to identify outliers. The probe sets are usually summaried on a log-2 scale.

For each step different methods are available and it is not clear which ones, and in which combinations, are best suited. For example in [Harr & Schlötterer (2006)] they propose different preprocessing methods depending on the possible downstream analysis that will follow. Other large comparison studies are performed as well [Bolstad et al. (2003)], but with no golden standard so far.

In Bioconductor plenty easy-to-use preprocessing methods are available combining all three steps and they are widely used as a first step in microarray data analysis. Some of the most used methods are Affymetrix's Microarray Suite 5 (`MAS5`, [Affymetrix (2002)]), Robust Multichip Average (`RMA`, [Irizarry et al. (2003)]), RMA using sequence information (`GCRMA`, [Wu & Irizarry (2004)]), Variance Stabilization Normalization (`VSN`, [Huber et al. (2002)]) and frozen RMA (`FRMA`, [McCall et al. (2010)]). Their usage in the Bioconductor framework is very straightforward:

```
#path to example directory with CEL files
path = "/home/data/CEL/";

# read CEL files...
library(affy);
batch = ReadAffy(filenames=path);

# Perform preprocessing...
library(rma);
eset = rma(batch);
```

`RMA` is one of the mostly used preprocessing methods. Since it was shown that the assumption of MM probes measuring only background intensities was not correct, RMA avoided this problem and uses only PM measurements. It assumes a global background distribution common for all probes that is normally distributed. For each sample, probe intensities that are smaller than the empirical mode are used to estimate the mean and variance of the background distribution. For the normalization step it uses quantile normalization [Irizarry et al. (2003)], which forces all samples to have the same empirical distribution of intensities after normalization. As the name robust multichip average suggests, a multiarray summarization is performed using median polish [Irizarry et al. (2003)] as a robust procedure to protect against outliers.

After Affymetrix made their probe sequences public it was found out that the middle base being purine (A or G) or pyrimidine (C or T) affects the hybridization and partially explained why MM probes did not always offer the expected background measurements. GCRMA is an extension of RMA that calculates affinity values for each probe sequence based on its exact base sequence and use this extra information to adjust for the probe specific background [Wu & Irizarry (2004)].

In FRMA a small adoption of the RMA methods was introduced. The quantile normalization part is done based on a reference training set of publicly available samples from a diverse population instead of using only the samples of the study to normalize. Estimates of probe-specific effects and variances are also obtained and all information is *frozen* [McCall et al. (2010)]. For each new array to be preprocessed, background correction is performed similar to the training set and then it is quantile normalized based on the reference distribution. During summarization batch effects are removed and variances of the gene-expressions are estimated by taking into account these probe-specific effects.

All these methods try to remove potential systematic bias and to make different studies more comparable among each other. We can divide the above mentioned methods in three groups: MAS5 can only be used on a single array, RMA, GCRMA and VSN can be used on a set of arrays and finally we have FRMA which always uses a diverse cohort of samples from a broad range of different studies on the same platform, even if used on a single array. Most methods are not capable of removing batch effects and thus fail in making all microarray data comparable with each other [Scherer (2009), Leek et al. (2010)], with exception of FRMA if all samples are from the same platform. A more detailed discussion will follow in Chapter 5 and further.

For more information on available R/Bioconductor packages for Affymetrix preprocessing the reader is invited to consult Chapter 2 in [Gentleman et al. (2005)]. A good overview of all statistical methods for preprocessing of microarray data can be found in [Wu (2009)].

In the remainder of this thesis, preprocessed microarray data is denoted by $X^{m \times n}$, where each column represents a sample and each row represents a feature. x_{ij} represents the expression value of feature i in sample j. This notation is analog with the `ExpressionSet`, the central data structure from Bioconductor, a standard we fully support. Since most preprocessing methods summarize their probe sets on a log-2 scale, it is further assumed that every gene expression matrix is already log-2 scaled.

At this point it is important to note that the features after preprocessing are probes and although it is possible to perform analysis on this probe-level data it is often desirable to work with genes instead. There are two main reasons. Firstly, genes are easier to interpret by biologists since they have a function and annotation while a probe itself doesn't *mean* anything. Secondly and more important in the context of this thesis, probes can't be used to compare and integrate microarray data from different platforms because probes are platform dependent (see Section 2.3). Mapping probes to genes is, again, an active research domain on its own and the major challenge lies in the fact that multiple probes can match to the same gene. Probes are much shorter sequences than genes and thus multiple probes can map to different regions of one gene. In Section 4.3.1 a detailed description of the probe-to-gene mapping used in the InSilico DB pipeline is provided.

2.3.3 Microarray Data Analysis

The major objective of performing microarray experiments is to derive biological knowledge or test biological hypotheses. The volume of information in microarrays and other high-throughput genomic data favors machine learning techniques which are positioned for problems of pattern recognition in voluminous noisy data with minimal human input [Mitchell (1997)]. From a machine learning perspective, roughly two types of analysis can be performed: *unsupervised analysis* and *supervised analysis*.

In unsupervised analysis the input data is explored without using extra information and patterns and/or groupings are obtained which can be tested against biological hypotheses. It was one of the first statistical techniques to be applied on microarrays, due to its simplicity and lack of assumptions needed. While it is still a very popular technique, questions about validity and reproducibility are arising, leading to statements as "Unsupervised classification is overused." [Allison et al. (2006)].

Supervised analysis tries to find a relation between the input data and external information, for example by assigning instances (e.g. samples or genes) to *a priori*-defined classes (e.g. tumor subtypes). In supervised analysis sample size is critical and simpler methods often out-perform more complex approaches due to overfitting [Pranckeviciene & Somorjai (2006)].

Many analysis tool kits for gene expression data exists (R/Bioconductor [Gentleman et al. (2004)], GenePattern [Reich et al. (2006)], Galaxy [Goecks et al. (2010)], Spotfire [TIBCO Software Inc., CA, USA], etc.), offering a plethora of different analysis techniques. Many of these tools are publicly available and even open-source. As already mentioned before, the analysis techniques themselves are not anymore the problem or bottleneck in current microarray analysis, but building the complete and reproducible workflow from data to results is still the challenge. In Figure 2.4 some guidelines in each step of the workflow were proposed.

Although there are numerous studies, approaches and objectives reported in literature for the analysis of gene expression data most research can be divided into four broad categories: class comparison, class prediction, class discovery and pathways analysis. We will briefly explain each category:

Class Comparison

A very common objective and application of microarray studies is the identification of genes that are consistently and significantly expressed at different levels under different conditions. These genes are called infor-

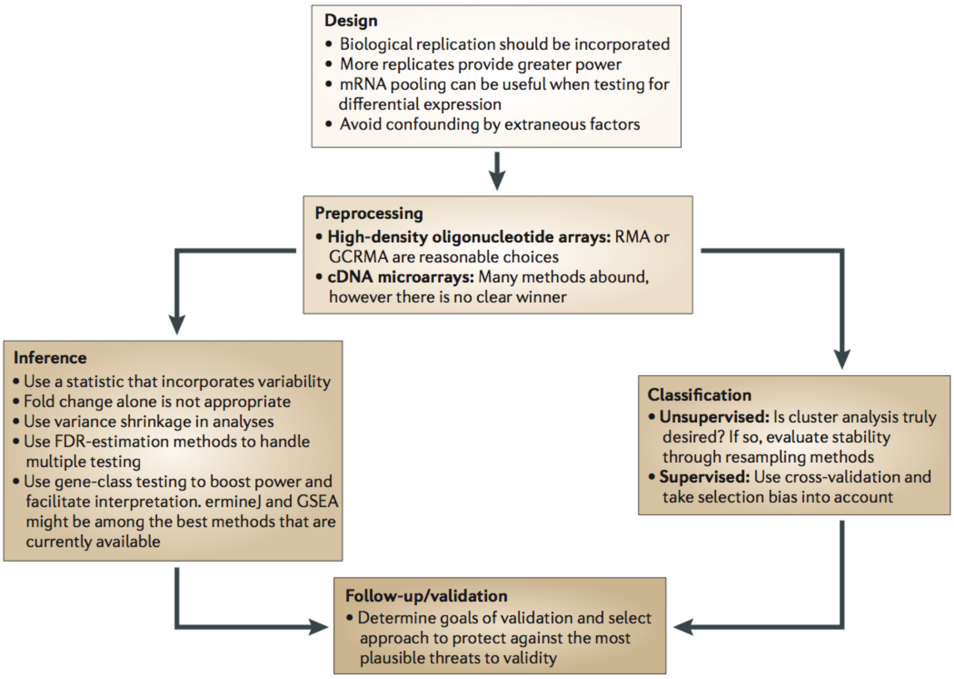

Design
• Biological replication should be incorporated
• More replicates provide greater power
• mRNA pooling can be useful when testing for differential expression
• Avoid confounding by extraneous factors
Preprocessing
• High-density oligonucleotide arrays: RMA or GCRMA are reasonable choices
• cDNA microarrays: Many methods abound, however there is no clear winner
Inference
• Use a statistic that incorporates variability
• Fold change alone is not appropriate
• Use variance shrinkage in analyses
• Use FDR-estimation methods to handle multiple testing
• Use gene-class testing to boost power and facilitate interpretation. ermineJ and GSEA might be among the best methods that are currently available
Classification
• Unsupervised: Is cluster analysis truly desired? If so, evaluate stability through resampling methods
• Supervised: Use cross-validation and take selection bias into account
Follow-up/validation
• Determine goals of validation and select approach to protect against the most plausible threats to validity

Figure 2.4: Guidelines for the statistical analysis of microarrays at each step in a typical workflow. Figure taken from [Allison et al. (2006)].

mative genes, biomarkers or differentially expressed genes (DEGs). The discovery of DEGs is one of the most important applications of microarray analysis. It is valuable to physicians to diagnose patients but also to pharmaceutical companies aiming to identify genes for drug target identification [Lazar et al. (2012)b]. Many strategies for feature/gene selection are proposed over the last years, categorized in filter, wrapper, embedded and ensemble techniques [Saeys et al. (2007)].

Filter techniques are the most used in the context of finding DEGs due to its simplicity and its computational and statistical scalability. They can be seen as a supervised feature selection method in which the biological variable of interest is used as the discrimination factor. Most filter methods follow a typical scenario:

1. Use a scoring function to quantify the difference in expression between different groups of samples, according to the biological variable of interest and rank the genes.

2. Estimate the statistical significance (e.g. p-value, confidence intervals, etc.) of the estimated scores.

3. Select the top ranked genes which are statistically significant as the most informative genes.

4. Validate the selected subset of genes.

This workflow is also illustrated in Figure 2.5. Since all the steps are independent from each other, there is a lot of freedom in the way they can be performed and a large plethora of different methods indeed exist. We made an extensive overview of these methods [Lazar et al. (2012)b] which is unfortunately beyond the scope of this thesis.

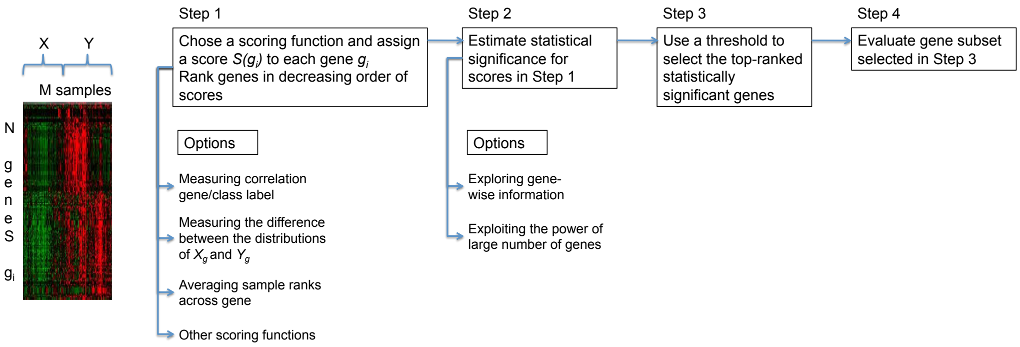

X
Y
M samples
N
g
e
n
e
S
g_i
Step 1
Chose a scoring function and assign a score $S(g_i)$ to each gene g_i Rank genes in decreasing order of scores
Options
Measuring correlation gene/class label
Measuring the difference between the distributions of X_g and Y_g
Averaging sample ranks across gene
Other scoring functions
Step 2
Estimate statistical significance for scores in Step 1
Options
Exploring gene-wise information
Exploiting the power of large number of genes
Step 3
Use a threshold to select the top-ranked statistically significant genes
Step 4
Evaluate gene subset selected in Step 3

Figure 2.5: Illustration of a typical workflow for finding differentially expressed genes (DEGs) or more generally filter ranking methods. For each main step the different options are listed. Figure taken from [Lazar et al. (2012)b].

In the example heatmap in Figure 2.6 for example (see Section 2.3.4), only the top ten discriminating genes are shown. They were selected using the R/Bioconductor `limma` package [Smyth (2004)] as can be seen in the code in Appendix A.2.

Class Discovery

Class Discovery is the search for a biologically relevant unknown taxonomy identified by a gene expression signature or a biologically relevant set of co-expressed genes. Since the aim is to identify and group together similarly expressed genes and then try to correlate the results to biology, it is a typical example of a unsupervised learning task.

The basic methodology for class discovery is clustering. First the data is clustered, based on a chosen clustering method and then the clusters are validated through gene annotations, enrichment analysis (are the clusters enriched by genes from functionally important categories, pathways, or processes), or by replicating the results in other data sets. Clustering techniques can be applied to group samples, genes or both together. Genes can be clustered in order to identify groups of co-regulated genes, spatial or temporal expression patterns, or to reduce redundancy in prediction models. Samples are mostly grouped together to identify new biological classes (i.e. new tumor classes or subtypes of existing classes).

Besides the many clustering algorithms also other less traditional methods are applied for class discovery via microarrays. Principal Component Analysis (PCA) for example is a statistical technique used in various fields, such as face recognition and image compression, and determines the key variables in a multidimensional data set that can explain the differences in observations. Its properties makes it very suitable for microarray data as well. Also matrix factorization methods can be used to reduce the dimension of the data via a decomposition by parts, as was for example reported in [Brunet et al. (2004)].

Class Prediction

In general, the goal of class prediction is to develop a multivariate function for accurately predicting class membership of a new instance and is often referred to as supervised learning. In the context of gene expression analysis the instances are the samples and their class memebership depends on the biological variable of interest and is often called its *phenotype*. Common examples of supervised analysis applications in microarrays are tumor classification, subtype prediction, survival analysis, etc.

The basic methodology for class prediction is to start with two data sets, a training set and test set. Use your training data set to build a classifier, or predictor, based on your chosen classification method and use your test data set to test the classifier. Many classification algorithms already exist in the machine learning and data mining literature and most of them (K-nearest neighbors (KNN), neural network (NN), decision trees, support vector machines (SVM), etc.) also are available in the gene expression analysis tool kits mentioned before.

Pathways Analysis

While a typical class discovery experiment looks for genes that are differentially expressed between two or more conditions, they result very often in long lists of genes which have been selected using some criteria to assign them statistical significance. A common approach to further interpret those lists is to relate the genes it contains with one or more functional annotation databases such as the Gene Ontology (GO, [Ashburner et al. (2000)]), to determine the biological function of the genes, or to pathway databases such as the Kyoto Encyclopedia of Genes and Genomes (KEGG, [Kanehisa & Goto (2000)]).

2.3.4 Visualization Methods for Gene Expression Data

In general, visualization is an essential part of exploring, analyzing and reporting data. High-throughput data, like gene expression measurements, creates many challenges for visualization methods. The dimensions of the expression matrix are high and in most cases it is essential that the data should be mapped to several biological variables of interest. Many visualizations are used in literature with a lot of specific, case-dependent customizations. In this chapter we will illustrate some of the *basic* tools, together with the accompanied R code to obtain them.

Heatmaps

From the beginning, heatmaps were used to represent microarray results [Eisen et al. (1998)]. A heatmap is a graphical representation of data where the individual values contained in a matrix are represented as colors and therefore it is a perfect visualization tool to present high-dimensional data like a gene expression matrix in a structured way, able to show both gene and sample clusterings in a single figure. Mostly, heatmaps are combined by dendrograms on the sides to extra highlight both clusterings. An example heatmap is shown in Figure 2.6. The columns typically correspond with the samples and the rows with the probes/genes. From the heatmap in Figure 2.6 one can immediately observe two clusters of samples, labeled by the biological variable of interest (smoker vs non-smoker).

Multidimensional Scaling

Another useful visualization technique is multidimensional scaling (MDS). Starting from a matrix of all pairwise distances between all samples, the aim of MDS is to arrange all samples in a 2-dimensional Euclidean space such that the distances between the samples are as much *similar as* the given distances as possible [Cox & Cox (2001)]. For all MDS plots in this thesis we used the `cmdscale` R function, which uses the least-squares definition of ”similar”. MDS is very similar to Principal Component Analysis (PCA), and for a comparison the reader is invited to consult Chapter 4 in [Lee & Verleyen (2007)]. An example MDS plot is shown in Figure 2.7. Each circle corresponds to a sample, colored with respect to the biological variable of interest (lung cancer vs control).

Other Methods

Scatterplots can also be used to plot data from two variables but often become dense and uninformative with a high number of observations. Alternatives to describe and visualize distributions are box plots and density plots. Volcano plots can be used to look at fold change and statistical

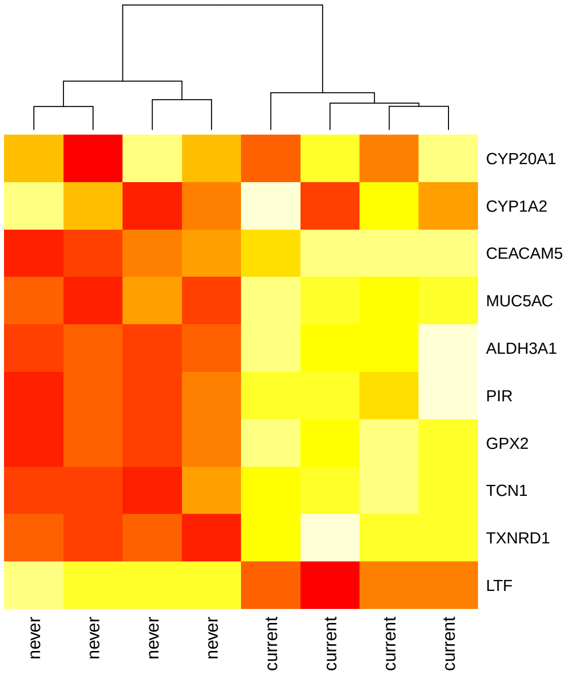


Figure 2.6: Example heatmap of the top discriminating features of dataset `GSE4635`. Expression values are colored from white (low expression) to orange (high expression values), each block represents the expression value of a gene in a specific sample. The samples cluster by `Smoker` status (smokers : `current` and non-smokers: `never`). R code to obtain this figure can be found in Appendix A.2. Example taken from [Taminau et al. (2011)b].

significance simultaneously and also dendrograms are widely used to represent distances between samples and/or genes.

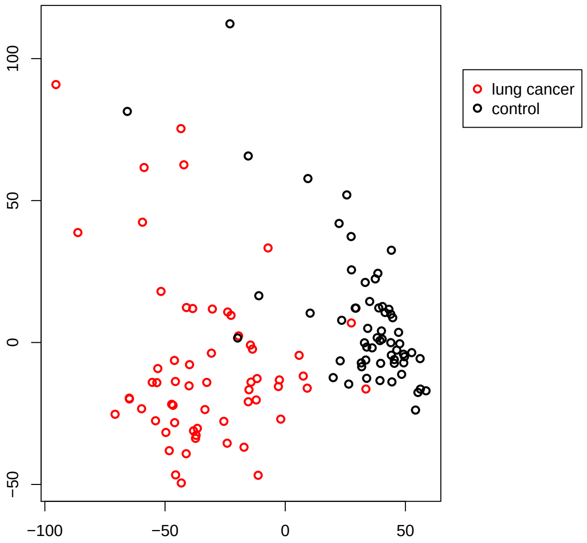


Figure 2.7: Example multidimensional scaling (MDS) plot of dataset `GSE19804`, visualizing the distances between samples within one dataset. Samples are colored corresponding their `Disease` status (lung cancer vs control). R code to obtain this figure can be found in Appendix A.3.

3

Retrieval of Genomic Data

A decade of microarray research has resulted in a plenitude of microarray studies. Many of these studies have been gathered in gene expression data repositories freely accessible for the scientific community. Amongst others are the Stanford Microarray Database (SMD, [Sherlock et al. (2001)]), ArrayExpress [Parkinson et al. (2011)] and Gene Expression Omnibus (GEO, [Edgar et al. (2002)]). Those repositories have to adapt fast to the current needs. GEO for example, established a decade ago as a public repository for high-throughput gene expression data generated mostly by microarray technology, already successfully switched to next-generation sequencing technologies and currently contains over 20000 genomic studies [Barrett et al. (2011)].

With this large amount of genome-wide data available today at each researchers fingertips, scientific results in biomedical research increasingly arise from complex statistical analysis pipelines. This workflow usually involves a wide range of different computational tools and/or software components, each with their specific set of constraints, models and parameters. It is essential that the complete analysis is appropriately and unambiguously described to ensure that researchers can independently reproduce or verify published results. This is unfortunately still not the case, although solutions are proposed [Gentleman et al. (2005), Mesirov (2010)]. In general, there is clearly a pressing need for transparency in computational science [Yale Law School Roundtable on Data and Code Sharing (2010)].

Prior to analysis, the raw data of most genomic experiments is normalized or preprocessed with sophisticated algorithms which are also often not described and documented in sufficient detail. Besides the numerical data, the phenotypical meta-data of the samples are often encoded in spreadsheet software and arbitrary mapped to the individual experiments without any standards or versioning. Moreover, the normalization methods, the gene annotation, and the phenotypic meta-data of the samples change in time as new insights are obtained, and must be kept up-to-date. Finally, the data has to be transformed into the format accepted by the data analysis tools. All those issues are also hindering reproducibility even before analysis is started and require manual intervention. Manual manipulation of genomic data before delivering it to the analysis tools is not reproducible and error-prone, and should be avoided. Without robust error-checking mechanisms and peer involvement, errors can, and do occur [Baggerly & Coombes (2009)].

The primary purpose of the public repositories is to guarantee the integrity of the data, not its usability. This means that the although the data is always available in the form presented in its original publication, it often lacks the necessary information for further analysis by other researchers. A common application of public data is to compare it with new results. In this case the data retrieved from public repositories does not only have to be correct, it also has to be compatible with the new data, e.g. the same up-to-date genomic features, preferably the same preprocessing procedure, etc. Currently this is very hard to achieve. In GEO, for example, datasets can be retrieved as a probe expression matrix without knowing the specific details of the preprocessing method. Blindly combining or comparing these data without verifying if they were preprocessed using the same methodology, can undesirably influence the results of any analysis. We have to note that for some studies in GEO also the raw CEL files are available for download but this is unfortunately still not obligatory. A study in 2006 pointed out that only 48% of all data in GEO and ArrayExpress was submitted with raw data [Larsson & Sandberg (2006)], in addition they also pointed out that only 38% of the data meet necessary quality and data format standards needed for integrative microarray research. Even with the raw CEL files it is for most researchers not a trivial task to perform all complex computational steps required for preprocessing in a consistent and correct way.

Another application of public data is to combine multiple sources in order to discover new or more robust findings. Inconsistently preprocessed data will also severely hinder this approach as will be explained in more detail in Chapter 5 and further.

A clear example of the effects of inconsistently retrieved data can be found in [Baggerly & Coombes (2011)], where they state: *"We were asked if we could implement this approach to guide treatment at our institution; however, when we tried to reproduce the published results, we found that poor documentation hid many simple errors that undermined the approach [...] We spent approximately 1500 person-hours on this issue, mostly because we could not tell what data were used or how they were processed. Transparently available data and code would have made checking results and their validity far easier. Because transparency was absent, an understanding of the problems was delayed, trials were started on the basis of faulty data and conclusions, and patients were endangered. Such situations need to be avoided."* From this example it is clear that not only the analysis part but also the data to start with should be trackable. After all, the longer it takes to retract a flawed clinical paper, the more patients are put at risk [Steen (2011)].

Most online repositories of genomic datasets encourage the use of standards for describing the biological samples. For example for microarray datasets, the Minimum Information About a Microarray Experiment (MIAME) standard has been proposed and established [Brazma et al. (2001)]. Despite its success, MIAME is only a set of guidelines of what information needs to be captured. MIAME does not provide, nor is intended to provide, a format for representing this information and data. Without a standard computer-readable format, the utility of these data is limited [Brazma (2009)]. Other initiatives where proposed as well, such as the XML-based MAGE-ML [Spellman et al. (2002)], but they were not adapted as standards, mainly due to their complexity.

The MIAME standard is particularly successful for describing experimental protocols but is however not sufficient to describe biological sample information or phenotypic meta-data. Because no standard has been accepted yet to address this issue, clinical annotations are not standardized in most genomic repositories. Handling information about the biological samples is still challenging in general. Biomedical ontologies exist such as Unified Medical Language System (UMLS [Bodenreider (2004)]), but these vocabularies are not universally accepted, can be subjective depending on the intended application, change in time as knowledge about the samples advances, or new information about the samples becomes available. Also, as top-down standards, they may not be appropriate for early stages, when the relevant sample data is only starting to be aggregated and understood by the scientific community [Quackenbush et al. (2006)]. The creation of an ontology also requires automatically text-parsing of biological annotation data through text-mining algorithms which is still inefficient and leads to a high rate of errors [Butte & Chen (2006)].

At this point we have to conclude that it is difficult to retrieve genomic data in a consistent and unambiguously way. This is mainly due to the complex preprocessing options for the numerical data which are not always sufficiently documented, and the lack of standards to describe phenotypic meta-data. Moreover, since both numerical data and meta-data can change in time, versioning of the data is necessary but currently absent. In the next chapter the InSilico DB will be presented, a genomic datasets hub that tries to solve those issues, thereby hopefully paving the road towards consistent retrieval of genomic data [Coletta et al. (2012)].

4
The InSilico Database

The InSilico Database (InSilico DB) can be seen as a genomic dataset hub and an efficient starting point for analysing genome-wide studies using existing gene expression analysis software [Coletta et al. (2012)]. Its strength lies in its ability to seamlessly connect genomic datasets repositories to state-of-the-art and free graphical user interface (GUI) and command-line analysis tools. It is a powerful collaborative environment with advanced capabilities for biocuration, dataset management and integration. The InSilico DB is the main result of the inSilico project[1], a inter-university initiative funded by the Brussels Institute for Research and Innovation (INNOVIRIS)[2], in which I participated the last five years. Inside this project we build up the expertise needed to combine all these different aspects of genomic research in order to create this tool.

In the following sections we will describe into more detail the different

[1] `https://insilico.ulb.ac.be`

[2] `http://www.innoviris.be/`

parts of this tool: We will start with an overview of its content and general architecture, next the different functionalities like browsing, exporting and managing datasets will be described briefly, and we end with more technical implementation details. Finally we introduce the `inSilicoDb` R/Bioconductor package [Taminau et al. (2011)b] which was developed to programmatically access the InSilico DB, enabling large-scale analysis of gene expression repositories.

4.1 Overview of the InSilico DB

4.1.1 Content

The content of InSilico DB mainly consists of Microarray (see Section 2.3) and Next-Generation Sequencing (see Section 2.2.2) datasets originating from NCBI Gene Expression Omnibus (GEO)[3], Short Read Archive (SRA)[4], the Cancer Genome Atlas project (TCGA)[5] and the Broad Institute[6]. This data is pre-installed, meaning that is cleaned, preprocessed in a consistent manner and curated, and therefore ready-to-use by the users. To make this possible, several specific pipelines were implemented and controlled by a framework called the *InSilico backbone*. More information on these genomic pipelines will follow in Section 4.3.1.

Currently, InSilico DB supports Illumina microarray platforms, Illumina NGS platforms and most of the Affymetrix microarray platforms. A more detailed list can be found in Table 4.1 or on the projects website[7]. As of April 2012, InSilico DB contains over more than 6000 individual public datasets, accounting for more than 180.000 samples. 3000 datasets and 120.000 samples are manually curated and available for download.

Owing to the accumulated in-house and contributed biocuration efforts,

[3] http://www.ncbi.nlm.nih.gov/geo
[4] http://www.ncbi.nlm.nih.gov/sra
[5] http://cancergenome.nih.gov
[6] http://www.broadinstitute.org
[7] https://insilico.ulb.ac.be/genomics-platforms

ID	*Title*	*Technology*	*Organism*
GPL91	Affymetrix Human Genome U95A	in situ oligonucleotide	Homo sapiens
GPL8300	Affymetrix Human Genome U95 Version 2	in situ oligonucleotide	Homo sapiens
GPL96	Affymetrix Human Genome U133A	in situ oligonucleotide	Homo sapiens
GPL97	Affymetrix Human Genome U133B	in situ oligonucleotide	Homo sapiens
GPL570	Affymetrix Human Genome U133 Plus 2.0	in situ oligonucleotide	Homo sapiens
GPL571	Affymetrix Human Genome U133A 2.0	in situ oligonucleotide	Homo sapiens
GPL3921	Affymetrix HT Human Genome U133A	in situ oligonucleotide	Homo sapiens
GPL6947	Illumina HumanHT-12 V3.0	oligonucleotide beads	Homo sapiens
GPL1261	Affymetrix Mouse Genome 430 2.0	in situ oligonucleotide	Mus musculus
GPL85	Affymetrix Rat Genome U34	in situ oligonucleotide	Rattus norvegicus
GPL1355	Affymetrix Rat Genome 230 2.0	in situ oligonucleotide	Rattus norvegicus
GPL9052	Illumina Genome Analyzer	high-throughput sequencing	Homo sapiens
GPL9115	Illumina Genome Analyzer II	high-throughput sequencing	Homo sapiens
GPL10999	Illumina Genome Analyzer IIx	high-throughput sequencing	Homo sapiens
GPL11154	Illumina HiSeq 2000	high-throughput sequencing	Homo sapiens
GPL13477	Illumina Genome Analyzer IIX	high-throughput sequencing	Homo sapiens

Table 4.1: List of platforms currently supported in InSilico DB. Each different technology requires a new and specific preprocessing pipeline. Information gathered on April 2012

it is possible to map the biological variety of the data inside InSilico DB. A wide variety of tissue types, cancer types, cell lines and control samples are available, making this tool a perfect candidate as basis for large-scale analysis and for comparison to in-house generated datasets. Table 4.2 gives more detailed statistics about the most commonly observed tissues.

4.1.2 Biocuration

As we have seen in the previous chapter, defining a system to structure the totality of the clinical information available for samples and studies is not straightforward. Within InSilico DB we proposed a bottom-up approach where users can structure samples information, starting from unstructured annotations, and define their own vocabulary. Because the curation of a dataset may differ depending on the intended application and context (e.g. smoking as a behavior versus as a carcinogen, or a human cell line as human tissue or not) InSilico DB allows every dataset to have multiple curations.

$TissueType$	$\#Cancers$	$\#Controls$	$\#Others$	$\#Total$
Breast	8664	757	94	**9515**
Brain	1165	1825	4229	**7219**
Bone Marrow	4929	1975	306	**7210**
Lung	2376	1064	1001	**4441**
Liver	831	372	2227	**3430**
Prostate	2243	384	0	**2627**
Colon	1718	254	479	**2451**
Blood	177	648	1386	**2211**
Muscle	3	556	1640	**2199**
Kidney	887	346	811	**2044**
Ovary	1547	106	344	**1997**
Lymph node	1159	58	132	**1349**
Skin	319	278	743	**1340**

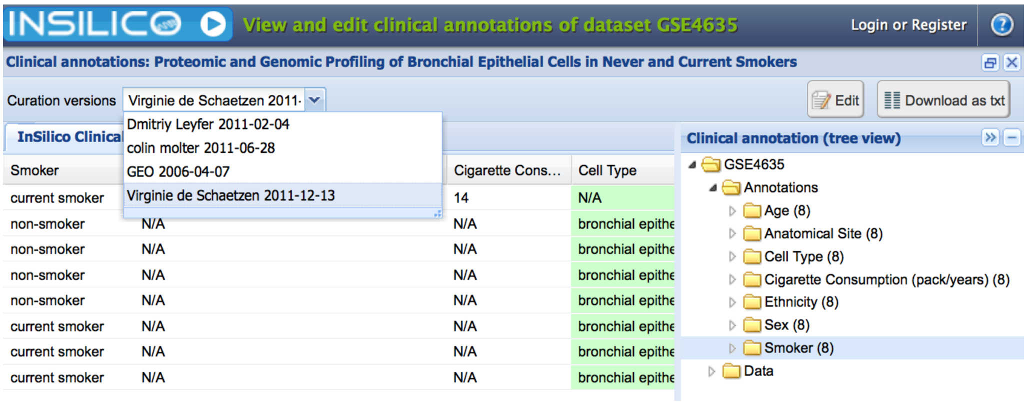
Table 4.2: List of most observed tissue types in InSilico DB. For each tissue type the number of cancer, control and other samples is listed. Information gathered on April 2012

Additionally, InSilico DB accepts batch submissions from independent bio-curation efforts. Batch submissions from the Broad Institute Library of Integrated Network-based Cellular Signature project[8] and from the Gemma initiative [Zoubarev et al. (2012)] have been submitted and added.

In order to facilitate the curation process, an interface to visualize, semi-automatically curate and enrich clinical annotations of genomic datasets was developed. In this interface, pictured in Figure 4.1, information is displayed using both a spreadsheet view and a tree view to make this high-dimensional data *workable*. Curations can be created from scratch or imported from comma-separated-value (CSV) files. In addition, existing curations can be modified or enriched with for example analysis results.

This *collaborative* curation aspect of InSilico DB will hopefully help the community to annotate all publicly available data sets with more complete and structured meta-data. For all of the data sets present in InSilico DB the original authors were contacted and many of them already en-

[8] http://www.broadinstitute.org/LINCS/


INSILICO
View and edit clinical annotations of dataset GSE4635
Login or Register
Edit
Download as txt
InSilico Clinica
Clinical annotation (tree view)

Clinical annotations: Proteomic and Genomic Profiling of Bronchial Epithelial Cells in Never and Current Smokers

Curation versions: Virginie de Schaetzen 2011-

- Dmitriy Leyfer 2011-02-04
- colin molter 2011-06-28
- GEO 2006-04-07
- Virginie de Schaetzen 2011-12-13

Smoker		Cigarette Cons...	Cell Type
current smoker		14	N/A
non-smoker	N/A	N/A	bronchial epithe
non-smoker	N/A	N/A	bronchial epithe
non-smoker	N/A	N/A	bronchial epithe
non-smoker	N/A	N/A	bronchial epithe
current smoker	N/A	N/A	bronchial epithe
current smoker	N/A	N/A	bronchial epithe
current smoker	N/A	N/A	bronchial epithe

- GSE4635
  - Annotations
    - Age (8)
    - Anatomical Site (8)
    - Cell Type (8)
    - Cigarette Consumption (pack/years) (8)
    - Ethnicity (8)
    - Sex (8)
    - Smoker (8)
  - Data

Figure 4.1: Screenshot of the curation interface of the InSilico DB. This view shows a specific curation for the `GSE4635` dataset in both a spreadsheet view (left) and a tree view (right). For this dataset four different curations exist as can be seen from the drop-down menu on the top.

riched or improved the available meta-data.

4.1.3 Architecture

Before explaining the functionalities and implementation details of InSilico DB we provide a conceptual overview of the architecture of the InSilico DB tool. In Figure 4.2 we can observe the global *three-tier-architecture* of the system, consisting of a *Presentation* part dealing with the Graphical User Interface (GUI) of the tool, a *Business Logic* part containing all the models and procedures and finally the *Storage* part which defines how and where all the relevant data is stored.

All three main parts of the InSilico DB will be described in more detail in Section 4.3. The functionality offered via the GUI to the users is illustrated as uses cases in Figure 4.2 and will be listed in more detail in Section 4.2.

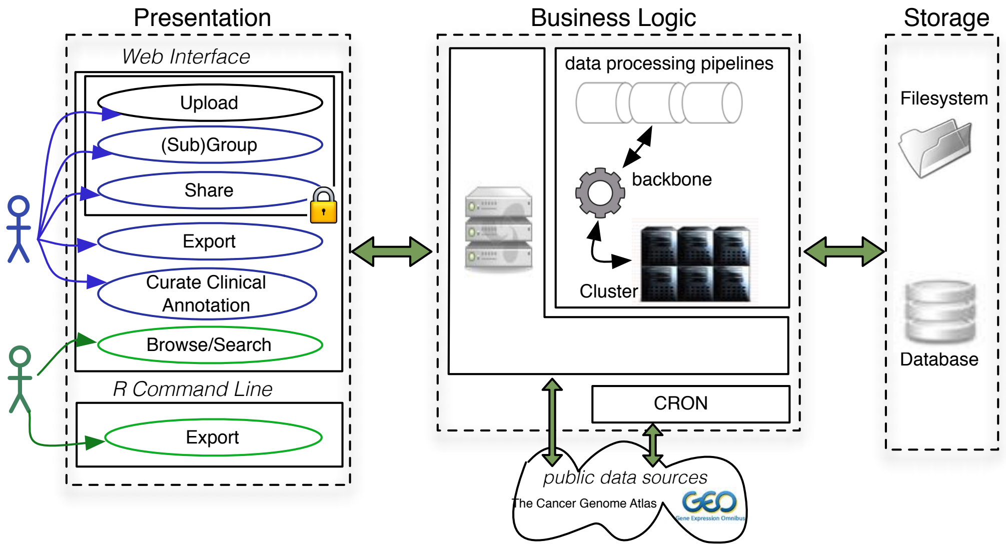


Figure 4.2: An overview of the InSilico DB architecture.

4.2 Functionality

As with most complex tools there are a lot of options and features the user has access to. We will however only describe in this section the main groups of functionality the InSilico DB has to offer:

Search/Browse

Every decent repository of data needs the option to query its content. In the InSilico DB this can be done via a google-like free-text search where the user can query for datasets based on an arbitrary keyword (e.g. thyroid, er status), a GEO platform or study identifier (e.g. GSE4635, GPL570) or a curator. The resulting datasets are listed and the user can browse through this list. In addition, the datasets can be filtered based on public/private status, curation status, platforms, available preprocessings and measurement type.

Export

Genomic datasets can be exported to different analysis tools with one click. Currently supported formats are: R/Bioconductor [Gentleman et al. (2004)], GenePattern [Reich et al. (2006)] and Integrated Genomic Viewer (IGV, [Robinson et al. (2011)]). For microarray data, users can choose to export molecular measurements per platform-specific probes or summarized by genes; and between the normalization provided by the original authors, or a fRMA-renormalisation performed by InSilico DB[9]. For RNA-Seq datasets, users can export gene-expression, splice junctions, transcript expression estimates, and differential expression results. For exome datasets, users can export annotated variants to IGV. Although for every dataset there are multiple options, almost every instance is precomputed by the InSilico backbone and immediately ready for download. If the precomputed dataset is not present in the system (because the dataset was recently imported, there were errors or the curation/annotation was changed recently) it will be generated by the InSilico backbone *on-the-fly* and the user will receive notification when it is ready. The InSilico DB content is also accessible from a programmatic interface that allows for batch queries through the R/Bioconductor package `inSilicoDb` [Taminau et al. (2011)b], see section 4.4.

Upload/Share

InSilico Db is a collaborative platform which allows users to upload private datasets, compare them with public datasets and share them among collaborators. A dedicated sharing interface enables users to define the visibility of their data. This counts for both the numerical expression data as for the clinical annotations. Comparable data management tools are currently lacking.

[9] For microarray datasets there are thus four different export options: probe/original, probe/frma, gene/original and gene/frma.

Curate

See Section 4.1.2 for a detailed description of the curation facilities of the InSilico Db tool.

4.3 Implementation

To discuss the more technical implementation details we will again conceptually break down the tool in the three layers depicted in Figure 4.2.

Presentation

InSilico DB as a tool is presented to the user as a interactive web application, accessible via any web browser. For the development of the web services there was opted for a combination of the `Zend` framework[10], an open source, object-oriented web application framework implemented in PHP, and `ExtJS`[11], a javascript library specialized in development of interactive web applications. The `Zend` framework provides a Model-View-Controller style of coding and is operating on the server-side. `ExtJs` is responsible for the necessary javascript code on the client-side and ensures compatibility with all kind of browsers, a vital requirement for any web application tool.

Business Logic

This layer encapsulates a wide variety of tools and scripts. The main part is developed inside the `Zend` framework and handles user-interactive tasks like querying data, user authentication, etc. The real *logic* sits however behind the scene and is mainly performed through R and Python scripts repsonsible for creating and maintaining all genomic data. Those scripts are bundled in an ordered pipeline and controlled by the InSilico backbone (see Sections 4.3.1 and 4.3.2).

[10] `http://framework.zend.com/manual/en/introduction.overview.html`

[11] `http://www.sencha.com/products/extjs`

Storage

One of the most important aspects of a database or data repository is its data and how it is stored. A decade of microarray technology has resulted in a huge amount of raw data and with the increasing popularity of NGS studies this can and will even increase exponentially. This affects the way data has to be stored physically. From Figure 4.2 could already be noticed that data is stored at two different locations. The meta-data, such as relations, clinical annotations, curation information, etc. is stored in a traditional relation database, enabling complex querying, while the more heavy-weighted, numerical data is currently stored on a filesystem. This filesystem has a strict structure to make automatical interaction with the InSilico backbone possible.

4.3.1 Genomic Pipelines

InSilico DB currently contains more than 180.000 genomic profiles which are all generated and processed in a consistent way. This is ensured by strictly following *genomic pipelines*. A pipeline can be defined as an ordered sequence of preprocessing steps. Each of those steps, as already introduced in Section 2.3.2, requires a number of settings or decisions and for compatibility reasons it is important that those settings are not mixed up.

All genomic data inserted in InSilico DB are associated with a platform type (e.g. Affymetrix U133 Plus 2.0), a measurement type (e.g. RNA-Seq) and parameters the user can select (e.g. fRMA preprocessing). The combination of all these values defines the specific pipeline used to generate all data.

While the overall implementation of the InSilico DB tool can be seen as a collaborative effort of the entire InSilico team we will concentrate in this section on my main personal contribution: the microarray data pipelines. The other genomic pipelines implemented in the InSilico DB mainly deal with the different types of next generation sequencing (NGS) data.

For the proper generation of gene expression microarray data a simplified pipeline is presented in Figure 4.3, this pipeline is part of the actual implementation of the InSilico DB. This pipeline contains a number of tasks or jobs (grey boxes) and dependencies between jobs (black arrows). If job A is dependent on job B (in graph: A $\Rightarrow$ B) it means that job A can't be started before job B was successfully executed.

Every job performing a specific task in the pipeline can be considered as a script or stand-alone program. A job writes its result as a file on the filesystem. The organization of the filesystem (e.g. where each job has to write its results and using which name) is controlled by the InSilico backbone. Using files as communication medium between the different jobs makes it easy to debug and intervene between the different steps of the pipeline. Another advantage is that the scripts can be implemented in any programming or scripting language since the only constrained is that it has to be able to read and write files from/to the filesystem. In practice however, almost all jobs are R/Bioconductor scripts, due to the availability of many high-quality bioinformatics packages.

To further illustrate the pipeline in Figure 4.3 we will now briefly describe the separate jobs in top-down order. The `MEAS_PROBE_FRMA` job will be explained in more detail.

GCT

This job transforms a genomic study, imported as a Bioconductors ExpressionSet (*eset*) found on the filesystem, to a GCT file which can be used as input for the external analysis program GenePattern [Reich et al. (2006)]. For a defined curation of the study all phenotypic annotations are stored in separate CLS files. Both GCT and CLS data file formats are defined and supported by GenePattern[12].

[12] http://www.broadinstitute.org/cancer/software/genepattern/tutorial/gp_fileformats

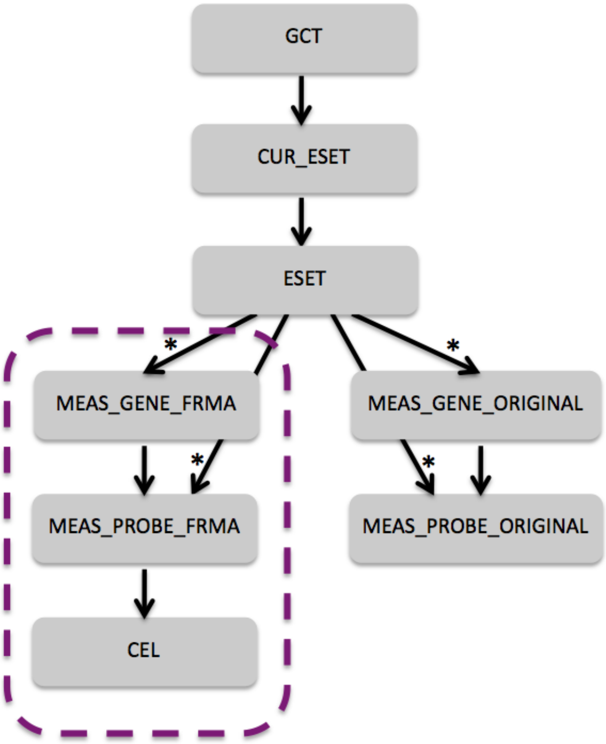


Figure 4.3: Genomic pipeline for microarray data. Each grey box represents a different job. Dependencies between jobs are denoted by black arrows. An arrow with a star (*) means the dependency on multiple jobs (e.g. for a study multiple samples have to be generated). The labeled subgraph on the left will be used as an example to demonstrate the InSilico backbone in Section 4.3.2. `CUR` stands for Curated, `MEAS` stands for Measurement.

CUR_ESET

This job is straight-forward: it loads an eset, retrieves all phenotypic information for a given curation from the database, adds this information to the ExpressionSet data structure and stores the resulting eset on the filesystem.

ESET

Not only complete esets but also the individual samples are stored as files on the filesystem. For this job all samples belonging to the studies are loaded from the filesystem and concatenated to each other to create the ExpressionSet data structure. Additional log information is added to the eset and extra control checks (e.g. have all samples the same number of features?) are executed before storing the newly created eset to the filesystem. As can be seen from Figure 4.3 this job can be dependent on different jobs depending on the specified parameters.

MEAS_GENE_FRMA

The main role of this job is to perform a probe-to-gene mapping. An eset with *probes* as features is loaded and an eset with *genes* as features is stored back to the filesystem. As already explained in Section 2.3.2, multiple probes can represent the same gene and their expression profiles have to be *collapsed* into one expression profile. In our pipeline we do this by taking for every sample the maximum expression value over all probes mapping to the same gene. Alternatives are taking the mean expression value for every sample or simply select the probe with the most variation across all samples. An illustrative example with three probes mapping to the same gene:

```
          sample1  sample2  sample3
 probe1         8       12        8
 probe2         0       11        9
 probe3         7       12        0
------------------------------------
 gene           8       12        9
```

MEAS_PROBE_FRMA

This job stores a preprocessed sample on the filesystem. We will explain this job in detail together with a simplified code example in Appendix A.4. (1) In a first step the necessary Bioconductor annotation package is identified, based on the defined platform. These annotation packages contain the mapping between the manufacturers features (the probes) and a Entrez Gene Identifier (ENTREZ ID). For example, for platform Affymetrix U133 Plus 2.0 the accompanying package is called `hgu133plus.db`[13]. (2) In a next step the raw CEL files are loaded and we perform frozenRMA preprocessing by applying the `frma` function [McCall et al. (2010)], which takes care of both background adjustment, normalization and summarization steps. (3) At this point our samples are stored as esets and we can use the appropriate Bioconductor annotation package to retrieve for each probe its Entrez Gene Identifier, the gene name and the gene description. This information can be used later for the probe-to-gene mapping. (4) The eset is completed with meta-information such as annotation and version of the used packages and finally (5) it is stored on the filesystem.

CEL

This job ensures that the required raw CEL file for a given sample is retrieved from its source outside the InSilico DB system (e.g. from Gene Expression Omnibus [Edgar et al. (2002)]) and stored on the filesystem. In Figure 4.3 we can indeed observe that this job is not dependent on another job.

MEAS_GENE_ORIGINAL

This job is similar to the `MEAS_GENE_FRMA` job and only performs a probe-to-gene mapping. The only difference is its different dependency.

MEAS_PROBE_ORIGINAL

Like already mentioned before, the InSilico DB provides all microarray data both in fRMA preprocessed format as in the format the authors of the study original published it. Similar to the `CEL` job, this job retrieves this original data. InSilico DB gives no guarantee how this data was preprocessed but it is important for reproducing published studies.

[13] http://www.bioconductor.org/packages/release/data/annotation/html/hgu133plus2.db.html

For other genomic data similar pipelines exist but they will not be described in detail. For RNA-Seq data, reads alignment, transcripts and gene expression abundance are computed using the TopHat-Cufflinks and Cummerbund pipelines [Trapnell et al. (2012)]. For exome data, InSilico DB uses the genome analysis toolkit (GATK) ”best practice variant detection method” pipeline [DePristo et al. (2011)].

4.3.2 The InSilico Backbone

To facilitate the task of applying a genomic pipeline on a large number of data simultaneously with minimal or no manual intervention, InSilico DB uses a workflow system developed in-situ. This system, called the *InSilico backbone* controls all jobs of the genomic pipeline and their dependencies. Given the fast evolution of the genomics field, its pipelines and dependencies there is need for such architecture to update and re-run preprocessing pipelines for all associated profiles in a stable, robust and reliable framework. The InSilico backbone uses similar underlying mechanisms as those developed by Ensembl for their eHive solution [Severin et al. (2010)]. Other similar initiatives, more focussed on the development of workflows in particular, are Taverna [Hull et al. (2006)] and Pipeline Pilot[14] [Accelrys Software Inc, San Diego, USA].

The InSilico backbone is implemented in PHP and Java and stores all its information in a MySQL database. The PHP part deals with requests and the Java part is a service that takes care of calling the specific jobs and monitoring them. A job is launched on our in-house cluster by relying on a queue mechanism provided by the `qsub` program, responsible for submitting batch jobs to the Sun Grid Engine. This way jobs that can run concurrently are executed in parallel. When the backbone gets a request for a certain file, it checks if that file has been already generated by its assigned job and if it thus is available on the filesystem. If this is not the case, it checks *recursively*, by following the jobs dependencies, how to generate this file.

[14] `http://accelrys.com/products/pipeline-pilot`

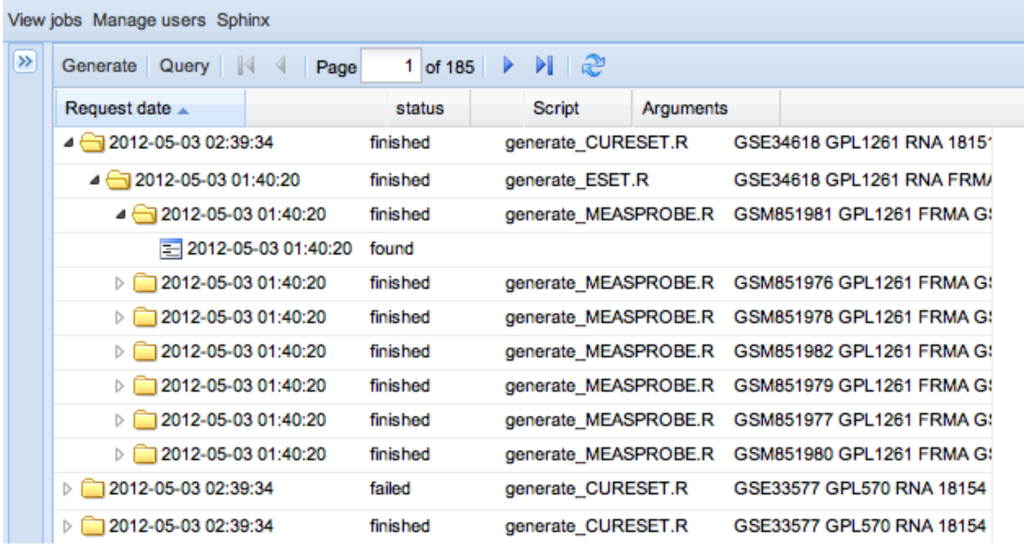
The properties of each job, for example which files it can generate, which files it will need, its dependencies, etc. are stored in a configuration file. In Figure 4.4 is an excerpt from such a configuration file defining the properties of the three jobs from the labeled subgraph in Figure 4.3. For every job X the following properties are defined: X.script and X.args tells the backbone which script has to be executed for this job with which arguments, X.dep defines the dependency of this job, i.e. what job has to be executed prior to this job, X.result and X.path specifies the exact path on the filesystem the result of this job will be written to and finally the X.qsub defines if a job has to be submitted to the qsub system or not.

```
MEAS_GENEFRMA.script   =  ’launchR.sh %scriptdir%/generate_MEASGENE.R’
MEAS_GENEFRMA.args     =  ’%dataName%%norm%%dataValues% %path% %scriptdir%’
MEAS_GENEFRMA.dep      =  ’MEAS_PROBE_FRMA’
MEAS_GENEFRMA.result   =  ’file’
MEAS_GENEFRMA.path     =  ’[...]%dataName%%norm%%dataValues%.RData’
MEAS_GENEFRMA.qsub     =  true

MEAS_PROBEFRMA.script  =  ’launchR.sh %scriptdir%/generate_MEASPROBE.R’
MEAS_PROBEFRMA.args    =  ’%dataName% %platform% %norm% %path% %scriptdir%’
MEAS_PROBEFRMA.dep     =  ’CEL’
MEAS_PROBEFRMA.result  =  ’file’
MEAS_PROBEFRMA.path    =  ’[...]%dataName%%norm%PROBE.RData’
MEAS_PROBEFRMA.qsub    =  true

CEL.script            =  ’DownloadCELFile.py’
CEL.args              =  ’%dataName% %platform% %path%’
CEL.result            =  ’file’
CEL.path              =  ’%cel_dir%/%platform%/%dataName%.CEL’
CEL.qsub              =  true
```

Figure 4.4: Excerpt from configuration file for the InSilico backbone. Properties of the three jobs MEAS_GENE_FRMA, MEAS_PROBE_FRMA and CEL are defined.

Job monitoring can be done through a specific administration interface. The activity of the InSilico backbone is visualized by listing all the current and the past jobs. A tree-view enables the monitoring of job dependencies as well. In Figure 4.5 the top job (CUR_ESET) could be expanded to see the job it is dependent on (ESET), conform to the pipeline in Figure

4.3. The next job is dependent on seven other jobs, for which the resulting files were found on the filesystem. This nicely illustrates the recursive behavior of the InSilico backbone.



Request date	status	Script	Arguments
2012-05-03 02:39:34	finished	generate_CURESET.R	GSE34618 GPL1261 RNA 1815
2012-05-03 01:40:20	finished	generate_ESET.R	GSE34618 GPL1261 RNA FRM
2012-05-03 01:40:20	finished	generate_MEASPROBE.R	GSM851981 GPL1261 FRMA G
2012-05-03 01:40:20	found		
2012-05-03 01:40:20	finished	generate_MEASPROBE.R	GSM851976 GPL1261 FRMA G
2012-05-03 01:40:20	finished	generate_MEASPROBE.R	GSM851978 GPL1261 FRMA G
2012-05-03 01:40:20	finished	generate_MEASPROBE.R	GSM851982 GPL1261 FRMA G
2012-05-03 01:40:20	finished	generate_MEASPROBE.R	GSM851979 GPL1261 FRMA G
2012-05-03 01:40:20	finished	generate_MEASPROBE.R	GSM851977 GPL1261 FRMA G
2012-05-03 01:40:20	finished	generate_MEASPROBE.R	GSM851980 GPL1261 FRMA G
2012-05-03 02:39:34	failed	generate_CURESET.R	GSE33577 GPL570 RNA 18154
2012-05-03 02:39:34	finished	generate_CURESET.R	GSE33577 GPL570 RNA 18154

Figure 4.5: Screenshot of the administration interface of the InSilico DB. This view shows the activity of the InSillico backbone by listing the current or latest jobs. Each job can be expanded to see its dependent jobs.

It is clear that the consistent and large-scale generation of current high-throughput data requires systems like the InSilico backbone, and to conclude this section we will list its advantages:

- *Flexibility and scalability.* New data of interest can be added in real-time without any human intervention. Additions of new genomic pipelines can be added by just adapting the configuration file of the backbone.
- *Intermediate results.* Since communication between jobs is done by files, every file can be seen as an intermediate result in the workflow and decisions can be made to store or delete them. Caching

of intermediate results has however also a drawback since there is a substantial overhead in loading and saving files in the overall workflow.

- *Efficient use of resources.* Only files of interest can be generated on-the-fly, resulting in fewer need for file storage and computation power. Pre-computation is however still possible for jobs that require high computational time and are known to be of interest.
- *Automatic monitoring.* Execution times and outcomes for every job are logged in a database and can be consulted through a specific administration interface.
- *Streamlined error management.* Since the outcome of each script is checked, the state of every file is known. These states can be: ready, executing, waiting, error, etc. Actions can be taken accordingly in a way the integrity of the data is kept.

Both the genomic pipelines and the InSilico backbone play a crucial role in the InSilico DB for providing high quality data.

4.4 The `inSilicoDb` R/Bioconductor Package

In addition to the InSilico DB as an interactive web service a different front-end in R was developed: the `inSilicoDb` package. This package is part of R/Bioconductor [Gentleman et al. (2004)] and was presented as a publicly available tool[15] through a *Bioinformatics* publication [Taminau et al. (2011)b]. The use of this package builds on the Bioconductor project's focus on reproducibility by enabling a clear workflow in which not only analysis, but also the retrieval of verified data, is supported. Through the `inSilicoDb` package, the InSilico DB content is made available for enhanced programmatic access. It enables large-scale genome-wide analysis through automated scripting by seamless integration with the R/Bioconductor genome-wide datasets visualization and

[15] `http://www.bioconductor.org/packages/release/bioc/html/inSilicoDb.html`

analysis platform.

Other and similar software packages to retrieve gene expression datasets in R/Bioconductor exist, like for example GEOquery [Davis & Meltzer (2007)]. However, the information about the samples is in a raw form requiring a manual curation step in transit between a data repository (e.g., GEO) and a data analysis platform (e.g., R/Bioconductor). In contrast, inSilicoDb streamlines this process by providing data verified by the underlying InSilico DB tool.

The basic use of the package is straightforward. To retrieve a dataset:

```
> library("inSilicoDb");
> eset = getDataset(dataset="GSE4635", platform="GPL96");
```

There are three possible outcomes for this request:

- *Ok.* The requested study is available (i.e. precomputed in the InSilico DB system) and is immediately downloaded and stored in the eset variable.
- *Error.* The dataset is not available and can not be generated by the InSilico backbone. This can be due to an internal error, due to corrupted CEL files, etc. A message indicating the error will be provided and the eset variable will be set to NULL.
- *Waiting.* The dataset is not available at this moment but will be generated by the InSilico backbone. A message with the estimated generation time will be provided. After that time the user can retry to download this study. The eset variable will also be set to NULL.

In addition to the dataset name and the platform (both required parameters), the getDataset function also accepts three optional parameters. To choose the normalization, the user can set the norm parameter to "FRMA" or "ORIGINAL". To select the features, the features parameter can be set to "GENE" or "PROBE". See Section 4.2 for more details. An example to demonstrate the use of the features parameter:

```
> library("inSilicoDb");
> eset = getDataset(dataset="GSE4635", platform="GPL96", features="PROBE");
> nrow(eset);
Features
   22283
> eset = getDataset(dataset="GSE4635", platform="GPL96", features="GENE");
> nrow(eset);
Features
   12718
```

The third optional parameter is the `curation` parameter. As already explained in Section 4.1.2 different curations are possible for each dataset. By default, the `inSilicoDb` package always returns the *preferred* curation. To have an overview of all curations an additional function `printCurationInfo` exists:

```
> printCurationInfo(dataset="GSE4635", platform="GPL96");

  INSILICODB: ============================================
  INSILICODB:  curation id: 14926   (preferred)
  INSILICODB: ============================================
  INSILICODB:  curator:   Virginie de Schaetzen
  INSILICODB:  date:      2011-12-13
  INSILICODB:  keywords:  Age,Sex,Anatomical Site,Cell Type,Smoker,Ethnicity,
  INSILICODB:             Cigarette Consumption (pack/years),platform
  INSILICODB:
  INSILICODB: ============================================
  INSILICODB:  curation id: 5092
  INSILICODB: ============================================
  INSILICODB:  curator:   colin molter
  INSILICODB:  date:      2011-06-28
  INSILICODB:  keywords:  Smoker,tissue,status,age,race,sex,pkyrs,history,
  INSILICODB:             patient_id
  INSILICODB:
  INSILICODB: ============================================
  INSILICODB:  curation id: 3743
  INSILICODB: ============================================
  INSILICODB:  curator:   Dmitriy Leyfer
  INSILICODB:  date:      2011-02-04
  INSILICODB:  keywords:  CELL,AGENT
  INSILICODB:
  INSILICODB: ============================================
  INSILICODB:  curation id: 9016
  INSILICODB: ============================================
  INSILICODB:  curator:   GEO
  INSILICODB:  date:      2006-04-07
  INSILICODB:  keywords:  description,title,source_name,characteristics
```

This dataset contains four different curations in the InSilico DB, the first one is labeled as preferred. By specifying a curation id a different curation can be retrieved. (The same example is visualized in Figure 4.1).

```
> eset = getDataset(dataset="GSE4635", platform="GPL96");
> colnames(pData(eset))
[1] "Age"                                   "Sex"
[3] "Anatomical Site"                       "Cell Type"
[5] "Smoker"                                "Ethnicity"
[7] "Cigarette Consumption (pack/years)"    "platform"

> eset = getDataset(dataset="GSE4635", platform="GPL96", curation="9016");
> colnames(pData(eset))
[1] "description"     "title"           "source_name"     "characteristics"
```

The `getDataset` returns an ExpressionSet object, the standard R/Bioconductor data structure containing both numerical, feature and phenotypical information. It is also possible to only retrieve phenotypical information with the `getAnnotations` function.

To make the returned ExpressionSet objects more trackable, versioning information is added to the data structure:

```
> eset = getDataset(dataset="GSE4635", platform="GPL96", curation="9016");
> notes(eset)
$hgu133aVersion
[1] "2.5.0"

$measurementType
[1] "RNA"

$inSilicoCurationId
[1] "14926"
```

In GEO, a Series is composed of (the same) samples assayed on one or more platforms. In InSilico DB, the series are conveniently represented by multiple datasets. Two auxiliary functions to allow flexible management of studies with multiple platforms are provided: `getDatasets` to retrieve, for a given series, all gene expression matrices, and `getPlatforms` to retrieve all platforms:

```
> getPlatforms(dataset="GSE781");
[1] "GPL96" "GPL97"

> esets = getDatasets(dataset="GSE781");
> sapply(esets, annotation);
[1] "hgu133a" "hgu133b"
```

Finally, a function to query the InSilico DB is provided: `getDatasetList`. This function retrieves the identifiers of all datasets in the InSilico DB. These identifiers can be used as input for the other functions. It is possible to limit the search to only datasets satisfying certain constraints such as a given platform, manually curated or not, fRMA preprocessing available or by querying for a specific keyword. The following example illustrates some of the possibilities of this search:

```
#Retrieve ALL datasets
> lst = getDatasetList()
  INSILICODB: 6120 datasets found.

#Retrieve only datasets for GPL570 platform which are curated
> lst = getDatasetList(platform="GPL570", curated="TRUE")
  INSILICODB: 1588 datasets found.

#Retrieve all fRMA preprocessed datasets dealing with thyroid cancer
> lst = getDatasetList(norm="FRMA", query="thyroid")
  INSILICODB: 42 datasets found.
```

In Table 4.3 all functions with their required and optional parameters are summarized. In the code fragment of Appendix A.1 we could already see the combination of using the `getDatasetList` and `getAnnotations`. In the remainder chapters more examples of large-scale analysis using the `inSilicoDb` package will be provided.

To conclude, the `inSilicoDb` R/Bioconductor package provides an efficient means of performing large-scale genomic analysis on the large and growing amount of gene expression profiles using automated scripting. The underlying InSilico DB framework allows search and browsing of curated datasets that can then be automatically retrieved, adding a means for reproducible data sourcing to the reproducible analysis platform R/Bioconductor. As of July 2012, the `inSilicoDb` package is downloaded more than 1500 times[16] and is already used in large-scale gene expression analysis published in high-quality journals [Tomás et al. (2012), Tamayo et al. (2011)].

[16] `http://www.bioconductor.org/packages/stats/bioc/inSilicoDb.html`

Function	*Req. Param*	*Opt. Param*
getDatasetList		platform, norm, query, curated
getDataset	dataset, platform	norm, features, curation
getDatasets	dataset	norm, features, curation
getPlatforms	dataset	
getAnnotations	dataset, platform	curation
printCurationInfo	dataset	

Table 4.3: Summary of all relevant functions in the `inSilicoDb` package with their Required and Optional parameters. More details can be found in the examples in the main text.

5
Integrative Analysis of Microarray Data

In Chapter 3 the retrieval of microarray data from public repositories was already discussed. These repositories contain a plentitude of data gathered in more than a decade of microarray gene expression research and it is one of the challenges for the near future to integrate this vast amount of data [Moreau et al. (2003),Larsson & Sandberg (2006),Sarmah & Samarasinghe (2010)].

In this chapter we first present the two main approaches or frameworks for conducting integrative analysis, further defined as meta-analysis and merging, and explain their main differences. The rest of this chapter consists of a discussion on the advantages and disadvantages of integrating individual gene expression microarray data sets. In the following chapters we go into further details for both the meta-analysis approach in Chapter 6 and the merging approach in Chapter 7 and finally we present an empirical comparison of both approaches in Chapter 8.

5.1 Terminology and Description

In literature it is not always clear what exactly is meant by *integrative-, comparative-, meta-,* or *large-scale* analysis in titles or descriptions of genomic studies. Before going into more details on the specific approaches we provide some definitions of the specific terms we will use in the following chapters:

Integrative Analysis: *- Combining the information of multiple, independent but related studies in order to extract more general and more reliable conclusions.*

This definition is very vague on purpose and only reflects the fact that several independent microarray studies should be combined in order to fully explore the potential of the wide variety of gene expression data available nowadays. For integrative analysis two main approaches exist, mainly differing in how, or more precisely at what level, the different studies are *combined*. We start by providing a definition for both approaches, followed by a more detailed comparison.

Meta-Analysis: *- The use of statistical techniques to combine multiple results, each derived from individual studies, into one more general result.*

To illustrate this framework a schematic overview of a typical meta-analysis to retrieve gene lists from microarray data can be found in Figure 5.1(a).

Merging: *- The combination of multiple data sets into a global data set on which results can be derived with more statistical power.*

A schematic overview of integrative analysis through merging to retrieve gene lists from microarray data can be found in Figure 5.1(b).

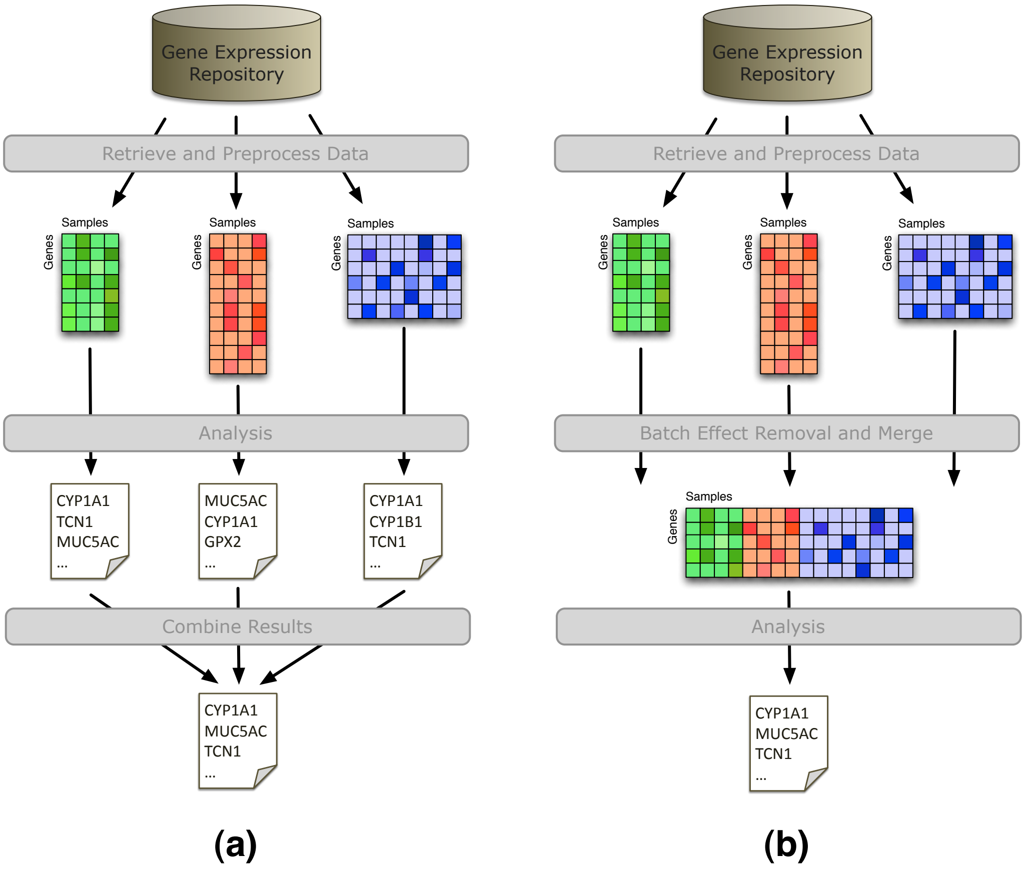


Figure 5.1: Schematic overview of the two main approaches of integrative microarray analysis in the context of the identification of differential genes (DEGs). (a) meta-analysis first derives results from each individual study and then combines the results. (b) merging first combines the data and then derives a result from this large data set.

Several other definitions or approaches exist in literature. In [Larsson et al. (2006)] for example they speak about *Comparative Microarray Analysis* to describe meta-analysis of microarrays. Moreover they refine meta-analysis to be summarizing, hypothesis-driven and exploratory. In their study they however conclude that *"[...] integration of microarray data from different studies or in comparison with whole data repositories could significantly refine the conclusions and interpretations obtained from single microar-*

ray experiments." [Larsson et al. (2006)], thereby sharing our point of view on the benefits of integrative analysis of microarray data.

Another terminology can be found in [Sarmah & Samarasinghe (2010)], where they refer to meta-analysis as *integration at the interpretation level* and to merging as *integration with rescaling of the expression values*.

5.2 Benefits of Integrative Analysis

One of the current limitations of microarray gene expression analysis is the fact that findings are not always reproducible in other independant studies and are, due to their small sample sizes, not robust to the mildest of data perturbations [Michiels et al. (2005), Ein-Dor et al. (2005)]. Typically tens of thousands of probes/genes are investigated in only tens or hundreds of biological samples, making microarray analysis extra vulnerable to the *curse of dimensionality* [Somorjai et al. (2003)]. For example, to generate robust gene signatures for predicting outcome of disease, actually thousands of samples are needed [Ein-Dor et al. (2006)].

Combining information from multiple existing studies can increase the reliability and generalizability of results. Through the integrative analysis-of microarray data we can increase the statistical power to obtain a more precise estimate of gene expression results, immediately overcoming the problem of the low sample sizes. At the same time the heterogeneity of the overall estimate is assessed, making results more generalizable. This way we avoid the danger of study-specific findings only applicable for example to patients or samples of a specific geographical region.

Besides strengthening and extending the results gathered from individual studies, integrative analysis can also provide a broader picture of gene expression in various biological processes since it is not limited to the specific objectives and constraints of individual studies. Moreover, it can also compensate for possible errors and undesired biases in individual studies.

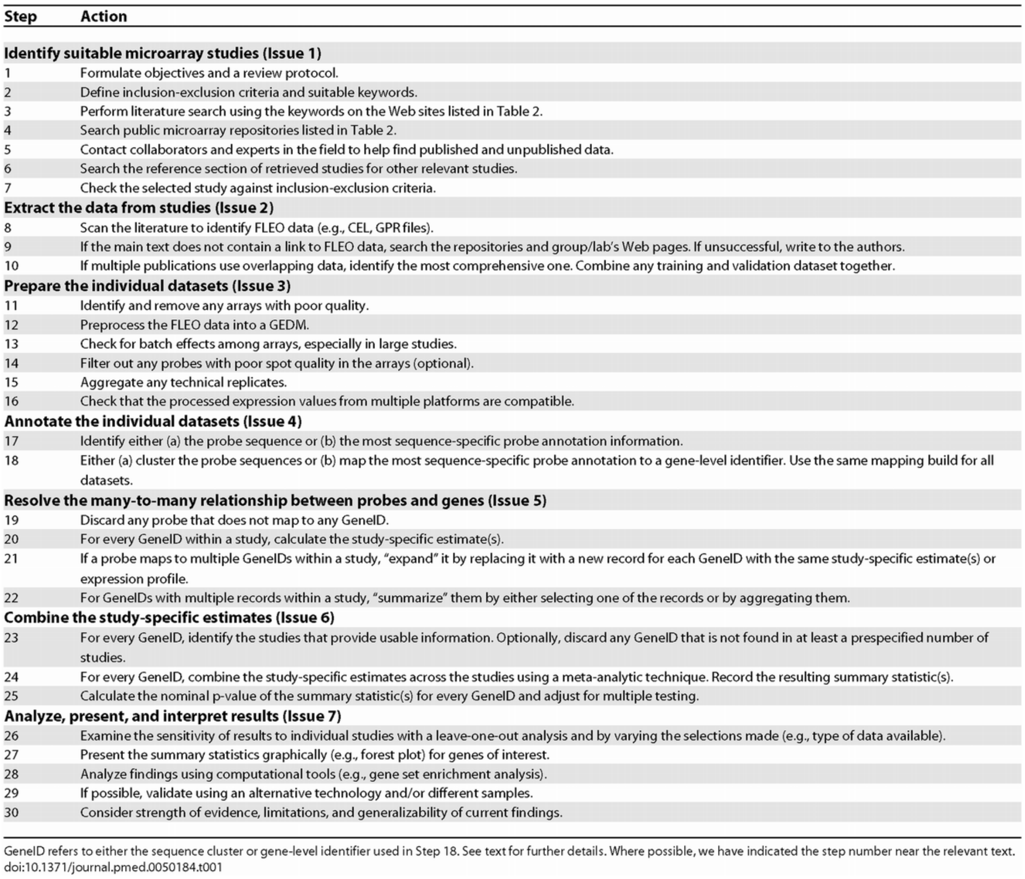
Integrative analysis is also a relatively easy and inexpensive way of gaining new biological insights since it makes comprehensive use of already available data, accumulated through the years by various groups all over the world.

5.3 Issues of Integrative analysis

Based on the definitions of integrative analysis through merging or meta-analysis, both approaches may sound easy and straightforward to conduct, but several issues always arise. In Figure 5.2 some key issues in conducting meta-analysis already identified by [Ramasamy et al. (2008)] are shown. While specific for meta-analysis, many of them are also relevant for integrative analysis via merging.

From the list of seven key issues taken from Figure 5.2 it is remarkable that the first five issues are completely related to data acquisition or retrieval of microarray data, together with its phenotypical meta-data. Issues like how to consistently extract the data from a given study (Issue 2), how to annotate the individual data sets (Issue 4) and how to resolve the many-to-many relationship between probe and genes (Issue 5) were already identified in Chapter 3 as issues for the analysis of single microarray studies. The InSilico DB, described in Chapter 4, provides already the solution for those issues. Moreover, by providing consistently preprocessed and expert-curated data, the InSilico DB also facilitates the preparation of data sets from different platforms (Issue 3) and its extensive browse and search capabilities successfully guide the identification process for suitable studies (Issue 1).

The last two issues are more study-specific. How the different results are combined (Issue 6) depends on the type of information that is available and requested while the final analysis and reporting (Issue 7) depends on the main objective of the integrative analysis study. In Chapter 6 we will describe the possible ways of combining results in the context of meta-



Step	Action
Identify suitable microarray studies (Issue 1)	
1	Formulate objectives and a review protocol.
2	Define inclusion-exclusion criteria and suitable keywords.
3	Perform literature search using the keywords on the Web sites listed in Table 2.
4	Search public microarray repositories listed in Table 2.
5	Contact collaborators and experts in the field to help find published and unpublished data.
6	Search the reference section of retrieved studies for other relevant studies.
7	Check the selected study against inclusion-exclusion criteria.
Extract the data from studies (Issue 2)	
8	Scan the literature to identify FLEO data (e.g., CEL, GPR files).
9	If the main text does not contain a link to FLEO data, search the repositories and group/lab's Web pages. If unsuccessful, write to the authors.
10	If multiple publications use overlapping data, identify the most comprehensive one. Combine any training and validation dataset together.
Prepare the individual datasets (Issue 3)	
11	Identify and remove any arrays with poor quality.
12	Preprocess the FLEO data into a GEDM.
13	Check for batch effects among arrays, especially in large studies.
14	Filter out any probes with poor spot quality in the arrays (optional).
15	Aggregate any technical replicates.
16	Check that the processed expression values from multiple platforms are compatible.
Annotate the individual datasets (Issue 4)	
17	Identify either (a) the probe sequence or (b) the most sequence-specific probe annotation information.
18	Either (a) cluster the probe sequences or (b) map the most sequence-specific probe annotation to a gene-level identifier. Use the same mapping build for all datasets.
Resolve the many-to-many relationship between probes and genes (Issue 5)	
19	Discard any probe that does not map to any GeneID.
20	For every GeneID within a study, calculate the study-specific estimate(s).
21	If a probe maps to multiple GeneIDs within a study, "expand" it by replacing it with a new record for each GeneID with the same study-specific estimate(s) or expression profile.
22	For GeneIDs with multiple records within a study, "summarize" them by either selecting one of the records or by aggregating them.
Combine the study-specific estimates (Issue 6)	
23	For every GeneID, identify the studies that provide usable information. Optionally, discard any GeneID that is not found in at least a prespecified number of studies.
24	For every GeneID, combine the study-specific estimates across the studies using a meta-analytic technique. Record the resulting summary statistic(s).
25	Calculate the nominal p-value of the summary statistic(s) for every GeneID and adjust for multiple testing.
Analyze, present, and interpret results (Issue 7)	
26	Examine the sensitivity of results to individual studies with a leave-one-out analysis and by varying the selections made (e.g., type of data available).
27	Present the summary statistics graphically (e.g., forest plot) for genes of interest.
28	Analyze findings using computational tools (e.g., gene set enrichment analysis).
29	If possible, validate using an alternative technology and/or different samples.
30	Consider strength of evidence, limitations, and generalizability of current findings.

GeneID refers to either the sequence cluster or gene-level identifier used in Step 18. See text for further details. Where possible, we have indicated the step number near the relevant text.
doi:10.1371/journal.pmed.0050184.t001

Figure 5.2: List of key issues for conducting meta-analysis of microarray gene expression data identified by [Ramasamy et al. (2008)] . For each issue a number of practical steps are provided. Figure taken from [Ramasamy et al. (2008)].

analysis into more detail.

In [Sarmah & Samarasinghe (2010)] a more extensive enumeration of issues for integrative analysis is presented. One of their issues is often neglected and questions the effect of the *excluded genes*, i.e. the genes that are not common in all data sets used in the large-scale analysis. In the merging approach the merged data set only consists of genes common between all studies and the more different platforms used, the smaller the intersection of genes will be. Assessing the effect of these excluding genes on the analysis results is indeed a very important concern which didn't receive much attention so far. For the meta-analysis approach it depends on the specific implementation if and how many genes will be discarded for further analysis.

We have to note that for the merging approach an additional issue, not mentioned in Figure 5.2, should be included. Before combining the expression values of different studies, they have to be made *compatible* to each other. Since the use of different experimentation plans, platforms and methodologies by different research groups introduces undesired batch effects in the gene expression values [Leek et al. (2010), Scherer (2009)] an additional transformation of the data to remove those effects is needed. In Chapter 7.3.2 we give an extensive overview of those methods.

In a recent study the question if the advantage of increased sample size is outweighing the disadvantages of merged data sets was addressed for the survival prediction in breast cancer [Yasrebi et al. (2009)]. They identified the following issues when merging different data sets: diversity of microarray platforms, heterogeneity of sample cohorts and reduced number of genes in the merged data sets. However, predictors derived from the merged data sets were robust, consistent and reproducible, and helped to better understand the biases and shortcomings of the individual studies [Yasrebi et al. (2009)].

5.4 Summary

The vast amount and increasing number of publicly available gene expression studies provide strong motivation for the integrative analysis of microarray data. Combining data or results from different studies carries the potential towards higher accuracy, consistency and robustness of findings. The integrated result often offers a more complete and generalizable insight in the biological processes.

Two possible approaches or frameworks for integrative analysis that are currently in practice are presented. Their main difference lies at the level at which information is combined; in meta-analysis information is combined at the result level while merging already combines the numerical data before deriving results. Both approaches will be further described in detail in the following chapters.

6
Meta-Analysis

In this chapter we will go further into detail on the meta-analysis approach for conducting integrative analysis of gene expression microarray studies. The statistical foundations of this set of techniques were already described long before microarray technology even exists and it was adopted quickly after the first online repositories were populated with similar and related studies.

The choice of the specific method for combing the results in a typical meta-analysis is important and depends on the type of response (e.g. binary, continuous, survival) and the final objective of the study. The identification of differentially expressed genes (DEGs) between two well-known conditions is still one of the most fundamental applications of microarrays and was therefore the focus and objective of most successful meta-analysis studies reported in literature. In Section 6.1 we provide an extensive state-of-the-art by describing several methods for conducting meta-analysis in the context of the identification of DEGs and group them in four generic approaches.

In the following Section 6.2 we briefly discuss what to do in other situations and/or objectives and we end with presenting a large-scale meta-analysis of gene expression microarray studies to identify robust and stable genes in Section 6.3. This study consists of more than 34000 different samples from 365 studies made available by the InSilico DB.

6.1 Meta-Analysis Methods for the Detection of Differentially Expressed Genes

To combine the information of multiple gene expression studies in the context of the identification of differentially expressed genes (DEGs) many different strategies are proposed. They can be grouped in four generic approaches, as identified by [Ramasamy et al. (2008)]. In order to provide a well covered state-of-the-art of the different meta-analysis procedures, we will describe each category into more detail.

Combining p-values

In individual studies, many statistical tests for the detection of DEGs can be applied. These tests usually score genes by reporting p-values, expressing the probability that the observed level of differential expression is significant or could have occurred by change. Once those p-values, or the adjusted p-values as discussed in Section 2.3.3, are obtained for each gene in each study, they can be combined. Some methods (also called *omnibus procedures* [Hedges & Olkin (1985)]) can then be used to test the statistically significance of the p-values of a given gene across all studies.

In one of the first real applications of meta-analysis for microarrays [Rhodes et al. (2002)] four data sets were combined to determine genes that are differentially expressed between benign and malignant prostate tissues, e.g the classical normal versus tumor case. They used a variant of Fisher's method, which sums the logarithm of the p-values across all four studies, obtained by a one-sided hypothesis test.

In general, using this approach of combining p-values, for each gene i a summary statistic S_i can be obtained by:

$$S_i = \sum_{k=0}^{n} log(p_i^k) \tag{6.1}$$

with p_i^k the p-value of gene i in study k, obtained after a statistical test. A final list of differentially expressed genes can then be selected by sorting the summary statistic and applying a specific cut-off value.

Although omnibus procedures are straightforward and versatile, they have the disadvantage that by working only with p-values, the actual level of differential expression is ignored as well as the fact if the differential expression is positive or negative. Because of this drawback new approaches emerged.

Combining Effect Sizes

Meta-analysis based on the *t*-statistic was already reviewed in 1999 by [Normand (1999)] in the broader context of biostatistical applications. A few years later this framework was adopted for the first time for microarray analysis [Choi et al. (2003)] to identify differentially expressed genes between tumor and non-tumor tissues in liver and prostate cancer. Their method estimates the *effect size* of gene i as the measure of differential expression, and earlier defined as the standardized mean difference [Hedges & Olkin (1985)]:

$$e_i = \frac{\overline{x}_i^T - \overline{x}_i^N}{\sigma_i} \tag{6.2}$$

with $\overline{x}_i^T$ and $\overline{x}_i^N$ representing the means of the tumor and normal groups for gene i respectively, and σ_i indicating an estimate of the pooled standard deviation.

Once the study-specific effect sizes for each gene are obtained they proposed a hierarchical modeling approach to assess both intra- and inter-study variation when combining effect sizes across the multiple data sets. Their model-based method estimates the overall effect size as the measurement of the magnitude of differential expression for each gene through parameter estimation and model fitting, see [Choi et al. (2003)] for more information.

In a later comparative study it was concluded that although this approach greatly improve over the individual studies it was outperformed by ranking methods based on sensitivity and selectivity criteria, because it usually suffers from a potentially large amount of false positives [Hong & Breitling (2008)].

Combing effect sizes is also used by others. In [Grützmann et al. (2005)] they used it to reveal that four different pancreatic cancer gene expression data sets shared a significant number of up- and down-regulated genes, independent of the technology used. In [Elo et al. (2005)] they proposed a small correction factor to calculate the estimate of the effect size, also known as the Hedges' adjusted g [Hedges & Olkin (1985)] which provided more stable results, especially in the case of small sample sizes.

Vote Counting

Vote counting is another very straightforward meta-analysis technique and in the context of microarray it was described in [Rhodes et al. (2004)]. For the identification of DEGs, each study can *vote* if a specific gene is considered differentially expressed or not and all votes are combined by counting them. Many adaptations to this scheme exist, for example weighting of the votes [Xu et al. (2008)].

Despite the fact that it is well known that vote counting is statistically inefficient [Friedman (2001)], naive vote counting is still selected as preferred tool in many biological applications, mainly for its convenience and simplicity [Tseng et al. (2012)].

Combining Ranks

Methods to combine robust rank statistics were introduced to alleviate the problem of outliers in methods combining p-values. Another advantage is the fact that the rank product is a non-parametric statistic. Instead of p-values or effect sizes, the ranks of differential expression evidence are calculated for each gene in each study. The test statistic can then be calculated as the product, mean or sum of ranks from all studies.

In [Hong et al. (2006)] the more advanced RankProd method was proposed that calculates the products of ranks of fold change in each intergroup pair of samples. In their comparative study earlier mentioned, they showed its better performance as compared to combining p-values and effect sizes [Hong & Breitling (2008)].

When only the (ordered) DEG lists per study are available, various rank aggregation methods were proposed [DeConde et al. (2006)], which were developed to combine lists from literature.

6.2 Meta-Analysis in Other Situations

In the previous section we focussed on the identification of DEGs as main objective. For other applications of microarray analysis like class discovery and prediction, the same approaches are generally applicable for combining results in individual studies. In the next section for example we used a method very similar to the method of combing effect sizes proposed by [Choi et al. (2003)] by only changing the implementation of the effect size for each gene.

Besides the objective of the analysis, also the data that is available can limit the possible techniques that can be used. For example, is the expression data available in order to calculate p-values and effect sizes, or are only the resulting gene lists available for each individual study? A number of other considerations for selecting the best approach were also proposed by [Ramasamy et al. (2008)]. What is for example the the ability of each techniques to handle studies with very low sample sizes? What to do with different platforms consisting of a different set of genes? etc.

It is clear that the actual implementation of a complete meta-analysis study is very customizable, depending on the specific details of the individual studies. The overview of the methods described in the previous section is thus far from complete. It can however serve as a good starting point for the selection of an appropriate technique but needs adaption in order to answer the specific objective of a specific study, as will be illustrated in the following section.

6.3 Identification of Stable Genes Through Meta-Analysis

In this section an application of the meta-analysis approach for integrative analysis of gene expression data is presented. A large-scale screening of 365 microarray studies was conducted to identify potential stable genes, i.e. genes with a constant expression across different tissues and biological conditions. We were able to identify a compact and diverse set of 12 promising stable genes which can be used as reference.

6.3.1 Introduction

Obtaining a set of genes with constant expression among different kind of tissues or experimental conditions is increasing in importance in current biomedical research since they can help to tackle the difficult and often subtle problem of normalizing various kinds of biological data, like gene expression microarray analysis or real-time reverse transcription polymerase reaction (qRT-PCR) [Vandesompele et al. (2002), Bär et al. (2009)]. Those stable genes, or reference genes, can also be used for validation purposes, for example for different normalization methods in the context of gene expression meta-analysis [Autio et al. (2009)]. In general, they provide a background model to compare other measurements against.

Initially, housekeeping genes were selected as natural candidates for the role of reference genes. By definition, housekeeping genes are ubiquitously expressed in all different types of cells [Zhu et al. (2008)], which reflects the biological concept of genes maintaining the basic functionality of the cell [Butte et al. (2001)]. However, numerous studies have shown that those housekeeping genes are actually regulated and their expression levels varies under certain experimental conditions [Cheng et al. (2011)].

The increasing amount of microarray analysis is an excellent source for the identification of genes with stable expression. Initially only single studies or a limited set of studies were used to retrieve reference genes [She et al. (2009), Szabo et al. (2004), Lee et al. (2007), Popovici et al. (2009)], mostly only for specific biological conditions. Recently also large-scale analysis of microarray data was performed to identify stable genes [de Jonge et al. (2007), Cheng et al. (2011)]. We indeed believe that the vast amount of gene expression data available offers a unique opportunity to increase the statistical power of large-scale analysis and is key in obtaining stability and generality among proposed stable gene lists. In this study we use a cohort of 365 studies retrieved from InSilico DB (see Chapter 4) consisting of in total more than 34000 samples, representing a wide variety of different biological conditions. Following the methodology presented in the next section, we end up with a list of 12 potential reference genes.

6.3.2 Methodology

Our methodology to screen for stable genes is illustrated in Figure 6.1 and is mainly a refinement of the more general methodology for meta-analysis presented in Figure 5.1(a).

From Figure 6.1 we can see that our methodology consists of four main

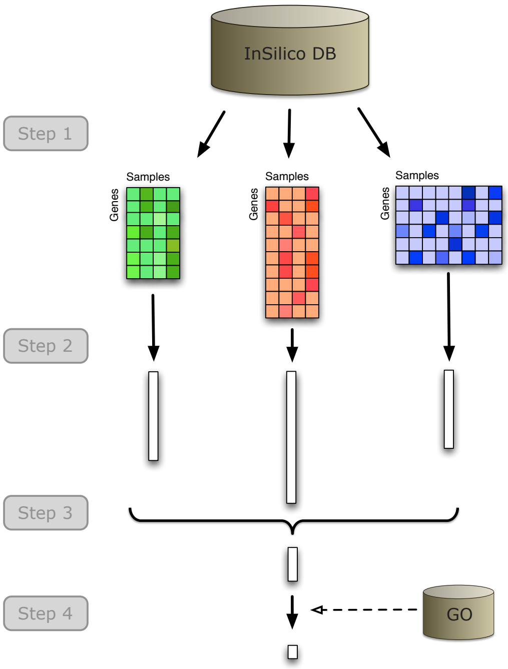


Figure 6.1: Schematic overview of the workflow for the identification of stable genes through meta-analysis in different steps. (Step1) retrieval of studies (Step2) calculate stability scores for each gene (Step3) combine stability scores per gene (Step4) filter based on semantic similarity. All steps are explained in more detail in the text. GO: Gene Ontology.

steps:

Step 1: Retrieval of appropriate studies

To select all individual studies for this large-scale analysis we used the query possibilities of InSilico DB. Using the `getDatasetList` function from the `inSilicoDb` package (see Section 4.4) we retrieved all studies with the following two constraints:

- Hybridized on the Affymetrix Human Genome U133 Plus 2.0 platform (GPL570).

- Contains at least 30 samples in order to be considered as statistically relevant.

This way we retrieved a list of 365 microarray gene expression studies, containing samples from a diverse set of tissues (breast, colon, blood, bone, lung, etc.) and cell lines under varying conditions. This list was retrieved programmatically without any manual interaction[1]. All studies together sum up to a total of 34578 samples.

Step 2: Calculate Stability Scores

For each gene in each data set a score can be calculated to measure its stability in each data set. We opted for the *Coefficient of Variation* (CV), previously used in [de Jonge et al. (2007)], and defined for gene i as follows:

$$CV_i = \frac{\sigma_{x_i}}{\overline{x}_i} \tag{6.3}$$

with $\overline{x}_i$ the mean expression of gene i and σ_{x_i} the standard deviation expression of gene i. By using the CV measure, we searched for genes that are expressed but have low variability in their expression in each available data set. This way we avoided lowly expressed genes which have a more stable expression by definition but are not usable or preferred as reference genes.

Step 3: Combine Stability Scores

Once the CV for each gene was calculated for each individual data set, we define for each gene the global stability score as the median value of all CV measures for that gene across all data sets, further referred to as CV^*. Based on this CV^* measure all genes can be ranked and the top 100 ordered genes were kept for further analysis.

[1] We have to note here that although the entire code to retrieve this data set list is reproducible, if we run the exact code again at a later point in time, we will obtain a different and larger list of studies due the almost weekly increasing content of InSilico DB.

Step 4: Semantic Similarity Filtering

In a last step we excluded genes that were too closely related or similar from a biological point of view to genes that were more stable. Starting in decreasing order, the semantic similarity of each of the genes of the top 100 list obtained in Step 3 was calculated with each higher ordered gene. When a lower ranked gene had a high semantic similarity with a higher ranked gene, the lower ranked gene was removed from the set.

To calculate the semantic similarity between two genes we adopted a measure to estimate the functional similarities of genes based on gene annotation information from Gene Ontology (GO), proposed by [Wang et al. (2007)]. The GO ontology provides a systematic language, or ontology, for the consistent description of attributes of genes and gene products, in three key biological domains that are shared by all organisms [Ashburner et al. (2000),Gene Ontology Consortium (2008)]:

- Cellular Component (CC): the parts of a cell or its extracellular environment.
- Molecular Function (MF): the elemental activities of a gene product at the molecular level, such as binding or catalysis.
- Biological Process (BP): the operations or sets of molecular events with a defined beginning and end, pertinent to the functioning of integrated living units: cells, tissues, organs, and organisms.

The graph-based method of [Wang et al. (2007)] determines the semantic similarity of two GO terms based on both the locations of these terms in the GO graph and their relations with their ancestor terms, for each subontology (MF, BP or CC). This method is available through the `mgoSim` function from the R/Bioconductor package `GoSemSim` [Yu et al. (2010)].

To calculate our semantic similarity we take an average of this method for molecular function (MF) and biological process (BP) and ignore the cellular component (CC) information. The details of our implementation can be found in Appendix A.5.

By applying this last step we reduced the ordered list of the 100 most stable genes into a compact and diverse list of 12 genes.

6.3.3 Results and Discussion

After applying the complete workflow described in the previous section we obtained a list of 12 genes:

```
> reference_genes
[1] "RPL37A" "EEF1A1" "TPT1"    "ATP13A5" "DCAF6"  "ACTG1"
[7] "OAZ1"   "CALM2"  "SH3KBP1" "MATR3"   "COX4I1" "FTL"
```

Although we considered a list of 100 stable genes (Step 3) we obtain a very compact and diverse final list of genes after applying the fourth and last step, the semantic filtering. To motivate the importance of this last step, let us first have a look at the top 100 genes we would have obtained without filtering for semantic similarity (for clarity only top 50 is shown):

```
> sorted_stable_genes[1:50]
 [1] "RPL37A"  "EEF1A1"  "RPL7"    "RPL41"   "RPL37"   "RPL9"    "RPS16"
 [8] "RPS23"   "RPS11"   "HUWE1"   "RPS2"    "RPL27"   "TPT1"    "RPL27A"
[15] "RPS3A"   "HNRNPA1" "RPL38"   "RPLP0"   "RPLP1"   "ATP13A5" "RPL34"
[22] "RPS27"   "DCAF6"   "RPS13"   "RPS18"   "RPS15A"  "RPL13"   "RPS28"
[29] "RPL3"    "EIF1"    "RPL23A"  "RPL39"   "RPL30"   "ACTG1"   "RPL10"
[36] "RPS12"   "RPL6"    "RPS20"   "EEF2"    "RPS15"   "EEF1G"   "RPL24"
[43] "ACTB"    "RPL21"   "RPL19"   "OAZ1"    "RPS24"   "HNRNPK"  "RPS10"
[50] "RPL4"
```

This list is very similar to the lists obtained in [de Jonge et al. (2007)] and [Popovici et al. (2009)]. Like those lists they are mainly dominated by ribosomal proteins, which makes sense from a biological point of view since ribosomal proteins are a major component of basic physiologic processes in all cells and a primary target of changing conditions [Popovici et al. (2009)].

We do believe however that the semantic filtering is a necessary step to obtain reference genes that are useful in applications that can benefit from the use of control genes. In order to motivate this we have to recall the main objective, i.e. finding genes that are stable under every possible biological condition. Strictly speaking it can be assumed that there simply exists no gene that is *universally* stable in its expression in all biological situations [Lee et al. (2002),Cheng et al. (2011)]. What we actually try to do however is finding genes that are stable in *most* situations. By selecting genes with diverse functionality we further decrease the chance that under a very specific condition, maybe not covered in our training cohort of 365 studies, all or most of our proposed genes are not stable.

To illustrate this reasoning we would like to point out that even for the ribosomal proteins there exists situations where their expression is not constant, as was demonstrated in a recent study [Thorrez et al. (2008)]. Based on these observations the authors concluded that ribosomal proteins could not be considered as true reference genes. In fact [de Jonge et al. (2007)] already made a similar statement: *Interestingly, the identified candidate novel housekeeping genes do not vary much in terms of functionality; they are predominantly ribosomal proteins involved in protein biosynthesis. Therefore, experimenters that tinker with this specific cellular process would better use other candidate housekeeping genes of our analysis, for example OAZ1.* Their final list of 15 stable genes consists of 13 genes coding for ribosomal proteins and thus for situations under which these genes are not stable only 13% of the reference genes can be used reliably.

With our approach of selecting the most diverse genes in terms of their functionality, we minimize the chance that under specific conditions several or most of the proposed genes are varying. In this perspective, we believe this set actually can be seen as a universal set of reference genes.

Our results are very comparable to the analysis conducted in [Cheng et al. (2011)] on 13 different organ/tissue types. Below their list of 9 genes that expressed a stable behavior in at least eight different organ/tissues:

```
> genes_cheng
 [1] "HUWE1"  "RPL37A" "TPT1" "RPL41" "EEF1A1" "LRRC40" "RPS20"
 [8] "NACAP1" "RPL23A"
```

Genes RPL37A, TPT1 and EEF1A1 are shared between both gene lists. Genes HUWE1, RPL41, RPS20 and RPL23A also appear in our top 100 gene list but have a high semantic similarity with higher ranked genes and where thus discarded. Genes LRRC40 and NACAP1 were not identified by our approach as being stable.

6.3.4 Conclusion

In this integrative analysis of gene expression data for the identification of stable genes through meta-analysis we obtain comparable results as was found in similar, but smaller studies. We implemented however an innovative new step in which genes were filtered based on their semantic similarity, thereby overcoming some of the issues previous attempts had. This results in a compact but diverse set of 12 genes which we believe is a good reference set for normalization purposes.

7 Merging

In this chapter we will go further into detail on the merging approach for conducting integrative analysis of gene expression microarray studies. Compared to meta-analysis, the merging approach is rather recently recognized as a potential solution for large-scale analysis and many of its aspects are not yet well understood. Attention and interest for this type of methods is however growing and we hope with the work presented in this chapter to contribute to the current needs.

In the first section we briefly introduce the merging approach with two concrete examples. In the next section we discuss batch effects, a central problem when merging gene expression microarray studies from different sources. In Section 7.3 an extensive overview of methods to remove batch effects are presented, for the first time in a unified terminology in order to spot their similarities and differences, as well as their strengths and limitations. In a next section we emphasis the current state-of-the-art validation framework for batch effect removal. Finally, the `inSilicoMerging` R/Bioconductor package [Taminau et al. (Subm)] is introduced in Section 7.4, which bundles many of the batch effect and validation methods which were earlier described.

7.1 Introduction

The merging of different gene expression data sets into a global one to conduct large-scale analysis is only recognized recently as a viable approach. Meta-analysis on the other hand was adopted much quicker. This is mainly because of the higher sensitivity of the merging approach to incompatibility issues of the data and its higher computational demands in terms of memory.

In a recent study a merged gene expression matrix of more than 5000 samples from 206 different studies was created [Lukk et al. (2010)] and visualized via Principal Component Analysis (PCA), see Figure 7.1.

This global map of human expression from a large microarray data set looks very impressive and a similar exercise on a much larger cohort of studies from the InSilico DB gave similar results[1]. This cohort contains more than 28000 samples across 985 studies and the creation of a Multidimensional Scaling (MDS) plot took more than a week of computation time (including more than 12 GB of memory), clearly approaching the boundaries of what is possible with standard modern computers.

From the study in [Lukk et al. (2010)] we can have the impression that the problem of integrating data is solved and that all information needed can be found in that joint gene expression space. Although it was not the original intention of the authors, an important question is if we can generalize these results. To find an answer let us first consider the following issues:

[1] Results of this study are not published but an internal report can be found here: `http://como.vub.ac.be/~jtaminau/Report_BigMDS.pdf.zip`

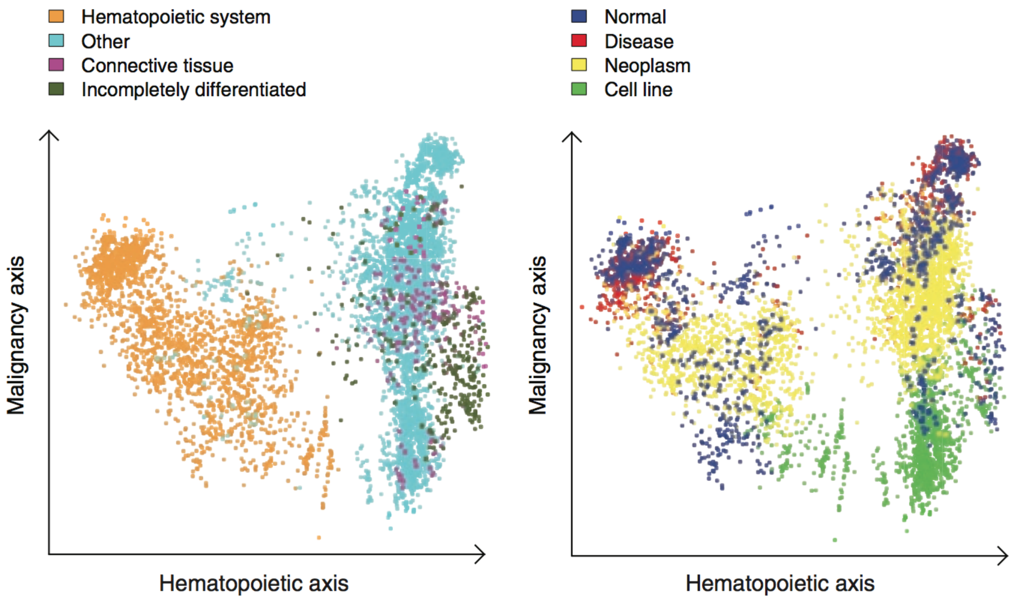


Figure 7.1: Principal Component Analysis (PCA) of the merged data set from [Lukk et al. (2010)]. Each dot represents one of the 5.372 samples and is colored according to the biological group the sample belongs to. On the left it is shown that the first principal axis separates hematopoietic system-derived samples from the rest of the samples. On the right, samples are colored to show that the second principal axis predominantly arranges the malignancy of the samples. Figure taken from [Lukk et al. (2010)].

- First of all we have to note that all samples come from the same platform (Affymetrix Human Genome U133A, GPL96) and are thus *by design* already very compatible with each other. When mixing different platform types this is not the case.

- This work just provides a very global picture of gene expression data and it still remains unclear if it is applicable to discover for example new biomarkers or to reveal new biological insights.

- The difference in the biological groups (e.g. blood samples versus tissue samples or tissues versus cell lines) they selected is known to be significantly large and is clearly bigger than the technical variation of the different studies. But the question is if this is always the case, for example what when we are interested in the difference between different tumor subtypes?

- What those plots don't show is the *study* bias, e.g. how are the samples positioned in this space with respect to the study they belong to. In our earlier work[1] we found out that samples from the same study often group together.

To illustrate some of these considerations we would like to refer to a study we performed and where we studied the merging of different microarray data sets in three different cases with increasing biological complexity [Taminau et al. (2009)c].

In the first case two studies, each containing 60 cell lines (NCI60) and assayed on different platforms, were merged the same way as in [Lukk et al. (2010)] and by using several appropriate normalization methods. A Multidimensional Scaling (MDS) plot visualizing both situations can be found in Figure 7.2. We would expect that samples from the same cell line but from different studies still would cluster together after merging, they are copies from each other after all. This was however only the case when an appropriate normalization technique was used (for visualization purposes we colored the samples based on tumor type instead of cell line).

In a second case four different studies with thyroid tissues were merged together. This time the biological variable of interest was the disease status, being either normal or tumor. Similarly, if no proper transformation was applied to make the different studies compatible which each other the specific biological variation was lost due to the higher technical variation present between the different studies. Note that two studies, in Figure 7.3 represented by the circle (○) and triangle (△) symbols, were even performed by the same lab. Although they are closer grouped together than compared to the other studies a significant study bias still can be observed.

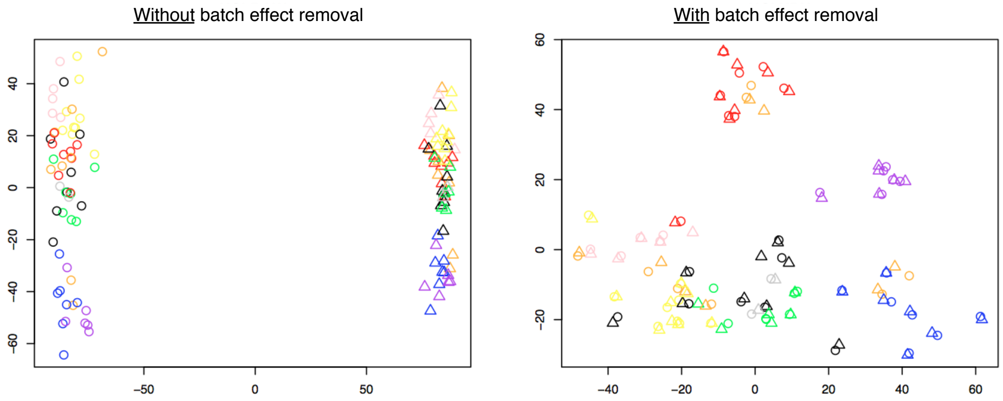


Figure 7.2: Multidimensional Scaling (MDS) plots for the NCI60 case described in [Taminau et al. (2009)c]. Each dot represents a sample and is colored by tumor type, the different symbols correspond to the study the sample belongs to. On the left we have the situation before batch effect removal and on the right after proper batch effect removal. Figure taken from [Taminau et al. (2009)c].

From these examples it is clear that the use of different experimentation plans, platforms and methodologies by different research groups introduces data incompatibilities in the gene expression values, severely hindering downstream analysis. The problems raised by the batch specific unwanted variation as well as the potential sources leading to batch effects have already been revealed and widely discussed in a number of publications [Scherer (2009), Leek et al. (2010)]. These data incompatibilities between studies will further be referred to as batch effects, and the normalization or transformation process to remove them as batch effect removal methods.

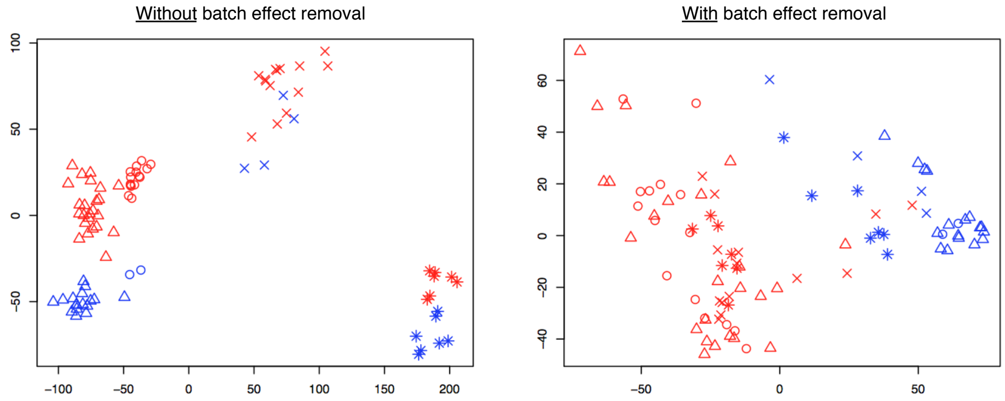


Figure 7.3: Multidimensional Scaling (MDS) plots for the Thyroid case described in [Taminau et al. (2009)c]. Each dot represents a sample with the different symbols corresponding to the studies the samples belong to. Normal samples are colored blue, thyroid cancer samples red. On the left we have the situation before batch effect removal and on the right after proper batch effect removal. Figure taken from [Taminau et al. (2009)c].

7.2 Batch Effects

In this section the concept of *batch effects* will be clarified by analyzing several definitions found in literature. Also a brief summarization of the possible sources leading to batch effects will follow.

7.2.1 Definitions

Providing a complete and unambiguous definition of the so-called batch effect is a challenging task, especially because its origins and the way it manifests in the data are not completely known or not recorded. This is the reason why here we enumerate several definitions as found in the literature. According to these definitions, the batch effect can be defined as one of the following:

Definition 1: - *the uncontrollable errors unrelated to the biological variation* [Gagnon-Bartsch & Speed (2011)].

Definition 2: - *the cumulative errors introduced by time and place-dependent experimental variations* [Chen et al. (2011)].

Definition 3: - *sub-groups of measurements that have qualitatively different behavior across conditions and are unrelated to the biological or scientific variables in a study* [Leek et al. (2010)].

Definition 4: - *systematic differences between the measurements of different batches of experiments* [Luo et al. (2010)].

Definition 5: - *systematic technical differences when samples are processed and measured in different batches* [Scherer (2009)].

From all these definitions two main complementary characteristics of the batch effects can be identified: a first one which makes the distinction between the batch effects and the biological information (Definitions 1 and 3), and a second one which generically reveals the sources of batch effects (Definitions 2, 4 and 5). We provide a more general definition of the batch effects by combining the two main ideas that derive from the definitions mentioned above, as follows:

Definition 6: - *the batch effect represents the systematic technical differences when samples are processed and measured in different batches and which are unrelated to any biological variation recorded during the MAGE experiment* [Lazar et al. (2012)a].

Here, the term *batch* denotes a collection of microarrays (or samples) processed at the same site over a short period of time using the same platform and under approximatively identical conditions, as mentioned in [Chen et al. (2011)].

It is important to highlight the distinction between the terms *noise*, *bias* and *batch effect*. Noise can be seen as the effects of technical components which are not part of the system under investigation but which, if they enter the system, lead to variability in the experimental outcomes. The main difference between noise and batch effect is the systematic nature of the latest. The term bias has a wider meaning which includes not only technical but also other confounding factors[2]. It is defined as unintentional, systematic association of some characteristic with a group in a way that distorts a comparison with another group.

7.2.2 Potential Sources

In order to develop adequate batch removal techniques it is important to understand the nature and the behavior of batch effects. Unfortunately this is not straightforward because at each step of a microarray experiment, a number of potential sources are susceptible to generate batch effects. There are several works in which the authors focused their efforts in identifying and explaining the potential sources of batch effects [Luo et al. (2010), Scherer (2009), Suárez-Fariñas et al. (2005)]. Here we will only list the potential sources of batch effects and the stage where they appear.

As a general accepted rule, microarray experiments can be summarized in five stages: growing the organism, tissue sampling, RNA processing, hybridization and data extraction, and different sources of batch effects can affect the outcome of the experiments, as illustrated in Figure 7.4. Every step that requires manual intervention can lead to batch-specific variance since every lab, and even every person, has its own methodology or procedure (in Figure 7.4 it can indeed be observed that Personnel Effects affect every step) . This can be improved by following strict and well-defined protocols for each step since less batch effect can be observed between studies performed by the same lab than studies performed by

[2] Confounding factors (also known as distorting factors), represent variables or factors that distort the observed association between the biological variation of interest and the conducted study.

different labs. However, many other factors which are more difficult to control (e.g. environmental conditions, location, date, etc.) are still having a combined influence, making batch effects very hard to avoid systematically.

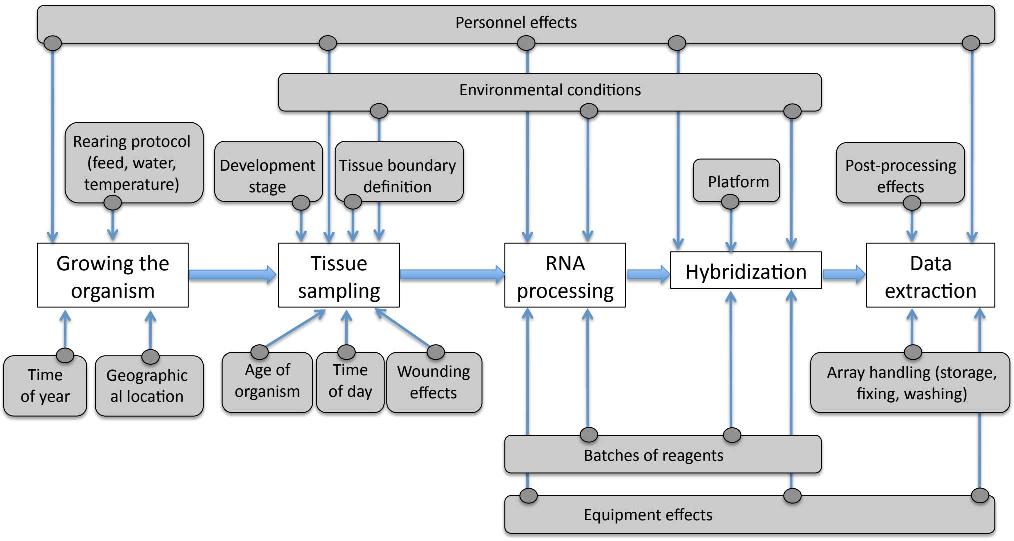


Figure 7.4: A visualization of potential batch effect sources at each stage of a microarray analysis experiment. The grey boxes represent the potential sources of batch effects affecting the different steps in a typical microarray experiment, illustrated by the white boxes. Figure taken from [Lazar et al. (2012)a].

For more details and complete explanations on the above mentioned batch effects the reader is invited to consult Chapter 4 in [Scherer (2009)].

7.3 Merging Microarray Gene Expression Data by Removing Batch Effects

Merging and batch effect removal of gene expression data is often used in the same context. In the reminder of this thesis we will refer to *merging* as the process to physically combine two or more data sets into a new larger data set and to *batch effect removal* as the transformation on the data in order to make the separate data sets more compatible. Merging without any removal of batch effects is thus possible and is nothing more than a sample-wise concatenation of the different single data sets.

A number of batch effect removal methods exist and are reasonably documented (and validated) in their original publications. Due to differences in notations it is however very hard to obtain a general overview of the batch effect removal methods to spot their weaknesses, strengths or similarities. In this section we will provide an extensive overview of the different methods using a unified framework. To avoid any confusion we start by elucidating our terminology, followed by a detailed description of each method.

To our knowledge, no other survey of batch effect removal methods using a unified terminology exists before and most of this work was published in *Briefings in Bioinformatics* [Lazar et al. (2012)a].

7.3.1 Terminology and General Assumptions

In Section 2.3.2 we already denoted a preprocessed microarray data set by $X^{m \times n}$, where each column represents a sample and each row represents a feature. x_{ij} represents the expression value of feature i in sample j. All samples from this data set belong to the same batch X. Other batches are represented similarly, for example $Y^{m' \times n'}$ for batch Y. In Table 7.1 an overview of all notations used to describe the different batch effect removal methods is given.

We have to note that in the context of merging of microarray data sets the features are always genes instead of probes. As we recall, probes are platform-dependent features which makes the merging of data sets from different microarray platforms impossible. Different probe-to-gene mapping procedures exist and it is important that this mapping is consistently performed over all single data sets before merging.

$Notation$	$Explanation$
$X^{m \times n}$, $Y^{m' \times n'}$	microarray data set (batch) with m (m') genes and n (n') samples
$\hat{X}^{m \times n}$, $\hat{Y}^{m' \times n'}$	microarray data set (batch) with m (m') genes and n (n') samples after batch effect removal
x_{ij}, y_{ij}	expression of gene i in sample j in corresponding batch
$\hat{x}_{ij}$, $\hat{y}_{ij}$	expression of gene i in sample j in corresponding batch after batch effect removal
$\overline{x}_i$, $\overline{y}_i$	mean expression of gene i in corresponding batch
σ_{x_i}, σ_{y_i}	standard deviation expression of gene i in corresponding batch
x^r_{ij}, y^r_{ij}	expression of gene i in j'th reference sample in corresponding batch
b^X_{ij}, b^Y_{ij}	bias for gene i in sample j of corresponding batch
ϵ^X_{ij}, ϵ^Y_{ij}	noise in gene i of sample j of corresponding batch
γ^X_i, γ^Y_i	additive gene i specific bias for corresponding batch
δ^X_i, δ^Y_i	multiplicative gene i specific bias for corresponding batch

Table 7.1: Overview of consistent and unified notations used in the remainder of this chapter to describe the different batch removal methods.

When combining the expression values from multiple data sets, it is assumed that all data sets have the same distribution of samples for each biological variable of interest. It is for example impossible to remove batch effects between two studies for which one study only contains control samples and another only diseased samples, when the disease status is the biological variable of interest.

In general, it is assumed that the batch effect comes in either multiplicative or additive form, or a combination of both. Since preprocessed gene expression data is always log transformed, these effects are both represented as additive terms. All batch effect removal methods assume that the *measured* expression values of gene i in sample j of batch X can be expressed in a general form as follows:

$$x_{ij} = x'_{ij} + b^X_{ij} + \epsilon^X_{ij} \tag{7.1}$$

with x'_{ij} the *actual* gene expression, b^X_{ij} the batch effect term and ϵ^X_{ij} a noise term. Clearly, x'_{ij} is the value of interest as this represents the actual abundance of mRNA of that gene in a particular sample. Different batch effect identification and removal methods refine this general description by splitting b^X_{ij} in different terms, or by adding terms that are specific for known covariates[3]. Within this general description, the term b^X_{ij} can indicate the batch effect related to any of the sources mentioned in 7.2.2.

7.3.2 Overview of Batch Effect Removal Methods

There are two main approaches for removing batch effects: location-scale (LS) methods and matrix-factorization (MF) methods. LS methods assume a model for the location (mean) and/or scale (variance) of the data within the batches and proceeds to adjust the batches in order to agree with these models. MF techniques assume that the variation in the data corresponding to batch effects is independent on the variation corresponding to the target biological variable of interest and can be captured in a small set of factors which can be estimated via matrix factorization methods. The strategy adopted by these methods is to identify and remove the influence of these factors. A smaller group of valuable methods for batch effect removal is based on data discretization. In Figure 7.5 a basic taxonomy of the batch effect removal methods that will be described in this section is provided. For completeness, methods not belonging to the three approaches described above are briefly mentioned as well.

Location-Scale Methods

The main idea behind location-scale (LS) methods is to transform the data from each batch to have similar (equal) mean and/or variance for each

[3] A covariate is a variable that is possibly predictive of the outcome under study, such as the condition of the experiments or biological information such as male/female.

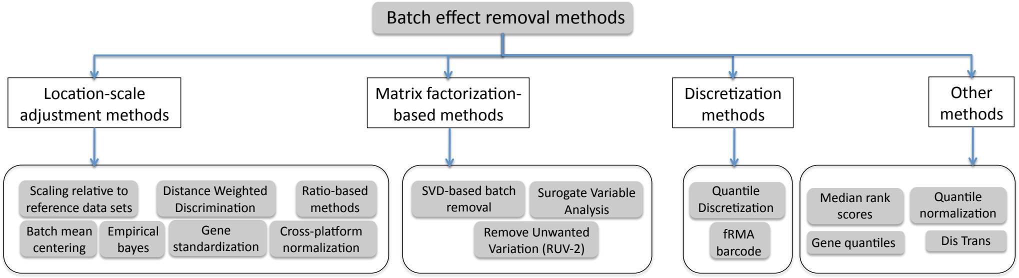

Batch effect removal methods
Location-scale adjustment methods
Matrix factorization-based methods
Discretization methods
Other methods
Scaling relative to reference data sets
Distance Weighted Discrimination
Ratio-based methods
Batch mean centering
Empirical bayes
Gene standardization
Cross-platform normalization
SVD-based batch removal
Surogate Variable Analysis
Remove Unwanted Variation (RUV-2)
Quantile Discretization
fRMA barcode
Median rank scores
Quantile normalization
Gene quantiles
Dis Trans

Figure 7.5: A basic taxonomy of the batch effect removal methods for gene expression data described in this chapter. Figure taken from [Lazar et al. (2012)a].

gene. It is assumed that these transformations, while trivially making data more comparable, do not remove any biological signal of interest.

Batch mean-centering

Assuming the prevalence of multiplicative systematic batch effects, batch mean-centering (BMC) was introduced in [Sims et al. (2008)]. This simple method transforms the data by subtracting the mean of each gene over all samples (per batch) from its observed expression value, such that the mean for each gene becomes zero. BMC assumes that in the general expression in Equation 7.1, b^X_{ij} represents the multiplicative gene specific batch effect.

$$\hat{x}_{ij} \quad = \quad x_{ij} - \overline{x}_i \tag{7.2}$$

Gene standardization

Gene-wise standardization transforms all genes to have zero mean and standard deviation one by subtracting the mean and dividing by the standard deviation of each gene over all samples within a batch. A Z-score standardization is used for this purpose. Similar to BMC, it is assumed that in the general expression in Equation 7.1, the b^X_{ij} represents the multiplicative gene specific batch effect. The pure (batch effect free) gene expression values are obtained as follows:

$$\hat{x}_{ij} = \frac{x_{ij} - \overline{x}_i}{\sigma_{x_i}} \tag{7.3}$$

Ratio based methods

Ratio based methods [Li & Wong (2001)] scale the expression value of each gene in each sample based on a set of reference samples in each batch. If there is more than one reference sample, the arithmetic or geometric mean value of the expression values in the reference samples can be used. It is also possible to use a universal set of reference samples [Novoradovskaya et al. (2004)]. We denote by x_{il}^r the value of the ith gene in the lth reference sample in batch X. It is assumed that the genes in the reference samples are subjected to the same batch effect as in the rest of the samples and therefore the term b_{ij}^X in Equation 7.1 will be removed by subtracting the mean of each gene of the reference samples of the corresponding batch. Assuming k reference samples for batch X, the following two methods are proposed:

Arithmetic mean ratio-based method (Ratio-A):

$$\hat{x}_{ij} = x_{ij} - \frac{1}{k}\sum_{l=1}^{k} x_{il}^r \tag{7.4}$$

Geometric mean ratio-based method (Ratio-G):

$$\hat{x}_{ij} = x_{ij} - \sqrt[k]{\prod_{l=1}^{k} x_{il}^r} \tag{7.5}$$

The geometric mean has the benefit that it is less sensitive to outliers. Instead of using the mean also the median could be used.

Scaling relative to reference data set

In [Kim et al. (2007)] the authors propose to change the distribution of a gene based on the distribution of that same gene in a reference data set. Samples are grouped by their biological variable of interest. Assume without loss of generality that $X^{m \times n}$ is the data to be adjusted and $Y^{m' \times n'}$ the reference data. Furthermore, assuming that the samples of each data set are divided into k categories such that x_{ij}^c is the expression value of gene i in the jth sample belonging to category c in batch X, the batch effect adjusted data are derived as follows:

$$\begin{aligned} \hat{x}_{ij}^c &= x_{ij}^c \frac{\sigma_{y_i^c}}{\sigma_{x_i^c}} - \left(\overline{x_i^c} \frac{\sigma_{y_i^c}}{\sigma_{x_i^c}} - \overline{y_i^c} \right) && (7.6) \\ \hat{y}_{ij}^c &= y_{ij}^c && (7.7) \end{aligned}$$

with c representing the category, while $\overline{x_i^c}$ ($\overline{y_i^c}$) and $\sigma_{x_i^c}$ ($\sigma_{y_i^c}$) are the means and the standard deviations of gene i in all X (Y) samples belonging to category c respectively.

COMBAT method

COMBAT [Johnson et al. (2007)], also known as Extended Johnson-Li-Rabinovich (EJLR) or Empirical Bayes (EB) method, is a method using estimations for the LS parameters (mean and variance) for each gene. The parameters are estimated by pooling information from multiple genes with similar expression characteristics in each batch. There exist both a parametric and nonparametric approach and we give a concise explanation. Details can be found in the original publication.

It is assumed that measured gene expression values of gene i in sample j in each batch can be expressed as a specialization of Equation 7.1 as:

$$x_{ij} = \alpha_i + \mathbf{C}\beta_i + \gamma_i^X + \delta_i^X \epsilon_{ij}^X \qquad (7.8)$$

where α_i is the gene expression not related to any known covariates, $\mathbf{C}$ is a design matrix for sample conditions (known covariates), β_i is the vector of regression coefficients corresponding to $\mathbf{C}$, γ_i^X and δ_i^X are the additive and multiplicative batch effects for gene i respectively and ϵ_{ij}^X are noise terms. ϵ_{ij}^X are assumed to follow a normal distribution with mean zero and variance σ_i^2. The first step in COMBAT is to standardize the data using estimates $\tilde{\alpha}_i$, $\tilde{\beta}_i$, $\tilde{\delta}_i^X$ and $\tilde{\sigma}_i^2$ for the corresponding variables. The standardized gene expression z_{ij} is assumed to be normally distributed according to $N(\gamma_i^X, (\delta_i^X)^2)$ and is given by:

$$z_{ij} = \frac{x_{ij} - \tilde{\alpha}_i - \mathbf{C}\tilde{\beta}_i}{\tilde{\sigma_i^X}} \tag{7.9}$$

The batch effect adjusted data are given by:

$$\hat{x}_{ij} \quad = \quad \frac{\tilde{\sigma}_i}{\hat{\delta}_i^{X*}}(z_{ij} - \hat{\gamma}_i^{X*}) + \tilde{\alpha}_i + \mathbf{C}\tilde{\beta}_i \tag{7.10}$$

where $\hat{\gamma}_i^{X*}$ and $\hat{\delta}_i^{X*}$ are estimates of batch effect parameters in Equation 7.8 estimated using parametric or nonparametric empirical priors. In case of parametric priors it is assumed that $\gamma_i^X \sim N(\gamma^X, (\tau^X)^2)$ and $(\delta_i^X)^2 \sim InverseGamma(\lambda^X, \theta^X)$, where $\gamma^X, (\tau^X)^2, \lambda^X$ and θ^X are estimated empirically.

Cross-platform normalization

The basic idea behind the cross-platform normalization approach (XPN [Shabalin et al. (2008)]) is to identify homogeneous blocks (clusters) of gene and samples in both data sets that have similar expression characteristics. In XPN, a gene measurement within one such block can be considered as a scaled and shifted block mean, where both scaling and shifting are dependent on the gene i and sample j. The recorded gene expression is then expressed as a specialization of Equation 7.1 by:

$$x_{ij} = A^X_{\alpha^*(i),\beta^*(j)} b_i^X + c_i^X + \sigma_i^X \epsilon_{ij}^X \tag{7.11}$$

where $A^X_{\alpha^*(i),\beta^*(j)}$ is a block mean and b_i^X and c_i^X represent gene and platform specific sensitivity and offset parameters respectively. The functions

$\alpha^*()$ and $\beta^*()$ map a specific gene measurement in a sample to their corresponding multi-platform cluster. The noise variables ϵ^X_{ij} are assumed to be independent and normally distributed. Using maximum likelihood methods estimates for the parameters in Equation 7.11 ($\tilde{A}^X_{ij}, \tilde{b}^X_i, \tilde{c}^X_i$ and $\tilde{\sigma}^X_i$) are obtained for each batch. Common model parameters ($\hat{A}_{ij}, \hat{b}_i, \hat{c}_i$ and $\hat{\sigma}_i$) were calculated as weighted averages of these batch specific estimates. Subsequently, the batch effect adjusted data is given by:

$$\hat{x}_{ij} \quad = \quad \hat{A}_{\alpha^*(i),\beta^*(j)}\hat{b}_i + \hat{c}_i + \hat{\sigma}_i \left(\frac{x_{ij} - \tilde{A}^X_{\alpha^*(i),\beta^*(j)}\tilde{b}^X_{x_i} - \tilde{c}^X_i}{\tilde{\sigma}^X_i} \right) \qquad (7.12)$$

Distance-weighted discrimination

Distance-weighted discrimination (DWD [Benito et al. (2004)]), an adaptation of the Support Vector Machines (SVM [Cristianini & Shawe-Taylor (2000)]) principle, can be used for batch effect removal as follows. As a starting point, samples from a single batch are regarded as belonging to a specific class and DWD is used as a classification algorithm by finding the optimal hyperplane $w \times x + b = 0$ separating samples from the different classes (batches), with w the normal vector of the hyperplane. Next the samples in each batch are projected in the direction of the normal vector to this hyperplane by calculating the mean distance from all samples in each batch to the hyperplane ($\overline{d^X}$) and then subtracting the normal vector to this plane multiplied by the corresponding mean distance.

$$\hat{x}_{ij} \quad = \quad x_{ij} - \overline{d^X} w_i \qquad (7.13)$$

Matrix factorization-based methods

The idea behind these methods resides in the observation that the most important source of differentially expression is nearly always across batches rather than across biological groups [Leek et al. (2010)]. Based on this observation, these methods rely on the following strategy:

1. Perform matrix factorization of the input data matrix (which is in general obtained by sample-wise concatenating of the data sets to be combined); the matrix factorization is usually performed using either Singular Value Decomposition (SVD [Alter et al. (2000)]) or Principal Components Analysis (PCA [Jolliffe (2002)]), such that the first factor has the highest possible variance (which is associated with batch effects).

2. Remove the factors associated with batch effects and reconstruct back the batch effect adjusted data set.

In the discussion below we assume that we wish to combine data from two batches $X^{m \times n}$ and $Y^{m' \times n'}$, and denote by $C^{m'' \times n''} = [X^{m'' \times n} \; Y^{m'' \times n'}]$ the sample wise concatenation over common genes of the studies, with m'' the number common genes between X and Y and $n'' = n + n'$.

Singular Value Decomposition based batch effect removal

Singular Value Decomposition can be used to adjust for batch effects by factorizing the input gene expression data matrix and then reconstructing it while filtering out those factors that are associated with the batch effect. In a first instance, the matrix $C^{m'' \times n''}$ is factorized using SVD as follows:

$$C^{m'' \times n''} = U^{m'' \times n''} \Sigma^{n'' \times n''} (V^{n'' \times n''})^T \tag{7.14}$$

where $C^{m'' \times n''} = [X^{m'' \times n} \quad Y^{m'' \times n'}]$, m'' is the number of common genes between X and Y, $n'' = n + n'$ is the total number of samples in X and Y, while the columns of $U^{m'' \times n''}$ and the rows of $(V^{n'' \times n''})^T$ form orthonormal basis for the samples (eigensamples)/ genes (eigengenes) respectively. The matrix $\Sigma^{n'' \times n''}$ is a diagonal matrix containing the singular values $(s_1 \geq \ldots \geq s_{n''} \geq 0)$. The reconstruction of the data, with the batch effect removed can be done by removing those components in the corresponding matrices that are believed to map to the batch effect:

$$\hat{C}^{m'' \times n''} = U^{m'' \times l} \Sigma^{l \times l} (V^{n'' \times l})^T \tag{7.15}$$

with $l \leq n''$ and $U^{m'' \times l}$, $\Sigma^{l \times l}$ and $(V^{n'' \times l})^T$ representing the same matrices with the rows (columns) corresponding to the components mapping to the batch effect removed. As an alternative matrix factorization method, PCA can be also employed.

Surrogate Variable Analysis (SVA)

In [Leek & Storey (2007)] the assumption made is that it is possible to identify the signal in $C^{m'' \times n''}$ due to the biological variance of interest and obtain the residuals $R^{m'' \times n''}$ after the removal of this signal. The variation in these residuals is then assumed to be unwanted variation caused by batch effects. In order to remove this unwanted variation, a matrix factorization is then applied on the residuals.

The main variation in the residuals is used as factors to be adjusted for in downstream analysis. This is done by estimating surrogate variables representing the unknown confounding effects by iteratively weighting a subset of the factors identified in the decomposition. For details the reader is referred to [Leek & Storey (2007)].

Remove Unwanted Variation, 2-step (RUV-2)

In [Gagnon-Bartsch & Speed (2011)] a similar method for batch effect removal is proposed which makes use of a set of control genes to identify the factors associated with the batch effect. It is assumed that the control genes are *a priori* known to be uncorrelated with the biological factor of interest. Assume that there are p control genes, then the RUV-2 method proposes to apply a matrix factorization on these genes to identify the components corresponding to the batch effects. Thus, instead of performing SVD on $C^{m'' \times n''}$ it is done on a submatrix $C_c^{p \times n''}$, where the subscript c indicates that only the p control genes are considered as input in this step. Similarly $U_c^{p \times l}$ and $U_c^{p \times n}$ are the submatrices concerning the control genes of the corresponding matrices in Equation 7.14.

Based on visual inspection or some variation criteria the first l components $U_c^{p \times l}$ of the eigensamples $U_c^{p \times n}$ are deemed relevant, and are then added as covariates to any type of downstream analysis. When this information is passed on to COMBAT this can be used to adjust the data for batch effects; another option is to reconstruct the data using the obtained decomposition by removing the first l components.

The last two methods (SVA and RUV-2) identify factors associated with batch effects but do not straightforward return an adjusted *merged* data matrix. These methods can however be used for batch effect removal in two ways: (a) combining them with another batch effect removal method (for instance COMBAT) including the identified batch effects as covariates, (b) reconstructing the data after removing factors identified as being associated with batch effects.

Discretization Methods

Discretization methods aim to transform the expression values into consistently defined categories, or *bins*, based on their level of expression. Quantile Discretization for example is a discretization method based on equal frequency binning by using the quantiles as cut points for the bins [Warnat et al. (2005)]. Based on fRMA (see Section 2.3.2, [McCall et al. (2010)]), a novel algorithm for generating *barcodes* was introduced in [McCall et al. (2011)]. The barcode representation of a sample is a vector of ones and zeros denoting which genes are estimated to be expressed and unexpressed, respectively. These estimates are based on huge set of samples which were collected and consistently normalized using fRMA (per platform).

After discretization, a loss of information is inevitable but it has been shown that these methods can sometimes even lead to similar or improved accuracy depending on the type of downstream analysis [McCall & Irizarry (2011)].

Other Methods

Other less popular techniques for batch effect removal not fitting in the above three categories also exist.

Quantile Normalization (QN) is more frequently used for normalization at the probe level, for example in `RMA` preprocessing (see Section 2.3.2, [Irizarry et al. (2003)]), but has been also used explicitly for batch effect removal [Bolstad et al. (2003), Lacson et al. (2010)]. two similar ideas to QN are Median Rank Scores (MRS, [Warnat et al. (2005)]) and gene Quantiles (GQ, [Xia et al. (2009)]).

In [Jiang et al. (2004)], Distribution Transformation (DisTran) was proposed, where a reference sample is constructed based on a combination of the mean expression of samples having the same biological value of interest. All other samples are then transformed to have the same distribution as this constructed reference sample.

7.3.3 Validation of Merged Gene Expression Data Sets

Evaluating and validating the batch effect removal methods is important but at the same difficult, due to the unclear objective of the batch removal process. It is clear that after batch removal, two data sets from different sources should be *comparable* or *compatible*, but researchers can use their freedom in choosing the specific criteria to evaluate *how* two data sets are comparable. This leads to a wide plethora of different validation techniques found in literature, as can be seen from Figure 7.6. In this section we will describe the most important ones, based on a simple taxonomy (see Figure 7.6).

We also implemented all described validation methods in the `inSilicoMerging` package, which was accepted to be part of the R/Bioconductor [Gentleman et al. (2004)] repository and thus public available. A complete description of the package can be found in Section 7.4.

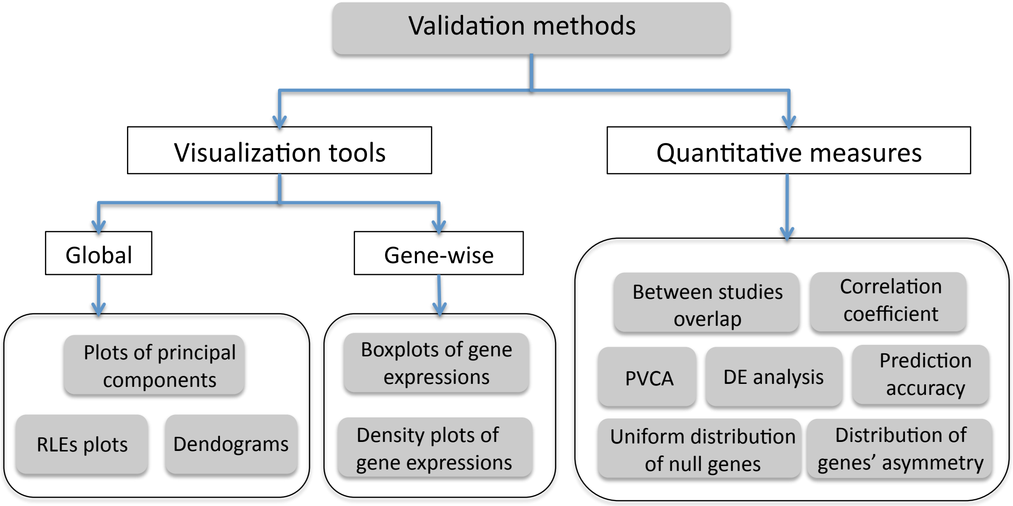


Figure 7.6: A basic taxonomy of the validation tools to evaluate the merging process of gene expression data described in this chapter. Figure taken from [Lazar et al. (2012)a].

Validation tools for batch effect removal can be mainly divided in two groups, qualitative or visualization tools and quantitative measures. In general, the visualization tools provide a crude approximation of the efficiency of the batch effect removal method and can be used to provide a first and rapid inspection of the results. It immediately shows if the merging did made sense and if the global batch effect is still present or not. For more rigorous evaluation the more accurate quantitative measures should be used.

Visualization Tools

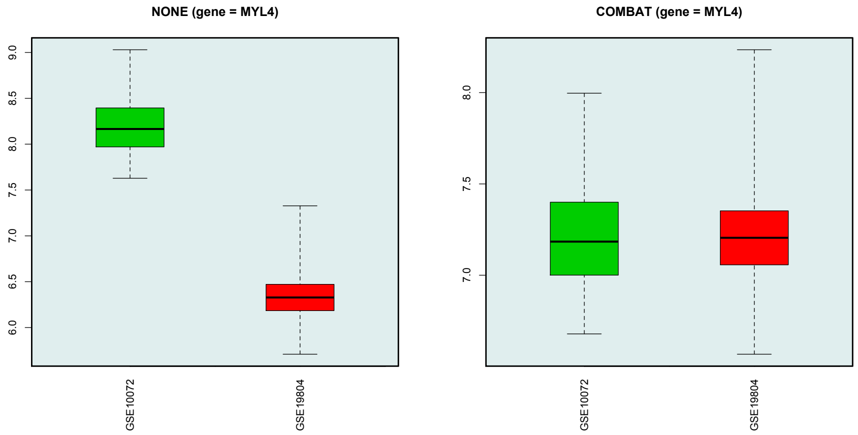
The most common and straightforward way to evaluate the effectiveness of batch effect removal methods is by visualization means. We divided the visualization tools in two groups: gene-wise and global tools. Both categories work on a different level. As the name suggests, gene-wise tools provide a local visualization of the batch effect at the gene level. It is expected that the gene expression levels of the same gene across different studies have similar distributions if no batch effect is present[4]. As will be demonstrated later, the batch effect can be different for each gene. The global tools provide a ″big picture″ of the presence of the batch effect. According to these tools, it is expected that the samples corresponding to the same category of the target biological variable of interest will group together, regardless of the study they originate from. The two groups of tools provide complementary information about the batch effect and it is advisable to be jointly used for evaluation.

All visualization tools are illustrated by using the appropriate functions from the `inSilicoMerging` package. Two lung cancer data sets with equal distribution of control and lung cancer samples (`GSE19804` and `GSE10072`) were retrieved using the `inSilicoDb` package, see Section 4.4. All examples illustrate the COMBAT method for merging, compared to merging without taking care for batch removal. Code to generate all plots in this section can be found in Appendix A.6.

Gene-wise boxplots

Gene-wise box plots are used to compare the distributions of genes in different data sets and its use in the context of batch effect removal validation was suggested in [Leek et al. (2010), Kim et al. (2007)]. Boxplots are a graphically summarization of a discrete distribution via five parameters: minimum, maximum and the lower, upper and median quartile. A batch effect removal method is considered to be effective if the box plots are located around the same value. An illustration is provided in Figure 7.7, where the boxplots of an arbitrary selected gene (MYL4) in the two data sets are shown without (NONE) and with (COMBAT) applying batch effect removal.

[4] This is of course under the assumption that two data sets have the same distribution of samples relative to the target biological variable of interest as defined as a general assumption for merging, see Section 7.3.1. Otherwise the different estimates of the statistical parameters will be different even when there is no batch effect affecting the data.



(a) (b)

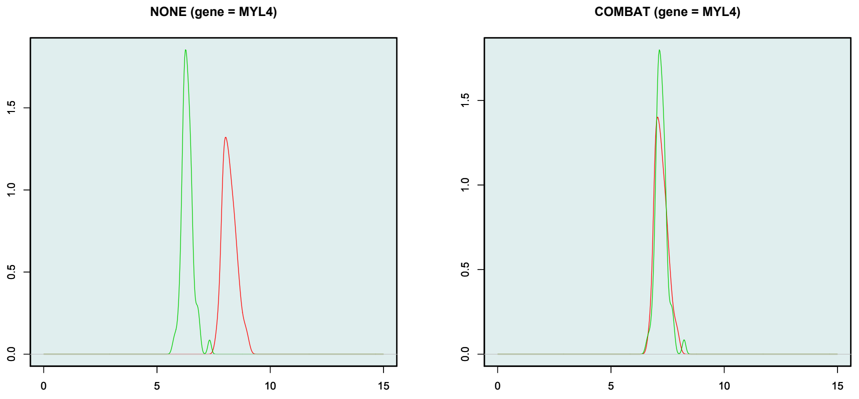
Figure 7.7: Illustration of gene-wise boxplots as validation tool for batch effect removal: a) before and b) after batch effect removal (using COMBAT method).

Gene-wise density plots

Another way to visually inspect the distribution of expression values of a gene is by plotting the *probability density function (pdf)* of that gene. In [Kim et al. (2007)] this was done by plotting the pdfs of several genes randomly selected genes. The densities can be estimated via the Parzen-Rosenblat method [Parzen (1962)]. A batch effect removal method is considered to be effective if the pdfs are fully overlapping. An illustration is provided in Figure 7.8, where the densities of an arbitrary selected gene (MYL4) in the two data sets are shown without (NONE) and with (COMBAT) applying batch effect removal.

Dendrograms

In cluster analysis, a dendrogram is a tree representation of the clustering solution obtained via hierarchical clustering. In general, dendrograms are commonly used to cluster either genes or samples in homogeneous groups. In the context of batch effect removal, the dendrograms



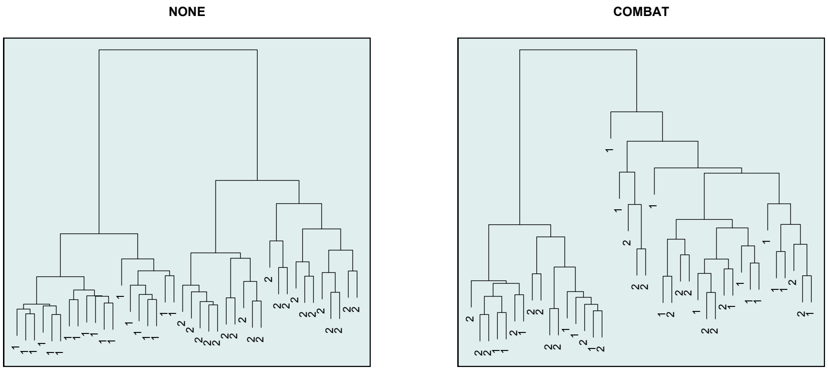
(a) (b)

Figure 7.8: Illustration of gene-wise density plots as validation tool for batch effect removal: a) before and b) after batch effect removal (using COMBAT method).

are mainly used to visualize how well the samples exhibiting the same biological characteristics originating from different studies, cluster together [Kim et al. (2007), Leek et al. (2010)]. Another important interpretation of dendrograms as validation tool for batch effect removal is that if the samples mainly cluster by study, it is a clear indication that batch effects are present. This can be helpful in situations where the annotations corresponding to the biological characteristics of samples might not be available or might not have a strong influence on gene expression. An illustration is provided in Figure 7.9, where the dendrograms after clustering the two data sets are shown without (NONE) and with (COMBAT) applying batch effect removal. For clarity purposes only 40 samples were arbitrary selected.

Multidimensional Scaling (MDS) plots

In MDS plots samples are positioned in a 2-dimensional Euclidean such that their underlying distances are as much like the underlying distances in the probe-sample space. Samples can be colored based on the target bi-



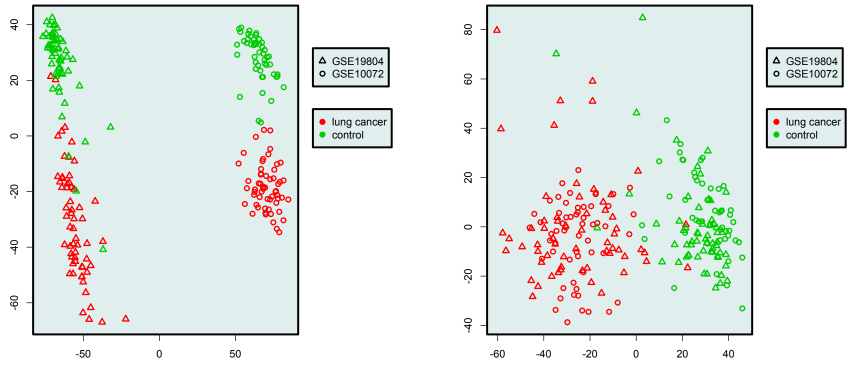
(a) (b)

Figure 7.9: Illustration of dendrogram plots as validation tool for batch effect removal: a) before and b) after batch effect removal (using COMBAT method). For clarity purposes, samples are labeled by a number corresponding to the study they originate from.

ological variable of interest on based on the study they originate from to determine if samples group by study or by the same biological characteristics. The `inSilicoMerging` package provides the option to double-label the samples, by using different colors and symbols. An illustration is provided in Figure 7.10, where the MDS plots after clustering the two data sets are shown without (NONE) and with (COMBAT) applying batch effect removal.

Relative Log Expression (RLE) plots

RLE plots were initially proposed to measure the overall quality of a data set aiming to identify bad chips [Brettschneider et al. (2008)] but were recently also proposed to validate batch effect removal methods [Gagnon-Bartsch & Speed (2011)]. For each gene, the median log expression level is computed over all samples, then for each gene on each sample, the deviation from the median log expression level is computed by:



(a) (b)

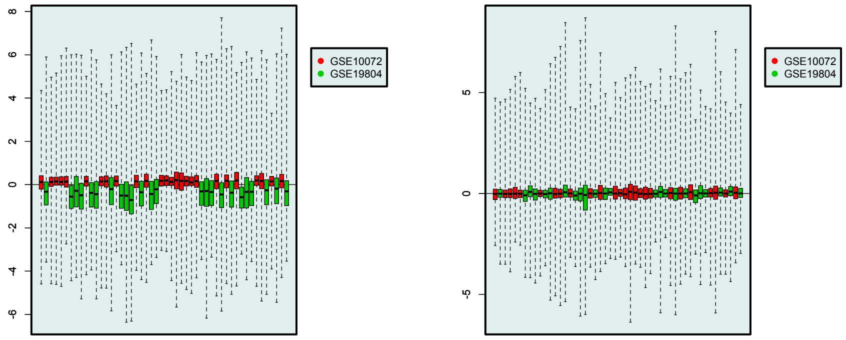
Figure 7.10: Illustration of multidimensional Scaling (MDS) plots as validation tool for batch effect removal: a) before and b) after batch effect removal (using COMBAT method).

$x_{i,j} - median(x_i)$. Finally, an RLE plot is obtained by plotting for each sample a boxplot for all its deviations. For an efficient batch effect removal method, the individual boxplots will all be distributed around 0. An illustration is provided in Figure 7.11, where the dendrograms after clustering the two data sets are shown without (NONE) and with (COMBAT) applying batch effect removal. For clarity purposes only 40 samples were arbitrary selected.

In addition to the five above described visualization tools, two other tools were also reported: correlation heat maps and variance components pie charts [Luo et al. (2010)].

Quantitative Measures

The following measures provide a more objective evaluation of the batch effect removal process by providing a numerical score. Those quantitative measures are very effective for comparing the results of different methods. In Figure 7.6 and in [Lazar et al. (2012)a] the most common



(a) (b)

Figure 7.11: Illustration of relative log expression (RLE) plots as validation tool for batch effect removal: a) before and b) after batch effect removal (using COMBAT method).

measures are described, but in this section we will limit ourselves to the measures implemented in the `inSilicoMerging` package (see Section 7.4).

Measuring the overlap of samples and genes

Measuring the expected overlap between samples of two independent studies before and after applying batch effect removal was proposed as a validation strategy in [Shabalin et al. (2008)]. The overlap is quantified as follows:

1. Compute the distance between each sample in the first study and its nearest neighbor in the second study.
2. Repeat step 1 by changing the roles of the studies
3. Average the results in steps 1 and 2

A method is considered to be effective if it results in a substantial overlap between samples and thus the higher the overlap, the better the integration process.

Following the same idea, also the overlap between genes can be used in this context. Therefore we implemented a novel quantitative validation index which calculates the average difference in the distribution of all genes in the individual studies as follows:

$$GOV_i = \frac{\sum |P_{x_i} - P_{y_i}|}{2} \tag{7.16}$$

where P_{x_i} and P_{y_i} are the normalized *pdfs* (such that $\sum_i P_{x_i} = 1$ and $\sum_i P_{y_i} = 1$) of gene g_i in the first respectively second data set and they are empirically estimated using Parzen-Rosenblatt density estimation method [Parzen (1962)]. Note that this index is bounded in $[0\ 1]$, the minimum value being obtained when the two distributions are identical while the maximum value is reached when the two distributions are completely separated. The global cross-studies genes' overlapping index is given by:

$$GOV = \frac{\sum_i^m GOV_i}{m} \tag{7.17}$$

where m is the number of common genes between the two data sets. Note that GOV is still bounded in $[0\ 1]$, providing a clear quantification of the batch effect removal or of the quality of the data integration process.

Both the overlap scores of samples and genes are implemented in the `inSilicoMerging` package by the `measureGenesOverlap` and `measureSamplesOverlap` functions respectively.

Comparing the distribution of genes' asymmetry

Another simple and efficient way to quantify the results of batch effect removal methods proposed in [Shabalin et al. (2008)], is to compare the distribution of samples' asymmetry before and after batch removal. For a given random variable, a raw approximation of its asymmetry is given by the difference between its mean and median values. However, the *skewness* is another efficient statistical measure that quantifies the asymmetry of a distribution, and it can be used instead. This index is computed as being the area between the *cumulative density functions* (cdfs) of samples' asymmetry, estimated before and after batch effect removal. This index can be defined as follows:

$$bias_{X,\hat{X}} = \sum_{i=a}^{b} (CDF_X(i) - CDF_{\hat{X}}(i)) \tag{7.18}$$

where CDF_X and $CDF_{\hat{X}}$ represent the cdfs of samples' asymmetry before or after batch effect removal, while a and b are the minimum, respectively the maximum values of samples' asymmetry before and after batch effect removal. A method is considered to be efficient if, after batch effect removal, the two cdfs are as similar as possible, and so the index should have a value close to 0.

The asymmetry index is implemented in the `inSilicoMerging` package by the `measureAsymmet ry` function.

Evaluation via differential expression analysis

Several studies propose to evaluate the effectiveness of the batch effect removal methods in the context of the differentially genes expression (DEG) analysis [Gagnon-Bartsch & Speed (2011),Sims et al. (2008),Leek & Storey (2007)]. It is generally assumed that DEG analysis performed on the adjusted data set should result in a more reproducible list of genes which are differentially expressed. The authors in [Gagnon-Bartsch & Speed (2011)] propose a quality metric to measure the effectiveness of a batch effect removal. The metric proposed is proportional to the number of *positive control genes*, i.e. genes that are known *a priori* to be truly differentially expressed, found in the top k ranked genes according to a particular method for differentially expressed genes (DEGs) discovery. A batch effect removal method should be considered as being effective if the number of *positive control genes* found in the adjusted data set increases with respect to those found in the original studies.

If the *positive/negative control genes* are unknown, the authors in [Nueda et al. (2011)] propose an evaluation strategy based on functional enrichment analysis [Al-Shahrour et al. (2007)] which assesses whether specific cellular functions are overrepresented within a set of significant genes.

In [Sims et al. (2008)], the authors propose a different way to use DEG analysis to assess the efficiency of batch effect removal methods, at gene/ probe level. The idea is to first identify lists of the most DEGs in the newly combined data set and to compare those lists with the most DEGs from other single or differently combined data sets. The efficiency of a method is in this case proportional to the number of overlapping probes in the compared lists. This approach has the advantage that no prior or case-specific information is needed.

The index implemented in the `inSilicoMerging` package is based on the approach by [Sims et al. (2008)] by the function `measureSignificantGenesOverlap`. To calculate the DEGs, the R/Bioconductor package `limma` [Smyth (2004)] was selected.

Correlation coefficients

In the context of batch effect removal, the correlation coefficient is used to observe and to quantify how much the batch effect removal methods affect the data [Shabalin et al. (2008), Kim et al. (2007)]. Note that this evaluation method does not give any clues on how *effective* a method is, but it is more a way to choose between two different methods that perform similarly according to other evaluation indices. In such situations, the method that least affects the data should be preferred. This index is computed as being the average correlation coefficient between genes or samples before and after removing the batch effect. Low values will indicate that the batch effect removal distorted the initial data, and hence the method will be considered as inefficient. A very similar idea led to the *integrative correlation coefficient*, as described in [Cope et al. (2007)].

The correlation coefficient of genes and samples is implemented in the `inSilicoMerging` package by the `measureGenesMeanCorrCoef` and `measureSamplesMeanCorrCoef` functions respectively.

7.4 The inSilicoMerging R/Bioconductor Package

A public available package combining most of the batch removal and validation methods described in the previous sections was developed and added to the Bioconductor repository [Gentleman et al. (2004)]. This package, called inSilicoMerging[5], is able to merge different data sets by applying five different batch effect removal techniques (BMC, COMBAT, DWD, GENENORM and XPN, see Section 7.3.2 for more details).

The implementation of COMBAT was based on the R code made available by the original authors on their website[6]. The implementation of DWD uses the kDWD function from the corresponding DWD R package [Huang et al. (2012)]. XPN has been implemented based on Matlab code provided by the authors[7]. The remaining merging algorithms were implemented using basic R functions. All implementations were tested by empirical validation and where possible, by comparing results with those from the original implementations. Some methods are only reported and implemented to merge exactly two studies (e.g. XPN and DWD). In order to be able to merge any number of studies, this package added an additional step. This step combines all studies two-by-two and is called recursively on the intermediate results until only one, merged, dataset remains. Its behavior is illustrated in the following pseudo-code:

```
list of studies = [ A ; B ; C ; D ; E ]
m(X,Y) = applying merging technique 'm' on dataset 'X' and 'Y'
combineByTwo:
  iteration 1 : [ E ; m(A,B) ; m(C,D) ]   => [ E ; AB ; CD ]
  iteration 2 : [ CD ; m(E,AB) ]          => [ CD ; EAB ]
  iteration 3 : [ m(CD,EAB) ]             => [ CDEAB ]
```

[5] http://www.bioconductor.org/packages/release/bioc/html/inSilicoMerging.html

[6] http://www.bu.edu/jlab/wp-assets/ComBat/Downloadfiles/ComBat.R

[7] https://genome.unc.edu/xpn/

Evaluating and validating the results of batch effect removal methods is perhaps as important as the batch effect removal process itself. Without good and reliable evaluation tools, these methods could result in an even increased distortion of the data, introducing serious errors in the results of any downstream analysis performed. Therefore, five simple but powerful visual inspection tools and six quantitative measures to evaluate the different batch-effect removal methods, are provided as well (see Section 7.3.3 for more details).

In Table 7.2 all functions with their required and optional parameters are summarized. `batchAnnot` and `targetAnnot` are labels to retrieve for each sample the batch information and the target biological variable of interest respectively. This information is supposed to be included in the `ExpressionSet` structure. The three last methods in the table also require the data set before applying batch effect removal (`esetBefore`) to calculate its measure.

Function	*Req. Param*	*Opt. Param*
merge	esets, method	
plotGeneWiseBoxPlot	eset, batchAnnot	targetAnnot, gene
plotGeneWiseDensity	eset, batchAnnot	gene
plotDendrogram	eset, batchAnnot	
plotMDS	eset, batchAnnot	targetAnnot
plotRLE	eset, batchAnnot	
measureGenesOverlap	eset, batchAnnot	
measureSamplesOverlap	eset, batchAnnot	
measureAsymmetry	eset, batchAnnot	
measureSignificantGenesOverlap	esetBefore, esetAfter, batchAnnot, targetAnnot	
measureGenesMeanCorrCoef	esetBefore, esetAfter, batchAnnot	
measureSamplesMeanCorrCoef	esetBefore, esetAfter, batchAnnot	

Table 7.2: Summary of all relevant functions in the `inSilicoMerging` package with their Required and Optional parameters.

Several of the batch effect removal techniques included in the `inSilicoMerging` package have been implemented prior to this package. A major benefit of our package is the consistent interface for all merging methods and its integration in the R/Bioconductor framework. We will briefly discuss the added value of our package compared to those that are already available in R.

The authors of the original COMBAT method for example, work in their implementation with several matrices containing the numerical data, relevant annotation and covariates. Within our package, all this information in encoded in the `ExpressionSet` structure.

The authors of the DWD method recently have constructed the DWD R/ Bioconductor package [Huang et al. (2012)]. This package is intended as a general use of the distance-weighted discrimination technique and not specific for the goal of batch effect removal. It is therefore not straightforward to use for merging gene expression data sets and once again the relevant data is dispersed over several objects as opposed to using the `ExpressionSet` data structure. In addition, the necessary transformation of the gene expression values was added to this package in order to obtain a merged data set as result.

The CONOR package most closely approaches our software as it includes multiple batch removal methods and several methods based on discretizing the gene expression data. However, it lacks the user friendliness provided by our package and the direct integration with the Bioconductor framework. The CONOR package is available through the CRAN repository [Rudy & Valafar (2011)].

To the best of our knowledge no explicit software has been developed for validating the quality of the batch removal process of microarray gene expression data sets. The collection of both qualitative and quantitative validation methods is therefore a unique property of the `inSilicoMerging` package.

Although this package can be used as a stand-alone tool, its power lies in the combination with tools like the inSilicoDb package (see Section 4.4, thereby paving the way towards large-scale meta-analysis of complete gene expression repositories. An illustration of this combination can be found in the code fragment of Appendix A.6.

As of July 2012, the inSilicoMerging package is downloaded more than 500 times[8], although not even published. An article presenting the tool is currently under submission [Taminau et al. (Subm)].

[8] http://www.bioconductor.org/packages/stats/bioc/inSilicoMerging.html

8

Integrative Analysis of Microarray Data: an Application

In this chapter a practical application is presented combining all the information and tools from the previous chapters. A large-scale integrative analysis is performed on lung cancer microarray data to identify consistent and robust differentially expressed genes (DEGs) which can be used as promising biomarkers for lung cancer.

Lung cancer was selected as a case study for the identification of DEGs since it is one of the leading causes of cancer-related deaths in the world, with unfortunately one of the lowest survival rates within 5 years after diagnosis [Hayat et al. (2007)].

For this large-scale study different but similar microarray data sets are combined or integrated in order to find DEGs using information from multiple experiments. We will compare the two possible approaches for this integrative analysis: meta-analysis (see Chapter 6) and merging (see Chapter 7) and discuss the different results.

The rest of this chapter is structured as follows: We first provide all details concerning the different data sets and how we obtained them. Then the process to identify differentially expressed genes and the global methodology is explained. For both cases we present the results, followed by a discussion.

8.1 Data and Methods

In this section we present the data sets that will be used in this application and how we obtained them through carefully querying the InSilico DB. We then explain into detail the experimental setup we used to compare both meta-analysis and merging as potential methodologies for the large-scale integration of multiple datasets for the discovery of differentially expressed genes (DEGs) for lung cancer.

8.1.1 Data

A list of potential data sets for this application was programmatically retrieved from the InSilico DB using the `getDatasetList` function from the `inSilicoDb` package (see Section 4.4, [Taminau et al. (2011)b]). This list was further restricted by defining the following constraints:

- Only fRMA processed studies were considered, i.e. studies for which the original CEL files were available and which were consistently preprocessed by the internal InSilico genomic pipeline. See Section 4.3.1 for more details.

- Each study should contain at least 30 samples in order to be able to be statistically relevant.

- Each study should contain both samples from normal tissue and from lung cancer tissue, more or less equally distributed. In order to achieve this we looked at the "Disease" keyword which is available in most curations and filtered on "lung cancer"|"adenocarcinoma" values and "control"|"normal"|"healthy" values for lung cancer and normal samples respectively.

- Only studies assayed on Affymetrix Human Genome U133A (GPL96) and Affymetrix Human Genome U133 Plus 2.0 (GPL570) were taken into consideration.

This search resulted in a list of six studies, summarized in Table 8.1. For each dataset a new curation was made and stored in the InSilico DB to make it trackable. These curations contain the Disease keyword with control and lung cancer as keywords and are used as such through the rest of this chapter.

Data set	*Platform*	*#Genes*	*#Samples* *(control/cancer)*	*Reference*
GSE10072	GPL96	12718	**107** (49/58)	[Landi et al. (2008)]
GSE7670	GPL96	12718	**66** (27/27)	[Su et al. (2007)]
GSE31547	GPL96	12718	**50** (20/30)	**NA**$^{(i)}$
GSE19804	GPL570	19798	**120** (60/60)	[Lu et al. (2010)]
GSE19188	GPL570	19798	**156** (65/91)	[Hou et al. (2010)]
GSE18842	GPL570	19798	**91** (45/46)	[Sanchez-Palencia et al. (2011)]
Total			**590** (312/266)	

Table 8.1: List of six lung cancer microarray data sets used in this application.
$^{(i)}$This dataset has no accompanying publication.

8.1.2 Identification of Differentially Expressed Genes.

A very common objective and application of microarray studies is the identification of genes that are consistently and significantly expressed according to a target biological variable of interest. These genes are called informative genes, biomarkers or differentially expressed genes (DEGs). Many methods and approaches to find DEGs exist and here we opted for the R/Bioconductor `limma` package [Smyth (2004)]. A recent and detailed overview of all possible methods can be found in [Lazar et al. (2012)b].

After applying limma we call every gene significantly differentially expressed if:

- it that has an adjusted p-value lower than 0.05
- it has a log fold change higher than 2

DEG lists should also be *robust* or *consistent*. In order to test the robustness of DEG lists we implemented an extra resampling step on top of the limma method. In each iteration, we randomly keep 90% of the samples and apply limma to obtain a DEG lists fulfilling the two above mentioned criteria on this subset. After n iterations we obtained n different DEG lists and our final, robust, DEG list will be the intersection of those lists. Since we take the intersection over all iterations it seems intuitive that by increasing the number of iterations n the size of the final intersection will be decreased. In our simulations we found however that there is mostly a convergence around 50 iterations, depending on the quality of the study. This is illustrated for the study `GSE10072` in Figure 8.1(a), the other five studies have similar graphs. To play safe we always used a iteration size n of 100 for all our experiments.

One could argue that taking the intersection over all iterations is quite strict and maybe we are missing DEGs that for example are present in 99% or 95% of the lists. This however rarely happens since almost all genes either appear either in all DEG lists of all iterations, either in the DEG list of only one iteration. This is again illustrated for study `GSE10072` in Figure 8.1(b). This means that genes found as DEGs on the complete data set, but not after using the resampling test are definitely *false positives* and by only looking at the intersection we almost don't miss any potential *false negatives*. This once more strengthens the importance of using appropriate robustness checks.

Finally, the procedure to find significantly and robust DEGs can be found in Appendix A.7.

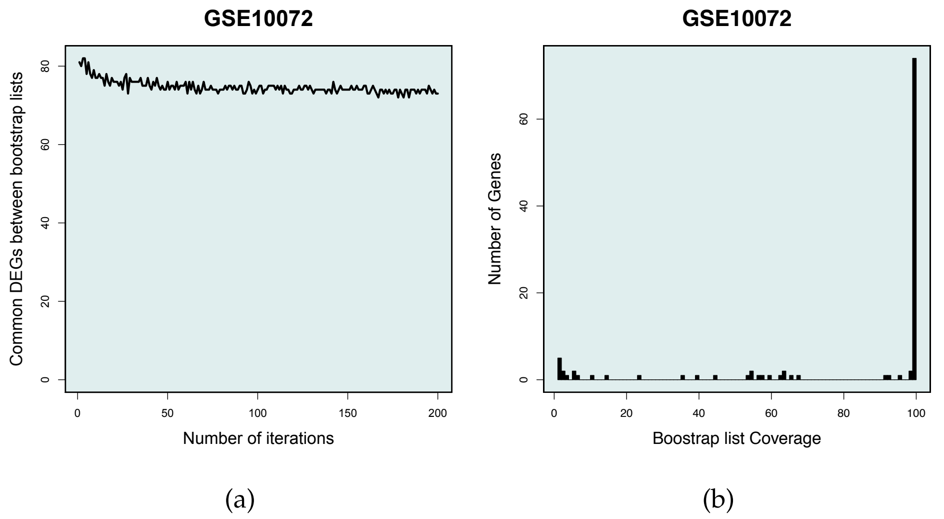


Figure 8.1: Result of simulations to motivate the selected parameters for the resampling test. (a) the length of the intersection of all DEG lists in function of the number of iterations. Convergence can be observed around 50 iterations. (b) In this plot the frequency of all genes over all iterations are shown. The most right bar are all genes belonging to each DEG list, i.e. our final robust DEG list. This simulation was performed with 100 iterations.

8.1.3 Experimental Setting

The workflow for both meta-analysis and merging approaches was already visualized in Figure 5.1. For the meta-analysis part, see Figure 5.1(a), we obtain a robust DEG list for each of the six studies and then look at the intersection of those DEG lists. This final list of DEGs will contain all genes that were found to be informative in all single studies.

For the merging part, see Figure 5.1(b), we first merge all six studies into one global one using the following batch effect removal methods from the `inSilicoMerging` package: NONE (no batch effect removal), BMC (batch-mean centering, [Sims et al. (2008)]), COMBAT (empirical bayes, [Johnson et al. (2007)]), DWD (distance-weighted discrimination, [Benito et al. (2004)]) and XPN (cross-platform normalization, [Shabalin et al. (2008)]. More information on the different batch effect can be found in Section 7.3.2. Then we applied on each merged data set the same procedure to find robust DEGs and we immediately have our final list of the DEGs.

For both cases we also applied the same methodologies by using renormalized data sets using RMA instead of fRMA in order to confirm all results.

8.2 Results and Discussion

In this section we will present and discuss the gene lists found by the two approaches:

8.2.1 Meta-Analysis Approach

We first look at the results of the meta-analysis case by looking at the number of DEGs obtained on the single data sets as listed in Table 8.2. In the third column the number of DEGs without using resampling is shown, followed by the number of DEGs after applying resampling as explained in the previous section. We notice that using this resampling strategy leads to a decrease in the number of DEGs for all data sets (ratios between 60-80% depending on the specific data set).

Another observation that can be made is the higher number of DEGs for the last three data sets. This difference is probably due to the difference in

$Preprocess$	$Data\ set$	$\#DEGs^{(i)}$	$\#DEGs$	$\#DEGs^{(ii)}$
			$(resamp)$	$(intersection)$
RMA	GSE10072	81	70	25
	GSE7670	81	49	
	GSE31547	52	35	
	GSE19804	161	112	
	GSE19188	405	325	
	GSE18842	632	503	
$fRMA$	GSE10072	90	74	25
	GSE7670	79	52	
	GSE31547	67	43	
	GSE19804	158	109	
	GSE19188	351	284	
	GSE18842	499	398	

Table 8.2: Number of differentially expressed genes (DEGs) for all single data sets. The final result of this meta-analysis case is the intersection of the different lists in the last column.
$^{(i)}$Number of DEGs found on the complete data set without resampling.
$^{(ii)}$Number of DEGs in the intersection of the DEG lists of all single data sets after using resampling.

platform: GPL96 for the first three studies and GPL570 for the last three studies, see Table 8.1. Since the former platform has more than 7000 genes less than the latter platform a higher chance of finding DEGs is obvious. Also note from Table 8.1 that the average sample size for platform GPL96 is around 74, while for platform GPL570 it is around 122, this also can have a minor effect on the robustness of DEGs.

The final list of DEGs in the meta-analysis case can be obtained by taking the intersection of all single-study DEG lists. This list of 25 genes consists of genes that are consistently differentially expressed in all six studies and can be considered as the most promising list of biomarkers for lung cancer, based on our input data. Note that using RMA or fRMA does not significantly influence the number of found DEGs in all studies. The final gene lists in both cases are not identical but have an overlap of 23 genes. Below are the fRMA DEGs listed:

```
> degs_meta
 [1] "ABCA8"     "ADH1B"     "AGER"      "C10orf116" "CAV1"      "CD36"
 [7] "CDH5"      "CLDN18"    "CLEC3B"    "CLIC5"     "EDNRB"     "FABP4"
[13] "FAM107A"   "FCN3"      "FHL1"      "FOSB"      "HBB"       "KAL1"
[19] "LDB2"      "MT1M"      "SPP1"      "TCF21"     "TMEM100"   "TNNC1"
[25] "WIF1"
```

Many of those genes are already identified as potential biomarkers: `AGER`, `CLIC5` and `TNNC1` for example were confirmed to be related to survival outcome in non-small cell lung cancer via RT-PCR validation [Urgard et al. (2011)]. `EDNRB` was already identified as a promising candidate biomarker for lung cancer [Knight et al. (2009)]. `FHL1`, `FABP4`, `EDNRB` and `AGER` were also selected as lung adenocarcinoma marker genes after joint analysis of two microarray gene expression data sets [Jiang et al. (2004)]. `CAV1` is believed to be a tumor suppressor. and was found to be down-regulated in many types of cancers including lung cancer [Sunaga et al. (2004)]. Glycoproteomic analysis revealed `HBB` as a differentially expressed protein [Rho et al. (2009)].

Other genes (e.g. `CD36`, `CLEC3B`, `C10orf116` and `MT1M`) have not been reported to be (lung) cancer related yet and might present new discoveries.

8.2.2 Merging Approach

In the merging part, all six data sets are first *merged* into one global data set and thus only one DEG list is finally retrieved (see Figure 5.1 for a reminder). Different batch effect removal methods can however be applied, resulting in different DEG lists. The results are presented in Table 8.3 by listing the number of found DEGs for every batch effect removal method and preprocessing method (RMA or fRMA).

$Preprocess$	$BERM^{(i)}$	$\#DEGs^{(ii)}$	$\#DEGs$	$\#DEGs^{(iii)}$
			$(resamp)$	$(intersection)$
RMA	NONE	160	128	118
	BMC	151	129	
	COMBAT	151	129	
	DWD	151	130	
	XPN	170	146	
$fRMA$	NONE	131	112	102
	BMC	124	109	
	COMBAT	125	110	
	DWD	125	111	
	XPN	143	123	

Table 8.3: Number of differentially expressed genes (DEGs) for all merged data sets
$^{(i)}$BERM: Batch Effect Removal Method.
$^{(ii)}$Number of DEGs found on the complete data set without resampling.
$^{(iii)}$Number of DEGs in the intersection of the DEG lists of all single data sets after using resampling.

We can make a number of interesting observations based on the results of Table 8.3:

Resampling is still needed

After merging there is still a need for resampling as it clearly helps to remove false positives, although the difference in number of DEGs with and without resampling is less prominent than in the meta-analysis approach (ratios between 85-89% for the different batch effect removal methods).

Using fRMA results in less DEGs than using RMA

Using fRMA as preprocessing method results in significantly less DEGs than using RMA preprocessing for all batch effect removal methods. This was not the case in the meta-analysis approach for the single data sets and was not expected. No reasonable explanation can be found at this moment and although very interesting to find out, it currently remains an open question. For the remainder of this application we will concentrate us on the fRMA results.

Relatively low impact of using batch effect removal

With the exception of the XPN method, the methods BMC, COMBAT and DWD are not able to find significantly more DEGs than when no batch effect removal at all is performed. In Figure 8.2 the MDS plots of the merged data set NONE is shown after both RMA and fRMA preprocessing and in both cased a clear batch effect can be observed[1]. With this undesirable effect present, the similar results of NONE and the other batch effect removal methods are remarkable.

The explanation lies in the fact that MDS plots provide a *global* view on the data, while the identification of DEGs is more based on local effects, i.e. the specific expression of one gene in certain conditions. Not all genes are affected by batch effect removal in the same way, and for those genes batch effect removal methods will not change anything. Moreover, even if a gene is affected by batch effect removal we observed that the *difference* between the two modes or conditions of the gene (in our case control versus lung cancer) is almost always preserved over all samples of the merged data set. Two more in-depth examples of genes are provided to illustrate this further.

To select specific genes to investigate in detail we can look at all genes

[1] Note that fRMA already succeeds in grouping the samples per platform instead of per study like RMA. In this regard fRMA can already be seen as a special, platform-specific batch effect removal method.

identified as DEGs by the NONE method but not by for example the COMBAT method:

```
> setdiff(degs_merged[["NONE"]], degs_merged[["COMBAT"]])
[1] "ADRB1"   "AKR1B10" "CCNB1"   "DLGAP5"  "DSP"
```

Gene ADRB1 for example is identified as a DEG without batch effect removal but not afterwards. Is the batch effect removal method distorting the biological signal of this gene?

We can look at the boxplots of this specific gene in Figure 8.3. On the top left plot we can notice that this gene is only differentially expressed in three studies and those three studies are from the same GPL570 platform. For the other studies from the GPL96 platform, the situation is completely different with an almost stable expression of the ADRB1 gene. The difference in expression in the three studies is however big enough to bias the global expression as being differentially expressed, as can be seen in the bottom left plot. If we apply batch effect removal, all samples from both platforms are brought closer together and thereby decreasing the influence of the differentially expression of platform GPL570. This results in a global expression that is not differentially expressed anymore, see bottom right plot[2].

For the other genes AKR1B10, CCNB1, DLGAP5 and DSP which were only identified by the NONE method a similar situation occurs. From one point of view COMBAT and the other batch effect removal methods indeed *remove* a biological relevant signal that *is* present in the data, or at least part of the data, but one can argue that this signal is not consistent across all individual studies and can be due to a technical, platform-dependent artifact.

We also investigate another set of genes, i.e. genes that are only found after applying batch effect removal. This time we list all genes that were

[2] Note that a log fold change higher than 2 was defined as a requirement for being identified as *differentially expressed*

identified as beging a DEG by the COMBAT method, but not by the NONE method:

```
> setdiff(degs_merged[["COMBAT"]], degs_merged[["NONE"]])
[1] "LRRN3" "PDK4"  "TPX2"
```

The boxplots in Figure 8.4 provide us again a more in-depth look for the first gene in this list, gene LRRN3. If we compare the top left and top right plots we can see that COMBAT nicely removes the batch effect between the different studies for this gene and creates a clear and consistent differentially expression profile across all samples. This leads to a situation in which this gene is labeled as differentially expressed by the COMBAT method, but not by the NONE method since it, just slightly, fails for the log fold change requirement. In this case, instead of a technical artifact, it is actually the batch effect that distorts the global expression profile of the LRRN3 gene.

These two examples show that there are small differences between the different batch effect removal methods but merely comes down to genes that fall just above or below the defined thresholds for the log fold change. To compare the merging approach with the meta-analysis approach we therefore use the intersection of the DEGs found by the different batch effect removal methods for fRMA. This list consists of 107 genes (see Table 8.3).

8.2.3 Comparing Meta-Analysis and Merging Approaches

If we compare the final DEGs for the meta-analysis approach with the list obtained in the merging approach we can conclude that more DEGs are identified through merging. Moreover, all 25 identified DEGs through meta-analysis are also identified in the merging approach. Below we list the additional genes identified as being DEGs and for a few selected ones we demonstrate their relation to lung cancer through literature.

```
> setdiff(degs_merged, degs_meta)
 [1] "ACADL"    "ADAMTS8"  "ADRB2"    "AOC3"     "AQP1"     "AQP4"
 [7] "ASPM"     "C13orf15" "C7"       "CA4"      "CACNA2D2" "CAV2"
[13] "CDC20"    "CEACAM5"  "CFD"      "CHRDL1"   "CLIC3"    "COL10A1"
```

```
[19] "COL11A1" "CPB2"    "CXCL13"  "CXCL2"   "DLC1"    "DUOX1"
[25] "EMCN"    "FIGF"    "FMO2"    "FOXF1"   "GIMAP6"  "GINS1"
[31] "GPM6A"   "GPX3"    "GREM1"   "HMGB3"   "HPGD"    "HSD17B6"
[37] "IL33"    "IL6"     "KLF4"    "KRT6A"   "LAMP3"   "LEPR"
[43] "LIMCH1"  "LPL"     "MARCO"   "MFAP4"   "MMP1"    "MMP12"
[49] "OLR1"    "PDZD2"   "PGC"     "PIP5K1B" "PLA2G1B" "PPBP"
[55] "RAMP3"   "RRM2"    "S100A2"  "SCGB1A1" "SCN7A"   "SDPR"
[61] "SFTPA2"  "SFTPC"   "SFTPD"   "SLC1A1"  "SLC39A8" "SLC6A4"
[67] "SOSTDC1" "SPINK1"  "SPOCK2"  "STXBP6"  "SULF1"   "TGFBR3"
[73] "TMPRSS4" "TOP2A"   "UPK3B"   "VIPR1"   "ZBTB16"
```

Together with CAV1, which was also identified in the meta-analysis approach, CAV2 also is related to survival in lung cancer [Wikman et al. (2004)]. C13orf15, formerly named RGC32, is playing an important role in the pathogenesis of nonsmall cell lung cancer [Kim et al. (2011)]. It was also demonstrated that chemokines (CXCL13 and CXCL2) are pivotal determinants of the angiogenic activity of non-small cell lung cancer [Strieter et al. (1995)]. The results in [Feng et al. (2001)] suggest that S100A2 expression is suppressed early during lung carcinogenesis and that its loss may be a contributing factor in lung cancer development or a biomarker in this process.

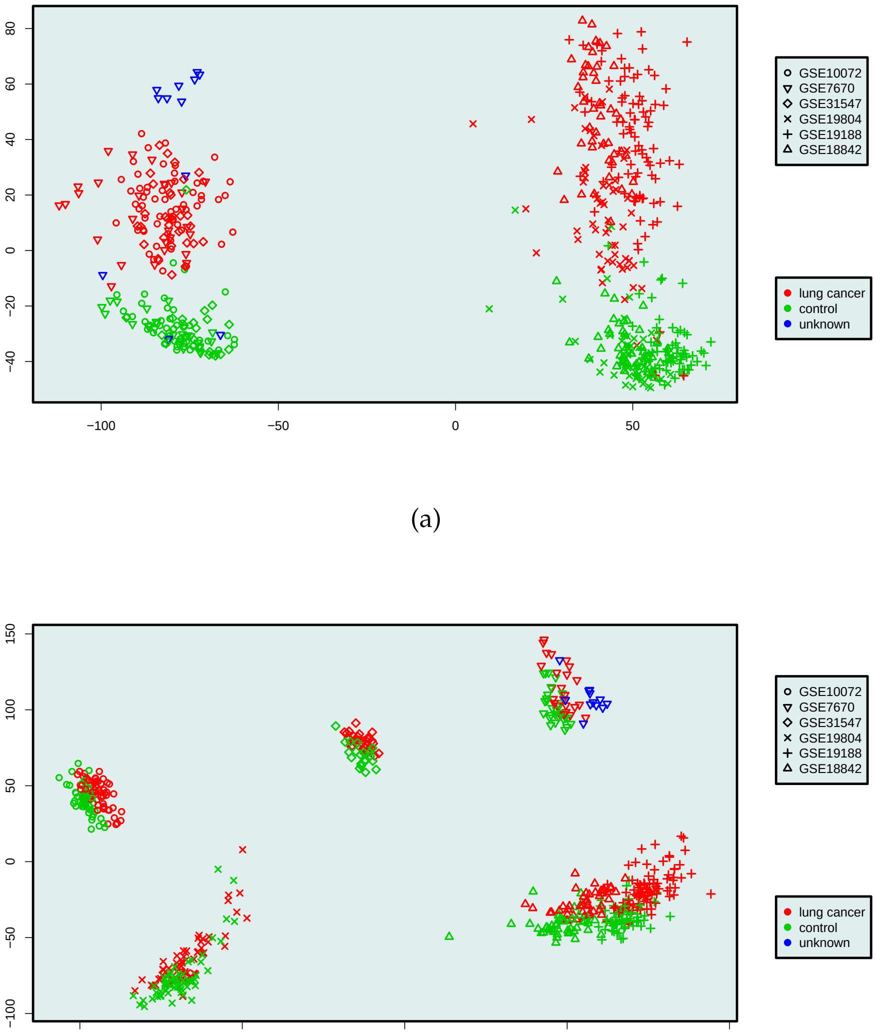
A more in depth and complete scan of the literature is beyond the scope of this thesis but similarly most genes are proven or suggested to play an important role in the development of lung cancer and are potential biomarkers.

Significantly more DEGs can be found with the merging approach than with the meta-analysis approach. This confirms an earlier statement made by [Xu et al. (2008)]: *"The major limitation of meta-analyses is that the small sample sizes of individual studies, coupled with variation due to differences in study protocols, inevitably degrades the results. Also, deriving separate statistics and then averaging is often less powerful than directly computing statistics from aggregated data."*

8.3 Conclusions

After comparing meta-analysis and merging as two potential approaches for the identification of differentially expressed genes through the integration of multiple microarray gene expression data sets we can conclude that significantly more DEGs can be found through merging than through meta-analysis. The final DEG list found after merging the six data sets consists of 107 genes, includes all genes found after meta-analysis, and is showed to be stable and robust after applying a strict resampling procedure. A brief literature study showed relevance for most of these genes and this list can be considered as an ideal starting point for further analysis.

Interestingly, the identification of DEGs is not hindered by batch effects when the different data sets are merged together. The different batch effect removal methods give very similar gene lists. We looked at two genes that behave differently due to batch effects in detail to explain this difference and found out that they always are flirting with the log fold change threshold we defined.



(a)

(b)

Figure 8.2: Multidimensional scaling (MDS) plots of the merged data set with no batch effect removal. (a) after applying fRMA as preprocessing method for the individual data sets and (b) after applying RMA as preprocessing method for the individual data sets. Samples are colored based on the target biological variable of interest and the different symbols correspond to the individual studies.

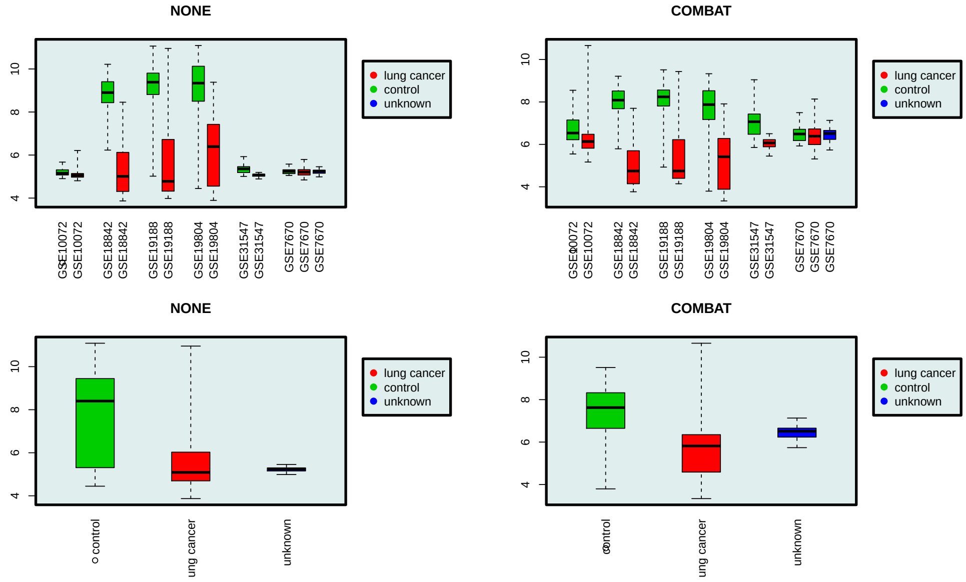


Figure 8.3: Different boxplots for `ADRB1` gene. On the left we have two boxplots for the merged data set without batch effect removal (NONE) and on the right for the merged data set with batch effect removal (COMBAT). All boxplots are grouped and colored based on the target biological variable of interest, the boxplots on top are further grouped per original data set.

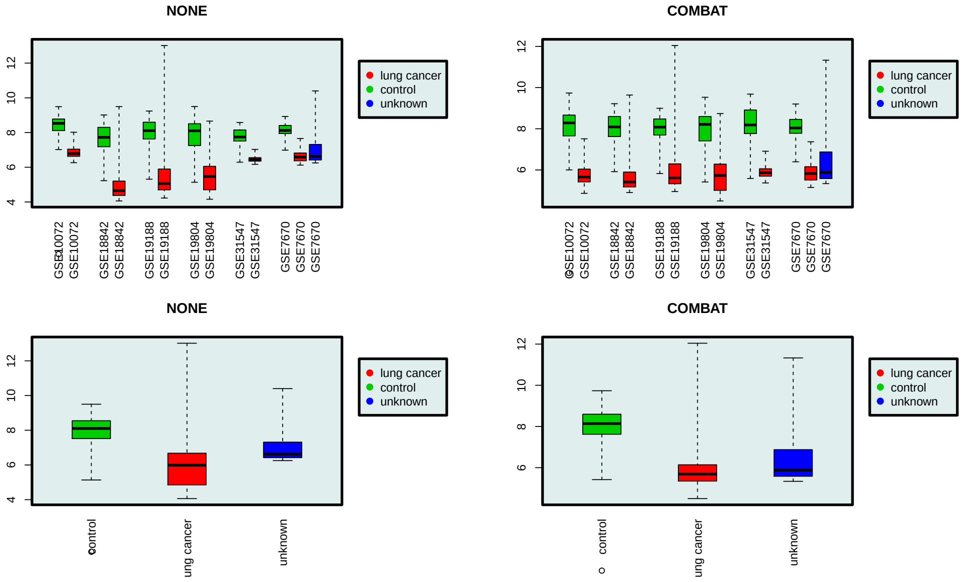


Figure 8.4: Different boxplots for `LRRN3` gene. On the left we have two boxplots for the merged data set without batch effect removal (NONE) and on the right for the merged data set with batch effect removal (COMBAT). All boxplots are grouped and colored based on the target biological variable of interest, the boxplots on top are further grouped per original data set.

9 Conclusions

In this last chapter we will summarize the results and contributions of this thesis, followed by an enumeration of the open and promising research directions for future work.

9.1 Overview

After more than a decade of microarray gene expression research a vast amount of data became publicly available through online repositories. It is clear that the new challenges for this technology lie in the integration of this plenitude of available data in order to obtain more robust, accurate and generalizable results. The integrative analysis of this data is inexpensive since it happens on a computational level without need for extra wet-lab experiments. Information from different studies can be integrated or combined on the interpretation level in a typical *meta-analysis* set-up or the gene expression values can be combined directly, creating a virtual *combined* gene expression matrix.

For both approaches we identified however several issues and two main hurdles are currently hindering the success of large-scale integrative analysis of microarray gene expression data.

The first hurdle is the problem of retrieving genomic data in a consistent and unambiguously way. This is mainly due to the complex preprocessing options for the numerical data which are not always sufficiently documented, and the lack of standards to describe phenotypic meta-data. These issues lead to many unnecessary manual interventions to parse data from one format to another, leading to error-prone situations that are not trackable or reproducible. To overcome these limitations we presented the InSilico DB, a tool that provides expert-curated and consistently preprocessed gene expression data with direct access to various analysis software kits.

The actual integration of different gene expression studies is also hindered by study-specific biases, or batch effects, originating at the various technical steps throughout a classic microarray experiment. Labs use different protocols, experimental settings, parameters and even different arrays, leading to incompatible data, both on the numerical level as on the meta-data or phenotypical level. An extensive survey of batch effect removal methods and the corresponding validation tools are provided in a unified framework. A R/Bioconductor package combining all this functionality was also developed and presented.

Finally we presented two practical applications to demonstrate the usefulness of integrating multiple microarray gene expression studies. These two applications use all software we made available in a clear and illustrative manner.

9.2 Contributions

Our first contribution was in the development of highly specialized genomic pipelines for the preprocessing of gene expression microarray data. These pipelines are controlled and guided by a supervised process called the *inSilico* backbone. Both the pipelines and the backbone are essential pieces of the inSilico DB framework and enables the management of expert-curated and consistently preprocessed genomics data.

Next, an extensive overview of batch effect removal methods for the merging of gene expression data was presented. This survey is novel and complete since for the first time a unified notation was used to describe all the methods. No other similar work exists so far and an article submission was well accepted as a useful contribution to the research domain.

Two open-source and well-documented R/Bioconductor packages were developed and released. Both have the aim of overcoming the current issues of integrative analysis of gene expression analysis. The `inSilicoDb` package enables consistent and trackable retrieval of genomic data and the `inSilicoMerging` package provides tools to merge multiple datasets. Both packages seamlessly integrate which each other and are already used by the scientific community.

Two large-scale integrative applications using the massive amount of data available in the InSilico DB were presented, both containing innovative elements in their workflow. In the first application we screened for genes with a stable expression across various biological conditions and obtained a compact but diverse set of 12 reference genes. In the second application we used both a merging and a meta-analysis approach to identify differentially expressed genes for lung cancer. Both applications extensively use the developed and freely available `inSilicoDb` and `inSilicoMerging` packages and key code excerpts are provided for full transparency and reproducibility.

A more detailed summation of the contributions can be found in Section 1.3 and in Appendix B a complete list of the publications obtained during this thesis can be found.

9.3 Future Directions

We can divide the future work into two main categories: extensions of the developed R/Bioconductor packages and further validation of the presented applications:

9.3.1 Extensions of the R/Bioconductor Packages

The `inSilicoDb` package needs to adapt fast to any new type of data or to new methodologies. For example the retrieval of RNA-Seq data (see Section 2.2.2) was made possible upon request recently. Another ongoing update is the option of using *custom CDF* files from the BrainArray group of the University of Michigan [Dai et al. (2005)]. CDF files contain information about where each probe is located on the chip and which probes go together to form a probe set for a specific gene. By default this mapping is proposed by the manufacturer Affymetrix [Affymetrix (2002)], but alternatives emerged, showing an improvement in precision and accuracy [Sandberg & Larsson (2007)]. The custom CDF proposed by the BrainArray group has the additional advantage that all probes have a one-to-one mapping with the target genes, avoiding the need for a specific probe-to-gene mappings.

For the `inSilicoMerging` package the same can be applied. New batch effect removal techniques should be added to the package. Moreover we developed our own batch removal method: GENESHIFT, a non-parametric method based on two key elements from statistics: empirical density estimation and Kullback-Leibler divergence. This method is now being tested using the `inSilicoMerging` framework and will soon be published.

9.3.2 Further Validation of the Two Applications

The application presented in Section 6.3 with the screening of stable genes through meta-analysis needs some extra validation. Currently we have a collaboration with Dr. Bram de Craene from the VIB Departement Moleculair Biomedisch Onderzoek (UGent) and our proposed set of 12 reference genes is being tested in qRT-PCR experiments to compare its performance to current state-of-the-art genes. Based on these results an adaptation or refinement of the screening process can be made.

For the final application in which we discovered several potential biomarkers for lung cancer there is also room for a similar opportunity to test and validate the proposed genes. An initial analysis by mapping the genes onto pathways already revealed interesting results and can guide further analysis. For example overrepresentation of the following pathways were found when mapping genes to the KEGG database [Kanehisa & Goto (2000)]: Cytokine-cytokine receptor interaction, PPAR signaling pathway and Focal adhesion. Contacts with experienced clinicians in the oncology field were made for further collaborations.

A

Code Fragments Used in this Thesis

A.1 Code to obtain average number of samples per study from InSilico DB

```
#-------------------------------------------------------------------------------
# Get list of available datasets on GPL570 platform
#-------------------------------------------------------------------------------
library("inSilicoDb");
lst = getDatasetList(gpl="GPL570");

#-------------------------------------------------------------------------------
# Sum number of samples for all studies
#-------------------------------------------------------------------------------
total = 0;
for(gse in lst)
{
  annot = pData(getAnnotations(gse, gpl="GPL570"));
  total = total + nrow(annot);
}

```

```
#-------------------------------------------------------------------------------
# Print output
#-------------------------------------------------------------------------------
cat("Number of studies on GPL570 platform in InSilicoDB:",length(lst),"\n");
cat("Average number of samples:",total/length(lst),"\n");
```

A.2 Code to obtain heatmap from GSE4635 dataset

```
#-------------------------------------------------------------------------------
# Retrieve dataset
#-------------------------------------------------------------------------------
library("inSilicoDb");
eset = getDataset("GSE4635", "GPL96", norm="FRMA", genes=TRUE);

#-------------------------------------------------------------------------------
# Find significant genes
#-------------------------------------------------------------------------------
library("limma");
labels = pData(eset)[ ,"Smoker"];
design = model.matrix(~labels);
fit = eBayes(lmFit(eset, design));
t = topTable(fit, coef=2)[,c("SYMBOL","logFC","adj.P.Val")];
deg = t[sort(t[,"logFC"],index=TRUE, decr=TRUE)$ix,"SYMBOL"];

#-------------------------------------------------------------------------------
# Save heatmap in pdf file
#-------------------------------------------------------------------------------
pdf(file="../Figures/heatmap.pdf");
heatmap(exprs(eset[deg,]), labCol=labels, margins=c(4,4), Rowv=NA);
dev.off();
```

A.3 Code to obtain mds plot from GSE19804 dataset

```
#-------------------------------------------------------------------------------
# Retrieve dataset
#-------------------------------------------------------------------------------
library("inSilicoDb");
eset = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);

#-------------------------------------------------------------------------------
# Custom plot function
#-------------------------------------------------------------------------------
myPlot = function(mds, labels)
{
  #-- Add margin to the right for the legend
  tmp = par()$mar;
  par(xpd=T, mar=par()$mar+c(0,0,0,8));

  plot(mds$points,col=labels,lwd=2,xlab="",ylab="");

  #-- Plot legend
  range_x = range(mds$points[,1]);
  range_y = range(mds$points[,2]);
  x = range_x[2] + (range_x[2]-range_x[1])*0.1;
  y = range_y[2] - (range_y[2]-range_y[1])*0.1;
  labels = unique(labels);
  legend(x,y,legend=labels,pt.lwd=2,pch=1,col=labels);

  #-- Reset Margin
  par(xpd=F, mar=tmp);
}

#-----------------------------------------------------------------------------
# Save mds plot in pdf file
#-----------------------------------------------------------------------------
pdf(file="../Figures/mds.pdf");
mds = cmdscale(dist(t(exprs(eset))), x.ret=TRUE);
labels = pData(eset)[ ,"Disease"];
myPlot(mds, labels);
dev.off();
```

A.4 Example code to generate fRMA sample in InSilico DB

```
#-----------------------------------------------------------------------------
# Parameters of script:
#-----------------------------------------------------------------------------
# gpl       (platform id)
# cel_path  (path of input CEL files)
# name      (name of the output file)
# out_path  (path of the ouput file)

#-----------------------------------------------------------------------------
# function to find information from annotation maps
#-----------------------------------------------------------------------------
getFromMap = function(annot, id, probes)
{
  res = mget(probes, ifnotfound=NA, get(paste(annot,id,sep="")));
  return(as.vector(unlist(as.list(res))));
}

#-----------------------------------------------------------------------------
# (1) define appropriate Bioconductor packages
#-----------------------------------------------------------------------------

annot = NULL;
```

```
if(gpl=="GPL570")       { annot = "hgu133plus2"; }
if(gpl=="GPL571")       { annot = "hgu133a2"; }
# [...] for all platforms
if(is.null(annot)) { stop("Unknown platform: ",gpl); }

#-------------------------------------------------------------------------------
# (2) perform frma preprocessing
#-------------------------------------------------------------------------------

library(frma);
library(affy);
abatch = ReadAffy(filenames=cel_path, cdfname=annot);

#-- For some platforms background info is public
if(annot=="hgu133plus2" | annot=="hgu133a")
{
  eset = frma(batch);
}

#-- For others we had to build them ourselves
else
{
  frmavecs = paste(annot,"frmavecs",sep="");
  library(frmavecs, character.only=TRUE);
  eset = frma(abatch,input.vecs=data(package=frmavecs));
}

#-------------------------------------------------------------------------------
# (3) Add feature information (probe/gene info) to eset
#-------------------------------------------------------------------------------

lib = paste(annot,".db",sep="");
library(lib, character.only=TRUE);

features = cbind(getFromMap(annot, "ENTREZID", rownames(data)),
                 getFromMap(annot, "SYMBOL",   rownames(data)),
                 getFromMap(annot, "GENENAME", rownames(data)));
colnames(features) = c("ENTREZID", "SYMBOL", "GENENAME");
rownames(features) = rownames(exprs(eset));

featureData(eset) = new("AnnotatedDataFrame",
                        data = as.data.frame(features));

#-------------------------------------------------------------------------------
# (4) Add annotation info to eset
#-------------------------------------------------------------------------------

annotation(eset) = annot;
version = installed.packages()[lib,"Version"];
notes(eset)[[paste(annot,"Version",sep="")]] = version;

#-------------------------------------------------------------------------------
```

```
# (5) Store eset to filesystem
#-------------------------------------------------------------------------------

assign(name,eset);
save(list=name, file=out_path);
```

A.5 Function to calculate the semantic similarity of two genes

```
library("org.Hs.eg.db");
library("GO.db");
library("GOSemSim");

#-------------------------------------------------------------------------------
# Generic function to retrieve info from bioconductors 'Bimaps' structure
#-------------------------------------------------------------------------------
getFromMap = function(map, id)
{
  res = mget(as.character(id), ifnotfound=NA, map);
  return(as.vector(unlist(as.list(res))));
}

#-------------------------------------------------------------------------------
# Get Gene Ontology ids for a specific ontology (MF, BP or CC)
#-------------------------------------------------------------------------------
getGoIds = function(gene, ont)
{
  id = getFromMap(org.Hs.egSYMBOL2EG, gene);
  goIds = mappedRkeys(org.Hs.egGO[as.character(id)])
  idx = sapply(goIds, function(x){Ontology(x)==ont});
  return(goIds[idx]);
}

#-------------------------------------------------------------------------------
# Get similarity score for a specific ontology (MF, BP or CC)
#-------------------------------------------------------------------------------
getSimilarityForOnt = function(gene1, gene2, ont)
{
  go1 = getGoIds(gene1, ont);
  go2 = getGoIds(gene2, ont);
  return(mgoSim(go1,go2,ont));
}

#-------------------------------------------------------------------------------
# Get semantic similarity score
#-------------------------------------------------------------------------------
getSimilarity = function(gene1, gene2)
```

```
{
  sim_MF = getSimilarityForOnt(gene1, gene2, ont="MF");
  sim_BP = getSimilarityForOnt(gene1, gene2, ont="BP");
  return((sim_MF + sim_BP) / 2);
}
```

A.6 Code illustrating all visualization tools for the validation of batch effect removal methods

```
source("http://bioconductor.org/biocLite.R")

#-------------------------------------------------------------------------------
# retrieve two datasets
#-------------------------------------------------------------------------------
biocLite("inSilicoDb");  # get latest version from bioconductor
library("inSilicoDb");
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

#-------------------------------------------------------------------------------
# merge the two datasets (using "NONE" and "COMBAT" methods)
#-------------------------------------------------------------------------------
biocLite("inSilicoMerging");  # get latest version from bioconductor
library("inSilicoMerging");
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

#-------------------------------------------------------------------------------
# (1) Boxplots (code to generate Figure 7.7)
#-------------------------------------------------------------------------------

gene = "MYL4"
# ... or take take random gene
#gene = sample(rownames(exprs(eset_NONE)), 1);

main = paste("NONE"," (gene = ",gene,")",sep="");
plotGeneWiseBoxPlot(eset_NONE, batchAnnot="Study", gene=gene, main=main,
                    legend=FALSE);
main = paste("COMBAT"," (gene = ",gene,")",sep="");
plotGeneWiseBoxPlot(eset_COMBAT, batchAnnot="Study", gene=gene, main=main,
                    legend=FALSE);

#-------------------------------------------------------------------------------
# (2) Density Plots (code to generate Figure 7.8)
#-------------------------------------------------------------------------------
```

```

gene = "MYL4"
# ... or take take random gene
#gene = sample(rownames(exprs(eset_NONE)), 1);

main = paste("NONE"," (gene = ",gene,")",sep="");
plotGeneWiseDensity(eset_NONE, batchAnnot="Study", gene=gene, main=main,
                    legend=FALSE);
main = paste("COMBAT"," (gene = ",gene,")",sep="");
plotGeneWiseDensity(eset_COMBAT, batchAnnot="Study", gene=gene, main=main,
                    legend=FALSE);

#-------------------------------------------------------------------------------
# (3) Dendrograms (code to generate Figure 7.9)
#-------------------------------------------------------------------------------

# take 40 random samples...
idx = sample(1:ncol(eset_NONE), 40);

plotDendrogram(eset_NONE[,idx], batchAnnot="Study", main="NONE",
               legend=FALSE);
plotDendrogram(eset_COMBAT[,idx], batchAnnot="Study", main="COMBAT",
               legend=FALSE);

#-------------------------------------------------------------------------------
# (4) MDS plots (code to generate Figure 7.10)
#-------------------------------------------------------------------------------

plotMDS(eset_NONE, batchAnnot="Study", targetAnnot="Disease", main="NONE",
        legend=TRUE);
plotMDS(eset_COMBAT, batchAnnot="Study", targetAnnot="Disease", main="COMBAT",
        legend=TRUE);

#-------------------------------------------------------------------------------
# (5) RLE plots (code to generate Figure 7.11)
#-------------------------------------------------------------------------------

# take 40 random samples...
idx = sample(1:ncol(eset_NONE), 40);

plotRLE(eset_NONE[,idx], batchAnnot="Study", main="NONE",
        legend=FALSE);
plotRLE(eset_COMBAT[,idx], batchAnnot="Study", main="COMBAT",
        legend=FALSE);
```

A.7 Function used to obtain robust differentially expressed genes through bootstrapping

```
library(limma);
library(Biobase);

#-------------------------------------------------------------------------------
# Returns the intersection of several lists
#-------------------------------------------------------------------------------
intersectOfLists = function(lst, coverage=1.0)
{
  n = length(lst);
  t = table(unlist(lst));
  idx = (t >= coverage*n);

  return(dimnames(t[idx])[[1]]);
}

#-------------------------------------------------------------------------------
# apply limma and return gene lists for each permutation
#-------------------------------------------------------------------------------
applyLimma = function(eset,
                      nbrPerm = 100,
                      filter_fc = 2.0,
                      filter_pvalue = 0.05)
{
  incl = 0.9;
  lsts = list();
  for(i in 1:nbrPerm)
  {
    #-- Take random incl% of samples
    idx = sample(1:ncol(eset), ncol(eset)*incl);

    x = exprs(eset)[,idx];
    y = pData(eset)[idx,"Disease"];

    design = cbind(Grp1=1,Grp2vs1=y);
    fitted = lmFit(x,design);

    res = eBayes(fitted);
    res = topTable(res, coef=2, number=dim(x)[1], adjust.method="BH",confint=FALSE);
    idx = abs(res[,"logFC"]) > filter_fc;
    res = res[idx,];
    idx = abs(res[,"adj.P.Val"]) < filter_pvalue;
    res = res[idx,];
    res = res[order(abs(res[,"adj.P.Val"]),decreasing=FALSE),];

    lsts[[i]] = res[,"ID"];
  }

  lst = intersectOfLists(lsts);
  return(lst);
}
```

B

Complete List of Publications

Below a complete list of all publications obtained during this dissertation. Note that some of the topics or content are not covered in this final manuscript due to space limitations or lack of relation with the rest of my work. This list however gives a good indication of the different paths I travelled in order to finally create this dissertation.

2012

J. Taminau, S. Meganck, C. Lazar, D. Steenhoff, A. Coletta, C. Molter, R. Duque, V. de Schaetzen, D. Y. Weiss Solís, H. Bersini and A. Nowé, "Unlocking the potential of publicly available microarray data using inSilicoDb and inSilicoMerging R/Bioconductor packages," *BMC Bioinformatics*, Submitted.

Abstract: With an abundant amount of microarray gene expression data sets available through public repositories, new possibilities lie in combining multiple existing data sets. In this new context, analysis itself is no longer the problem, but retrieving and consistently integrating all this data before delivering it to the wide

variety of existing analysis tools becomes the new bottleneck. We present the newly released inSilicoMerging R/Bioconductor package which, together with the earlier released inSilicoDb R/Bioconductor package, allows consistent retrieval, integration and analysis of publicly available microarray gene expression data sets. Inside the inSilicoMerging package a set of five visual and six quantitative validation measures are available as well. By providing (i) access to uniformly curated and preprocessed data, (ii) a collection of techniques to remove the batch effects between data sets from different sources, and (iii) several validation tools enabling the inspection of the integration process, researchers are now able to fully explore the potential of combining gene expression data for downstream analysis. The power of using both packages is demonstrated by programmatically retrieving and integrating gene expression studies from the InSilico DB repository [https://insilico.ulb.ac.be/app].

A. Coletta, C. Molter, R. Duque, D. Steenhoff, **J. Taminau**, V. de Schaetzen, S. Meganck, D. Venet, C. Lazar, V. Detours, A. Nowé, H. Bersini and D. Y. Weiss Solís, "InSilico DB genomic datasets hub: an efficient starting point for analysing genomewide studies in GenePattern, Integrative Genomics Viewer, and R/Bioconductor," *Genome Biology*, Accepted.

Abstract: Genomics datasets are increasingly useful for gaining biomedical insights, with adoption in the clinic underway. However, multiple hurdles related to data management stand in the way of their efficient large-scale utilization. The solution proposed is a web-based data storage hub: InSilico DB. Having clear focus, flexibility and adaptability, InSilico DB seamlessly connects genomics datasets repositories to state-of-the-art and free GUI and command-line data analysis tools. The InSilico DB platform is a powerful collaborative environment, with advanced capabilities for biocuration, datasets sharing, and datasets subsetting and combination.

Y. Pérez-Castillo, C. Lazar, **J. Taminau**, M. Froeyen, M. Á. Cabrera-Pérez and A. Nowé, "GA(M)E-QSAR: A novel, fully automatic Genetic-Algorithm-(Meta)-Ensembles approach for binary classification in ligand-based drug design," *J Chem Inf Comput Sci*, Accepted.

Abstract: Computer-aided drug design has become an important component of the drug discovery process. Despite the advances in this field, there is not a unique modeling approach than can be successfully applied to solve the whole range of problems faced during QSAR modeling. Feature selection and ensemble modeling are active areas of research in ligand-based drug design. Here we introduce the GA(M)E-QSAR algorithm that combines the search and optimization capabilities of Genetic Algorithms with the simplicity of the Adaboost ensemble-based classification algorithm to solve binary classification problems. We also explore the

usefulness of Meta-Ensembles trained with Adaboost and Voting schemes to further improve the accuracy, generalization and robustness of the optimal Adaboost Single Ensemble derived from the Genetic Algorithm optimization. We evaluated the performance of our algorithm using five data sets from the literature and found that it is capable to yield similar or better classification results to what has been reported for these data sets with a higher enrichment of active compounds relative to the whole actives subset when only the most active chemicals are considered. More important, we compared our methodology with state of the art feature selection and classification approaches and found that it can provide highly accurate, robust and generalizable models. In the case of the Adaboost Ensembles derived from the Genetic Algorithm search, the final models are quite simple since they consist of a weighted sum of the output of single feature classifiers. Furthermore, the Adaboost scores can be used as ranking criterion to prioritize chemicals for synthesis and biological evaluation after virtual screening experiments.

C. Lazar^{+}, S. Meganck^{+}, **J. Taminau**$^{+}$, D. Steenhoff, A. Coletta, C. Molter, D. Y. Weiss Solís, R. Duque, H. Bersini and A. Nowé, "Batch effect removal methods for microarray gene expression data integration: a survey," *Briefings in Bioinformatics*, Accepted.

($^{+}$**Authors with equal contribution**)

Abstract: Genomic data integration is a key goal to be achieved towards large scale genomic data anal- ysis. This process is very challenging due to the diverse sources of information resulting from genomics experiments. In this work we review methods designed to combine genomic data recorded from microarray gene expression (MAGE) experiments. It has been acknowledged that the main source of variation between different MAGE data sets is due to the so-called batch effects. The methods reviewed here perform data integration by removing (or more precisely attempting to remove) the unwanted variation associated with batch effects. They are presented in a unified framework together with a wide range of evaluation tools, which are mandatory in assessing the efficiency and the quality of the data integration process. We provide a systematic description of the MAGE data integration methodology together with some basic recommenda- tion to help the users in choosing the appropriate tools to integrate MAGE data for large scale analysis; and also how to evaluate them from different perspectives in order to quantify their efficiency. All genomic data used in this study for illustration purposes were retrieved from InSilicoDB [http://insilico.ulb.ac.be].

C. Lazar, **J. Taminau**, S. Meganck, D. Steenhoff, A. Coletta, C. Molter, V. de Schaetzen, R. Duque, H. Bersini, and A. Nowé, "A Survey on Filter Techniques for Feature Selection in Gene Expression Microarray Analysis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 99, iss. 4, pp. 1106-1119, 2012.

Abstract: a plenitude of feature selection (FS) methods is available in the literature, most of them rising as a need to analyse datasets of very high dimension, usually hundreds or thousands of variables. Such datasets are available in various application areas such as combinatorial chemistry, text mining, hyperspectral imaging or bioinformatics. As a general accepted rule, these methods are grouped in three generic categories: filter, wrapper and embedded techniques. The focus in this survey is on filter feature selection methods for informative feature discovery in microarray analysis which is also known as differentially expressed genes (DEGs) discovery, gene prioritization or biomarker discovery. We present them in a unified framework, using standardized notations in order to reveal their technical details and to highlight their common characteristics as well as their particularities. This survey ends with a taxonomy proposal on filter feature selection methods for microarrays.

2011

J. Taminau, D. Steenhoff, A. Coletta, S. Meganck, C. Lazar, V. de Schaetzen, R. Duque, C. Molter, H. Bersini, A. Nowé, and D. Y. Weiss Solís, ”InSilicoDb: an R/Bioconductor package for accessing human Affymetrix expert-curated datasets from GEO,” *Bioinformatics*, vol. 27, iss. 22, pp. 3204-3205, 2011.

Abstract: Microarray technology has become an integral part of biomedical research and increasing amounts of datasets become available through public repositories. However, re-use of these datasets is severely hindered by unstructured, missing or incorrect biological samples information; as well as the wide variety of preprocessing methods in use. The inSilicoDb R/Bioconductor package is a command-line front-end to the InSilico DB, a web-based database currently containing 86 104 expert-curated human Affymetrix expression profiles compiled from 1937 GEO repository series. The use of this package builds on the Bioconductor project’s focus on reproducibility by enabling a clear workflow in which not only analysis, but also the retrieval of verified data is supported.

J. Taminau, R. Hillewaere, S. Meganck, D. Conklin, A. Nowé, and B. Manderick, ”Applying Subgroup Discovery for the Analysis of String Quartet Movements,” in Proc. Proceedings of the 23rd Benelux Conference on Artificial Intelligence, Ghent, Belgium, pp. 435-437, 2011.

2010

J. Taminau, S. Meganck, C. Lazar, D. Y. Weiss-Solís, A. Coletta, N. Walker, H. Bersini, and A. Nowé, "Sequential Application of Feature Selection and Extraction for Predicting Breast Cancer Aggressiveness," *Communications in Computer and Information Science*, Springer Berlin Heidelberg, vol. 115, pp. 46-57, 2010.

Abstract: Breast cancer is a heterogenous disease with a large variance in prognosis of patients. It is hard to identify patients who would need adjuvant chemotherapy to survive. Using microarray based technology and various feature selection techniques, a number of prognostic gene expression signatures have been proposed recently. It has been shown that these signatures outperform traditional clinical guidelines for estimating prognosis. This paper studies the applicability of state-of-the-art feature extraction methods together with feature selection methods to develop more powerful prognosis estimators. Feature selection is used to remove features not related with the clinical issue investigated. If the resulted dataset is still described by a high number of probes, feature extraction methods can be applied to further reduce the dimension of the data set. In addition we derived six new signatures using three independent data sets, containing in total 610 samples.

J. Taminau, R. Hillewaere, S. Meganck, D. Conklin, A. Nowé, and B. Manderick, "Applying Subgroup Discovery for the Analysis of String Quartet Movements," in *Proc. Third International Workshop on Machine Learning and Music*, Firenze, Italy, 2010.

Abstract: Descriptive and predictive analyses of symbolic music data assist in understanding the properties that characterize specific genres, movements and composers. Subgroup Discovery, a machine learning technique lying on the intersection between these types of analysis, is applied on a dataset of string quartet movements composed by either Haydn or Mozart. The resulting rules describe subgroups of movements for each composer, which are examined manually, and we investigate whether these subgroups correlate with metadata such as type of movement or period. In addition to this descriptive analysis, the obtained rules are used for the predictive task of composer classification; results are compared with previous results on this corpus.

J. Taminau, S. Meganck, and A. Nowé, "Validation of Merging Techniques for Cancer Microarray Data Sets," in *Proc. The 22nd Benelux Conference on Artificial Intelligence*, Luxembourg, Luxembourg, 2010.

2009

J. Taminau, S. Meganck, D. Y. Weiss-Solís, W. C. G. van Staveren, G. Dom, D. Venet, H. Bersini, V. Detours, and A. Nowé "Validation of Merging Techniques for Cancer Microarray Data Sets," *Australian Journal of Intelligent Information Processing Systems*, vol. 10, iss. 4, pp. 4-11, 2009.

Abstract: There is a vast amount of gene expression data that has been gathered in microarray studies all over the world. Many of these studies use different experimentation plans, different platforms, different methodologies, etc. Merging information of different studies is an important part of current research in bioinformatics and several algorithms have been proposed recently. There is a need to create large data sets which will allow more statistically relevant analysis. In this article we concisely describe several data merging techniques and apply them on cancer microarray data sets. We study three cases of increasing complexity and test all methods by using a number of popular validation criteria. Furthermore, we test the compatibility of the transformed data sets by performing cross-study classification.

Y. Gómez, R. Bello, A. Nowé, E. Casanovas, and **J. Taminau** "Multicolony ACO and Rough Set Theory to Distributed Feature Selection Problem," *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, vol. 5518, pp. 458-46, 2009.

Abstract: In this paper we present a model to distributed feature selection problem (DFSP) based on ACO and RST. The algorithm looks for reducts by using a multi-colony ACO as search method and RST offers the heuristic function to measure the quality of one feature subset. General results of using this approach are shown and formers results of apply ACO and RST to the feature selection problem are referenced.

J. Taminau, R. Hillewaere, S. Meganck, D. Conklin, A. Nowé, and B. Manderick, "Descriptive Subgroup Mining of Folk Music," in Proc. Second International Workshop on Machine Learning and Music, Bled, Slovenia, pp. 1-6, 2009.

Abstract: Descriptive analysis of music corpora is important to musi- cologists who are interested in identifying the properties that characterize specific genres of music. In this study we present such an analysis of a large corpus of folk tunes, all labeled by their origin. Subgroup Discovery (SD) is a rule learning technique located at the intersection of predictive and descriptive induction. One of the advantages of using this technique is the intuitive and interpretable result in the form of a collection of simple rules. Classification accuracy is not the goal of this study. Instead, we discuss some of the highest scoring rules with respect to their descriptive power.

J. Taminau, R. Hillewaere, S. Meganck, D. Conklin, A. Nowé, and B. Manderick ”Descriptive Mining of Folk Music: A Testcase,” in *Proc. The 21st Benelux Conference on Artificial Intelligence*, Eindhoven, The Netherlands, pp. 377-379, 2009.

List of Figures

List of Tables

Bibliography

[Affymetrix (2002)] AFFYMETRIX (2002). Statistical algorithms description document. Tech. rep., Santa Clara, CA, USA. [Cited on pages 15, 23, 142, and 163].

[Al-Shahrour et al. (2007)] AL-SHAHROUR, F., MINGUEZ, P., TÁRRAGA, J., MEDINA, I., ALLOZA, E., MONTANER, D. & DOPAZO, J. (2007). FatiGO +: a functional profiling tool for genomic data. Integration of functional annotation, regulatory motifs and interaction data with microarray experiments. *Nucleic Acids Res* **35**(Web Server issue), W91–6. [Cited on page 115].

[Allison et al. (2006)] ALLISON, D. B., CUI, X., PAGE, G. P. & SABRIPOUR, M. (2006). Microarray data analysis: from disarray to consolidation and consensus. *Nat Rev Genet* **7**(1), 55–65. [Cited on pages 26, 27, and 163].

[Alter et al. (2000)] ALTER, O., BROWN, P. O. & BOTSTEIN, D. (2000). Singular value decomposition for genome-wide expression data processing and modeling. *Proc Natl Acad Sci USA* **97**(18), 10101–6. [Cited on page 102].

[Alwine et al. (1977)] ALWINE, J. C., KEMP, D. J. & STARK, G. R. (1977). Method for detection of specific RNAs in agarose gels by transfer to diazobenzyloxymethyl-paper and hybridization with DNA probes. *Proc Natl Acad Sci USA* **74**(12), 5350–4. [Cited on page 13].

[Ashburner et al. (2000)] ASHBURNER, M., BALL, C. A., BLAKE, J. A., BOTSTEIN, D., BUTLER, H., CHERRY, J. M., DAVIS, A. P., DOLIN-

SKI, K., DWIGHT, S. S., EPPIG, J. T. et al. (2000). Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet* **25**(1), 25–9. [Cited on pages 30 and 80].

[Autio et al. (2009)] AUTIO, R., KILPINEN, S., SAARELA, M., KALLIONIEMI, O., HAUTANIEMI, S. & ASTOLA, J. (2009). Comparison of Affymetrix data normalization methods using 6,926 experiments across five array generations. *BMC Bioinformatics* **10 Suppl 1**, S24. [Cited on page 76].

[Baggerly & Coombes (2009)] BAGGERLY, K. A. & COOMBES, K. R. (2009). Deriving chemosensitivity from cell lines: forensic bioinformatics and reproducible research in high-throughput biology. *The Annals of Applied Statistics* **3**(4), 1309–34. [Cited on page 36].

[Baggerly & Coombes (2011)] BAGGERLY, K. A. & COOMBES, K. R. (2011). What Information Should Be Required to Support Clinical "Omics" Publications? *Clin Chem* **57**(5), 688–690. [Cited on page 37].

[Bär et al. (2009)] BÄR, M., BÄR, D. & LEHMANN, B. (2009). Selection and validation of candidate housekeeping genes for studies of human keratinocytes–review and recommendations. *J Invest Dermatol* **129**(3), 535–7. [Cited on page 76].

[Barrett et al. (2011)] BARRETT, T., TROUP, D. B., WILHITE, S. E., LEDOUX, P., EVANGELISTA, C., KIM, I. F., TOMASHEVSKY, M., MARSHALL, K. A., PHILLIPPY, K. H., SHERMAN, P. M. et al. (2011). NCBI GEO: archive for functional genomics data sets–10 years on. *Nucleic Acids Res* **39**(Database issue), D1005–10. [Cited on page 35].

[Benito et al. (2004)] BENITO, M., PARKER, J., DU, Q., WU, J., XIANG, D., PEROU, C. M. & MARRON, J. S. (2004). Adjustment of systematic microarray data biases. *Bioinformatics* **20**(1), 105–14. [Cited on pages 21, 101, and 126].

[Birol et al. (2009)] BIROL, I., JACKMAN, S. D., NIELSEN, C. B., QIAN, J. Q., VARHOL, R., STAZYK, G., MORIN, R. D., ZHAO, Y., HIRST, M., SCHEIN, J. E. et al. (2009). De novo transcriptome assembly with ABySS. *Bioinformatics* **25**(21), 2872–7. [Cited on page 17].

[Bodenreider (2004)] BODENREIDER, O. (2004). The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Res* **32**(Database issue), D267–70. [Cited on page 38].

[Bolstad et al. (2003)] BOLSTAD, B. M., IRIZARRY, R. A., ASTRAND, M. & SPEED, T. P. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics* **19**(2), 185–93. [Cited on pages 22 and 105].

[Brazma (2009)] BRAZMA, A. (2009). Minimum Information About a Microarray Experiment (MIAME)–successes, failures, challenges. *ScientificWorldJournal* **9**, 420–3. [Cited on page 38].

[Brazma et al. (2001)] BRAZMA, A., HINGAMP, P., QUACKENBUSH, J., SHERLOCK, G., SPELLMAN, P., STOECKERT, C., AACH, J., ANSORGE, W., BALL, C. A., CAUSTON, H. C. et al. (2001). Minimum information about a microarray experiment (MIAME)-toward standards for microarray data. *Nat Genet* **29**(4), 365–71. [Cited on page 38].

[Brettschneider et al. (2008)] BRETTSCHNEIDER, J., COLLIN, F., BOLSTAD, B. M. & SPEED, T. P. (2008). Quality Assessment for Short Oligonucleotide Microarray Data. [Cited on page 110].

[Brown & Botstein (1999)] BROWN, P. O. & BOTSTEIN, D. (1999). Exploring the new world of the genome with DNA microarrays. *Nat Genet* **21**(1 Suppl), 33–7. [Cited on page 14].

[Brunet et al. (2004)] BRUNET, J.-P., TAMAYO, P., GOLUB, T. R. & MESIROV, J. P. (2004). Metagenes and molecular pattern discovery using matrix factorization. *Proc Natl Acad Sci USA* **101**(12), 4164–9. [Cited on page 29].

[Butte & Chen (2006)] BUTTE, A. J. & CHEN, R. (2006). Finding disease-related genomic experiments within an international repository: first steps in translational bioinformatics. *AMIA Annu Symp Proc* , 106–10. [Cited on page 38].

[Butte et al. (2001)] BUTTE, A. J., DZAU, V. J. & GLUECK, S. B. (2001). Further defining housekeeping, or "maintenance," genes Focus on "A compendium of gene expression in normal human tissues". *Physiol Genomics* **7**(2), 95–6. [Cited on page 77].

[Chen et al. (2011)] CHEN, C., GRENNAN, K., BADNER, J., ZHANG, D., GERSHON, E., JIN, L. & LIU, C. (2011). Removing batch effects in analysis of expression microarray data: an evaluation of six batch adjustment methods. *PLoS ONE* **6**(2), e17238. [Cited on page 91].

[Cheng et al. (2011)] CHENG, W.-C., CHANG, C.-W., CHEN, C.-R., TSAI, M.-L., SHU, W.-Y., LI, C.-Y. & HSU, I. C. (2011). Identification of reference genes across physiological states for qRT-PCR through microarray meta-analysis. *PLoS ONE* **6**(2), e17347. [Cited on pages 77 and 82].

[Choi et al. (2003)] CHOI, J. K., YU, U., KIM, S. & YOO, O. J. (2003). Combining multiple microarray studies and modeling interstudy variation. *Bioinformatics* **19 Suppl 1**, i84–90. [Cited on pages 73, 74, and 75].

[Coletta et al. (2012)] COLETTA, A., MOLTER, C., DUQUÉ, R., STEENHOFF, D., TAMINAU, J., DE SCHAETZEN, V., MEGANCK, S., VENET, D., LAZAR, C., DETOURS, V., NOWÉ, A., BERSINI, H. & SOLÍS, D. Y. W. (2012). InSilico DB genomic datasets hub: an efficient starting point for analysing genomewide studies in GenePattern, Integrative Genomics Viewer, and R/Bioconductor. *Genome Biology* . [Cited on pages 6, 39, and 41].

[Cope et al. (2007)] COPE, L. M., GARRETT-MAYER, L., GABRIELSON, E. & PARMIGIANI, G. (2007). The Integrative Correlation Coefficient : a Measure of Cross-study Reproducibility for Gene Expressionea

Array Data. *Johns Hopkins University, Dept. of Biostatistics Working Paper Series* (152). [Cited on page 115].

[Cox & Cox (2001)] COX, T. & COX, M. (2001). *Multidimensional Scaling*. Chapman & Hall CRC. [Cited on page 31].

[Crick (1970)] CRICK, F. (1970). Central dogma of molecular biology. *Nature* **227**(5258), 561–3. [Cited on page 11].

[Cristianini & Shawe-Taylor (2000)] CRISTIANINI, N. & SHAWE-TAYLOR, J. (2000). *An introduction to Support Vector Machines: and other kernel-based learning methods*. Cambridge University Press. [Cited on page 101].

[Dai et al. (2005)] DAI, M., WANG, P., BOYD, A. D., KOSTOV, G., ATHEY, B., JONES, E. G., BUNNEY, W. E., MYERS, R. M., SPEED, T. P., AKIL, H. et al. (2005). Evolving gene/transcript definitions significantly alter the interpretation of GeneChip data. *Nucleic Acids Res* **33**(20), e175. [Cited on page 142].

[Davis & Meltzer (2007)] DAVIS, S. & MELTZER, P. S. (2007). GEOquery: a bridge between the Gene Expression Omnibus (GEO) and BioConductor. *Bioinformatics* **23**(14), 1846–7. [Cited on page 58].

[de Jonge et al. (2007)] DE JONGE, H. J. M., FEHRMANN, R. S. N., DE BONT, E. S. J. M., HOFSTRA, R. M. W., GERBENS, F., KAMPS, W. A., DE VRIES, E. G. E., VAN DER ZEE, A. G. J., TE MEERMAN, G. J. & TER ELST, A. (2007). Evidence based selection of housekeeping genes. *PLoS ONE* **2**(9), e898. [Cited on pages 77, 79, 81, and 82].

[DeConde et al. (2006)] DECONDE, R. P., HAWLEY, S., FALCON, S., CLEGG, N., KNUDSEN, B. & ETZIONI, R. (2006). Combining results of microarray experiments: a rank aggregation approach. *Stat Appl Genet Mol Biol* **5**, Article15. [Cited on page 75].

[DePristo et al. (2011)] DEPRISTO, M. A., BANKS, E., POPLIN, R., GARIMELLA, K. V., MAGUIRE, J. R., HARTL, C., PHILIPPAKIS,

A. A., del Angel, G., Rivas, M. A., Hanna, M. et al. (2011). A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat Genet* **43**(5), 491–8. [Cited on page 54].

[Draghici et al. (2006)] Draghici, S., Khatri, P., Eklund, A. C. & Szallasi, Z. (2006). Reliability and reproducibility issues in DNA microarray measurements. *Trends Genet* **22**(2), 101–9. [Cited on page 20].

[Edgar et al. (2002)] Edgar, R., Domrachev, M. & Lash, A. E. (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res* **30**(1), 207–10. [Cited on pages 35 and 53].

[Efron & Tibshirani (1993)] Efron, B. & Tibshirani, R. (1993). *An Introduction to the Bootstrap*. Chapman & Hall/CRC.

[Ein-Dor et al. (2005)] Ein-Dor, L., Kela, I., Getz, G., Givol, D. & Domany, E. (2005). Outcome signature genes in breast cancer: is there a unique set? *Bioinformatics* **21**(2), 171–8. [Cited on pages 21 and 66].

[Ein-Dor et al. (2006)] Ein-Dor, L., Zuk, O. & Domany, E. (2006). Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *Proc Natl Acad Sci USA* **103**(15), 5923–8. [Cited on pages 20 and 66].

[Eisen et al. (1998)] Eisen, M. B., Spellman, P. T., Brown, P. O. & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci USA* **95**(25), 14863–8. [Cited on page 31].

[Elo et al. (2005)] Elo, L. L., Lahti, L., Skottman, H., Kyläniemi, M., Lahesmaa, R. & Aittokallio, T. (2005). Integrating probe-level expression changes across generations of Affymetrix arrays. *Nucleic Acids Res* **33**(22), e193. [Cited on page 74].

[Feng et al. (2001)] FENG, G., XU, X., YOUSSEF, E. M. & LOTAN, R. (2001). Diminished expression of S100A2, a putative tumor suppressor, at early stage of human lung carcinogenesis. *Cancer Res* **61**(21), 7999–8004. [Cited on page 133].

[Friedman (2001)] FRIEDMAN, L. (2001). Why vote-count reviews don't count. *Biol Psychiatry* **49**, 161–2. [Cited on page 74].

[Gagnon-Bartsch & Speed (2011)] GAGNON-BARTSCH, J. A. & SPEED, T. P. (2011). Using control genes to correct for unwanted variation in microarray data. *Biostatistics* . [Cited on pages 91, 103, 110, and 114].

[Gene Ontology Consortium (2008)] GENE ONTOLOGY CONSORTIUM (2008). The Gene Ontology project in 2008. *Nucleic Acids Res* **36**(Database issue), D440–4. [Cited on page 80].

[Gentleman et al. (2004)] GENTLEMAN, R. C., CAREY, V. J., BATES, D. M., BOLSTAD, B., DETTLING, M., DUDOIT, S., ELLIS, B., GAUTIER, L., GE, Y., GENTRY, J. et al. (2004). Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol* **5**(10), R80. [Cited on pages 2, 21, 26, 47, 57, 105, and 116].

[Gentleman et al. (2005)] GENTLEMAN, R. C., CAREY, V. J., HUBER, W., IRIZARRY, R. A. & DUTOIT, S. (eds.) (2005). *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Springer. [Cited on pages 24 and 36].

[Goecks et al. (2010)] GOECKS, J., NEKRUTENKO, A., TAYLOR, J. & TEAM, G. (2010). Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol* **11**(8), R86. [Cited on page 26].

[Gómez et al. (2009)] GÓMEZ, Y., BELLO, R., NOWÉ, A., CASANOVAS, E. & TAMINAU, J. (2009). Multi-colony ACO and Rough Set Theory to Distributed Feature Selection Problem. In: *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*, vol. 5518 of *Lecture Notes in Computer Science*. pp. 458–61. [Cited on page 9].

[Grützmann et al. (2005)] GRÜTZMANN, R., BORISS, H., AMMERPOHL, O., LÜTTGES, J., KALTHOFF, H., SCHACKERT, H. K., KLÖPPEL, G., SAEGER, H. D. & PILARSKY, C. (2005). Meta-analysis of microarray data on pancreatic cancer defines a set of commonly dysregulated genes. *Oncogene* **24**(32), 5079–88. [Cited on page 74].

[Guttman et al. (2010)] GUTTMAN, M., GARBER, M., LEVIN, J. Z., DONAGHEY, J., ROBINSON, J., ADICONIS, X., FAN, L., KOZIOL, M. J., GNIRKE, A., NUSBAUM, C., RINN, J. L., LANDER, E. S. & REGEV, A. (2010). Ab initio reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs. *Nat Biotechnol* **28**(5), 503–10. [Cited on page 17].

[Haas & Zody (2010)] HAAS, B. J. & ZODY, M. C. (2010). Advancing RNA-Seq analysis. *Nat Biotechnol* **28**(5), 421–3. [Cited on page 17].

[Harr & Schlötterer (2006)] HARR, B. & SCHLÖTTERER, C. (2006). Comparison of algorithms for the analysis of Affymetrix microarray data as evaluated by co-expression of genes in known operons. *Nucleic Acids Res* **34**(2), e8. [Cited on page 22].

[Hayat et al. (2007)] HAYAT, M. J., HOWLADER, N., REICHMAN, M. E. & EDWARDS, B. K. (2007). Cancer statistics, trends, and multiple primary cancer analyses from the Surveillance, Epidemiology, and End Results (SEER) Program. *Oncologist* **12**(1), 20–37. [Cited on page 121].

[Hedges & Olkin (1985)] HEDGES, L. V. & OLKIN, I. (1985). *Statistical Methods for Meta-Analysis*. Academic Press. [Cited on pages 72, 73, and 74].

[Heid et al. (1996)] HEID, C. A., STEVENS, J., LIVAK, K. J. & WILLIAMS, P. M. (1996). Real time quantitative PCR. *Genome Res* **6**(10), 986–94. [Cited on page 13].

[Hong & Breitling (2008)] HONG, F. & BREITLING, R. (2008). A comparison of meta-analysis methods for detecting differentially expressed genes in microarray experiments. *Bioinformatics* **24**(3), 374–82. [Cited on pages 74 and 75].

[Hong et al. (2006)] HONG, F., BREITLING, R., MCENTEE, C. W., WITTNER, B. S., NEMHAUSER, J. L. & CHORY, J. (2006). RankProd: a bioconductor package for detecting differentially expressed genes in meta-analysis. *Bioinformatics* **22**(22), 2825–7. [Cited on page 75].

[Hou et al. (2010)] HOU, J., AERTS, J., DEN HAMER, B., VAN IJCKEN, W., DEN BAKKER, M., RIEGMAN, P., VAN DER LEEST, C., VAN DER SPEK, P., FOEKENS, J. A., HOOGSTEDEN, H. C. et al. (2010). Gene expression-based classification of non-small cell lung carcinomas and survival prediction. *PLoS ONE* **5**(4), e10312. [Cited on page 123].

[Huang et al. (2012)] HUANG, H., LU, X., LIU, Y., HAALAND, P. & MARRON, J. S. (2012). R/DWD: distance-weighted discrimination for classification, visualization and batch adjustment. *Bioinformatics* **28**(8), 1182–3. [Cited on pages 116 and 118].

[Huber et al. (2002)] HUBER, W., VON HEYDEBRECK, A., SÜLTMANN, H., POUSTKA, A. & VINGRON, M. (2002). Variance stabilization applied to microarray data calibration and to the quantification of differential expression. *Bioinformatics* **18 Suppl 1**, S96–104. [Cited on page 23].

[Hull et al. (2006)] HULL, D., WOLSTENCROFT, K., STEVENS, R., GOBLE, C., POCOCK, M. R., LI, P. & OINN, T. (2006). Taverna: a tool for building and running workflows of services. *Nucleic Acids Res* **34**(Web Server issue), W729–32. [Cited on page 54].

[Ioannidis (2005)] IOANNIDIS, J. P. A. (2005). Microarrays and molecular research: noise discovery? *Lancet* **365**(9458), 454–5. [Cited on pages 19 and 21].

[Irizarry et al. (2003)] IRIZARRY, R. A., BOLSTAD, B. M., COLLIN, F., COPE, L. M., HOBBS, B. & SPEED, T. P. (2003). Summaries of Affymetrix GeneChip probe level data. *Nucleic Acids Res* **31**(4), e15. [Cited on pages 23 and 105].

[Jiang et al. (2004)] JIANG, H., DENG, Y., CHEN, H.-S., TAO, L., SHA, Q., CHEN, J., TSAI, C.-J. & ZHANG, S. (2004). Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes. *BMC Bioinformatics* **5**, 81. [Cited on pages 105 and 128].

[Johnson et al. (2007)] JOHNSON, W. E., LI, C. & RABINOVIC, A. (2007). Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics* **8**(1), 118–27. [Cited on pages 99 and 126].

[Jolliffe (2002)] JOLLIFFE, I. T. (2002). *Principal Component Analysis*. Springer, 2nd ed. [Cited on page 102].

[Kanehisa & Goto (2000)] KANEHISA, M. & GOTO, S. (2000). KEGG: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res* **28**(1), 27–30. [Cited on pages 30 and 143].

[Kerr et al. (2000)] KERR, M. K., MARTIN, M. & CHURCHILL, G. A. (2000). Analysis of variance for gene expression microarray data. *J Comput Biol* **7**(6), 819–37. [Cited on page 19].

[Kim et al. (2011)] KIM, D. S., LEE, J. Y., LEE, S. M., CHOI, J. E., CHO, S. & PARK, J. Y. (2011). Promoter methylation of the RGC32 gene in nonsmall cell lung cancer. *Cancer* **117**(3), 590–6. [Cited on page 133].

[Kim et al. (2007)] KIM, K.-Y., KIM, S. H., KI, D. H., JEONG, J., JEONG, H. J., JEUNG, H.-C., CHUNG, H. C. & RHA, S. Y. (2007). An attempt for combining microarray data sets by adjusting gene expressions. *Cancer Res Treat* **39**(2), 74–81. [Cited on pages 99, 107, 108, 109, and 115].

[Kitchen et al. (2011)] KITCHEN, R. R., SABINE, V. S., SIMEN, A. A., DIXON, J. M., BARTLETT, J. M. S. & SIMS, A. H. (2011). Relative impact of key sources of systematic noise in Affymetrix and Illumina gene-expression microarray experiments. *BMC Genomics* **12**, 589. [Cited on page 20].

[Knight et al. (2009)] KNIGHT, L. J., BURRAGE, J., BUJAC, S. R., HAGGERTY, C., GRAHAM, A., GIBSON, N. J., ELLISON, G., GROWCOTT, J. W., BROOKS, A. N., HUGHES, A. M. et al. (2009). Epigenetic silencing of the endothelin-B receptor gene in non-small cell lung cancer. *Int J Oncol* **34**(2), 465–71. [Cited on page 128].

[Lacson et al. (2010)] LACSON, R., PITZER, E., KIM, J., GALANTE, P., HINSKE, C. & OHNO-MACHADO, L. (2010). DSGeo: software tools for cross-platform analysis of gene expression data in GEO. *J Biomed Inform* **43**(5), 709–15. [Cited on page 105].

[Landi et al. (2008)] LANDI, M. T., DRACHEVA, T., ROTUNNO, M., FIGUEROA, J. D., LIU, H., DASGUPTA, A., MANN, F. E., FUKUOKA, J., HAMES, M., BERGEN, A. W. et al. (2008). Gene expression signature of cigarette smoking and its role in lung adenocarcinoma development and survival. *PLoS ONE* **3**(2), e1651. [Cited on page 123].

[Larsson & Sandberg (2006)] LARSSON, O. & SANDBERG, R. (2006). Lack of correct data format and comparability limits future integrative microarray research. *Nat Biotechnol* **24**(11), 1322–3. [Cited on pages 37 and 63].

[Larsson et al. (2006)] LARSSON, O., WENNMALM, K. & SANDBERG, R. (2006). Comparative microarray analysis. *OMICS* **10**(3), 381–97. [Cited on pages 65 and 66].

[Lazar et al. (2012)a] LAZAR, C., MEGANCK, S., TAMINAU, J., STEENHOFF, D., COLETTA, A., MOLTER, C., WEISS SOLÍS, D. Y., DUQUE, R., BERSINI, H. & NOWÉ, A. (2012a). Batch effect removal methods for microarray gene expression data integration: a survey. *Briefings in Bioinformatics* . [Cited on pages 7, 91, 93, 94, 97, 106, 111, and 166].

[Lazar et al. (2012)b] LAZAR, C., TAMINAU, J., MEGANCK, S., STEENHOFF, D., COLETTA, A., MOLTER, C., DE SCHAETZEN, V., DUQUE, R., BERSINI, H. & NOWÉ, A. (2012b). A Survey on Filter Techniques for Feature Selection in Gene Expression Microarray Analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **9**(4), 1106–19. [Cited on pages 8, 27, 28, 124, and 163].

[Lee & Verleyen (2007)] LEE, J. A. & VERLEYEN, M. (2007). *Nonlinear Dimensionality Reduction*. Springer. [Cited on page 31].

[Lee et al. (2002)] LEE, P. D., SLADEK, R., GREENWOOD, C. M. T. & HUDSON, T. J. (2002). Control genes and variability: absence of ubiquitous reference transcripts in diverse mammalian expression studies. *Genome Res* **12**(2), 292–7. [Cited on page 82].

[Lee et al. (2007)] LEE, S., JO, M., LEE, J., KOH, S. S. & KIM, S. (2007). Identification of novel universal housekeeping genes by statistical analysis of microarray data. *J Biochem Mol Biol* **40**(2), 226–31. [Cited on page 77].

[Leek et al. (2010)] LEEK, J. T., SCHARPF, R. B., BRAVO, H. C., SIMCHA, D., LANGMEAD, B., JOHNSON, W. E., GEMAN, D., BAGGERLY, K. & IRIZARRY, R. A. (2010). Tackling the widespread and critical impact of batch effects in high-throughput data. *Nat Rev Genet* **11**(10), 733–9. [Cited on pages 4, 24, 69, 89, 91, 101, 107, and 109].

[Leek & Storey (2007)] LEEK, J. T. & STOREY, J. D. (2007). Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genet* **3**(9), 1724–35. [Cited on pages 103 and 114].

[Li & Wong (2001)] LI, C. & WONG, W. H. (2001). Model-based analysis of oligonucleotide arrays: expression index computation and outlier detection. *Proc Natl Acad Sci USA* **98**(1), 31–6. [Cited on page 98].

[Lindsay (2003)] LINDSAY, M. A. (2003). Target discovery. *Nat Rev Drug Discov* **2**(10), 831–8. [Cited on page 13].

[Lockhart et al. (1996)] LOCKHART, D. J., DONG, H., BYRNE, M. C., FOLLETTIE, M. T., GALLO, M. V., CHEE, M. S., MITTMANN, M., WANG, C., KOBAYASHI, M., HORTON, H. & BROWN, E. L. (1996). Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nat Biotechnol* **14**(13), 1675–80. [Cited on page 14].

[Lu et al. (2010)] LU, T.-P., TSAI, M.-H., LEE, J.-M., HSU, C.-P., CHEN, P.-C., LIN, C.-W., SHIH, J.-Y., YANG, P.-C., HSIAO, C. K., LAI, L.-C. & CHUANG, E. Y. (2010). Identification of a novel biomarker, SEMA5A, for non-small cell lung carcinoma in nonsmoking women. *Cancer Epidemiol Biomarkers Prev* **19**(10), 2590–7. [Cited on page 123].

[Lukk et al. (2010)] LUKK, M., KAPUSHESKY, M., NIKKILÄ, J., PARKINSON, H., GONCALVES, A., HUBER, W., UKKONEN, E. & BRAZMA, A. (2010). A global map of human gene expression. *Nat Biotechnol* **28**(4), 322–4. [Cited on pages 86, 87, 88, and 165].

[Luo et al. (2010)] LUO, J., SCHUMACHER, M., SCHERER, A., SANOUDOU, D., MEGHERBI, D., DAVISON, T., SHI, T., TONG, W., SHI, L., HONG, H. et al. (2010). A comparison of batch effect removal methods for enhancement of prediction performance using MAQC-II microarray gene expression data. *Pharmacogenomics J* **10**(4), 278–91. [Cited on pages 91, 92, and 111].

[Malone & Oliver (2011)] MALONE, J. H. & OLIVER, B. (2011). Microarrays, deep sequencing and the true measure of the transcriptome. *BMC Biol* **9**, 34. [Cited on pages 14 and 17].

[Marshall (2004)] MARSHALL, E. (2004). Getting the noise out of gene arrays. *Science* **306**(5696), 630–1. [Cited on page 20].

[McCall et al. (2010)] MCCALL, M. N., BOLSTAD, B. M. & IRIZARRY, R. A. (2010). Frozen robust multiarray analysis (fRMA). *Biostatistics* **11**(2), 242–53. [Cited on pages 23, 24, 53, and 104].

[McCall & Irizarry (2011)] McCall, M. N. & Irizarry, R. A. (2011). Thawing Frozen Robust Multi-array Analysis (fRMA). *BMC Bioinformatics* **12**, 369. [Cited on page 104].

[McCall et al. (2011)] McCall, M. N., Uppal, K., Jaffee, H. A., Zilliox, M. J. & Irizarry, R. A. (2011). The Gene Expression Barcode: leveraging public data repositories to begin cataloging the human and murine transcriptomes. *Nucleic Acids Res* **39**(Database issue), D1011–5. [Cited on page 104].

[Mesirov (2010)] Mesirov, J. P. (2010). Computer science. Accessible reproducible research. *Science* **327**(5964), 415–6. [Cited on page 36].

[Metzker (2010)] Metzker, M. L. (2010). Sequencing technologies - the next generation. *Nat Rev Genet* **11**(1), 31–46. [Cited on page 16].

[Michiels et al. (2005)] Michiels, S., Koscielny, S. & Hill, C. (2005). Prediction of cancer outcome with microarrays: a multiple random validation strategy. *Lancet* **365**(9458), 488–92. [Cited on pages 20 and 66].

[Mitchell (1997)] Mitchell, T. M. (1997). *Machine Learning*. The McGraw-Hill Companies, Inc. [Cited on page 25].

[Moreau et al. (2003)] Moreau, Y., Aerts, S., Moor, B. D., Strooper, B. D. & Dabrowski, M. (2003). Comparison and meta-analysis of microarray data: from the bench to the computer desk. *Trends Genet* **19**(10), 570–7. [Cited on page 63].

[Normand (1999)] Normand, S. L. (1999). Tutorial in biostatistics-meta-analysis: formulating, evaluating, combining and reporting. *Stat Med* **18**, 321–59. [Cited on page 73].

[Novoradovskaya et al. (2004)] Novoradovskaya, N., Whitfield, M. L., Basehore, L. S., Novoradovsky, A., Pesich, R., Usary, J., Karaca, M., Wong, W. K., Aprelikova, O., Fero, M. et al. (2004). Universal Reference RNA as a standard for microarray experiments. *BMC Genomics* **5**(1), 20. [Cited on page 98].

[Nueda et al. (2011)] NUEDA, M. J., FERRER, A. & CONESA, A. (2011). ARSyN: a method for the identification and removal of systematic noise in multifactorial time course microarray experiments. *Biostatistics* . [Cited on page 114].

[Parkinson et al. (2011)] PARKINSON, H., SARKANS, U., KOLESNIKOV, N., ABEYGUNAWARDENA, N., BURDETT, T., DYLAG, M., EMAM, I., FARNE, A., HASTINGS, E., HOLLOWAY, E. et al. (2011). ArrayExpress update–an archive of microarray and high-throughput sequencing-based functional genomics experiments. *Nucleic Acids Res* **39**(Database issue), D1002–4. [Cited on page 35].

[Parzen (1962)] PARZEN, E. (1962). On Estimation of a Probability Density Function and Mode. *The Ann of Math Stat* **33**(3), 1065–76. [Cited on pages 108 and 113].

[Pennisi (2003)] PENNISI, E. (2003). Human genome. A low number wins the GeneSweep Pool. *Science* **300**(5625), 1484. [Cited on page 13].

[Pérez-Castillo et al. (2012)] PÉREZ-CASTILLO, Y., LAZAR, C., TAMINAU, J., FROEYEN, M., CABRERA-PÉREZ, M. A. & NOWÉ, A. (2012). GA(M)E-QSAR: A novel, fully automatic Genetic-Algorithm-(Meta)-Ensembles approach for binary classification in ligand-based drug design. *J Chem Inf Comput Sci* . [Cited on page 9].

[Popovici et al. (2009)] POPOVICI, V., GOLDSTEIN, D. R., ANTONOV, J., JAGGI, R., DELORENZI, M. & WIRAPATI, P. (2009). Selecting control genes for RT-QPCR using public microarray data. *BMC Bioinformatics* **10**, 42. [Cited on pages 77 and 81].

[Pranckeviciene & Somorjai (2006)] PRANCKEVICIENE, E. & SOMORJAI, R. (2006). On classification models of gene expression microarrays: the simpler the better. In: *Proceedings of the IEEE International Conference on Neural Networks*. [Cited on page 26].

[Quackenbush et al. (2006)] QUACKENBUSH, J., STOECKERT, C., BALL, C., BRAZMA, A., GENTLEMAN, R., HUBER, W., IRIZARRY, R., SALIT, M., SHERLOCK, G., SPELLMAN, P. & WINEGARDEN, N. (2006). Top-down standards will not serve systems biology. *Nature* **440**(7080), 24. [Cited on page 38].

[R Development Core Team (2005)] R DEVELOPMENT CORE TEAM (2005). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. [Cited on page 2].

[Ramasamy et al. (2008)] RAMASAMY, A., MONDRY, A., HOLMES, C. C. & ALTMAN, D. G. (2008). Key issues in conducting a meta-analysis of gene expression microarray datasets. *PLoS Med* **5**(9), e184. [Cited on pages 67, 68, 72, 76, and 165].

[Reich et al. (2006)] REICH, M., LIEFELD, T., GOULD, J., LERNER, J., TAMAYO, P. & MESIROV, J. P. (2006). GenePattern 2.0. *Nat Genet* **38**(5), 500–1. [Cited on pages 26, 47, and 50].

[Rho et al. (2009)] RHO, J.-H., ROEHRL, M. H. A. & WANG, J. Y. (2009). Glycoproteomic analysis of human lung adenocarcinomas using glycoarrays and tandem mass spectrometry: differential expression and glycosylation patterns of vimentin and fetuin A isoforms. *Protein J* **28**(3-4), 148–60. [Cited on page 128].

[Rhodes et al. (2002)] RHODES, D. R., BARRETTE, T. R., RUBIN, M. A., GHOSH, D. & CHINNAIYAN, A. M. (2002). Meta-analysis of microarrays: interstudy validation of gene expression profiles reveals pathway dysregulation in prostate cancer. *Cancer Res* **62**(15), 4427–33. [Cited on page 72].

[Rhodes et al. (2004)] RHODES, D. R., YU, J., SHANKER, K., DESHPANDE, N., VARAMBALLY, R., GHOSH, D., BARRETTE, T., PANDEY, A. & CHINNAIYAN, A. M. (2004). Large-scale meta-analysis of cancer microarray data identifies common transcriptional profiles of neoplastic transformation and progression. *Proc Natl Acad Sci USA* **101**(25), 9309–14. [Cited on page 74].

[Robinson et al. (2011)] ROBINSON, J. T., THORVALDSDÓTTIR, H., WINCKLER, W., GUTTMAN, M., LANDER, E. S., GETZ, G. & MESIROV, J. P. (2011). Integrative genomics viewer. *Nat Biotechnol* **29**(1), 24–6. [Cited on page 47].

[Rudy & Valafar (2011)] RUDY, J. & VALAFAR, F. (2011). Empirical comparison of cross-platform normalization methods for gene expression data. *BMC Bioinformatics* **12**, 467. [Cited on page 118].

[Saeys et al. (2007)] SAEYS, Y., INZA, I. & LARRAÑAGA, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics* **23**(19), 2507–17. [Cited on page 27].

[Sanchez-Palencia et al. (2011)] SANCHEZ-PALENCIA, A., GOMEZ-MORALES, M., GOMEZ-CAPILLA, J. A., PEDRAZA, V., BOYERO, L., ROSELL, R. & FÁREZ-VIDAL, M. E. (2011). Gene expression profiling reveals novel biomarkers in nonsmall cell lung cancer. *Int J Cancer* **129**(2), 355–64. [Cited on page 123].

[Sandberg & Larsson (2007)] SANDBERG, R. & LARSSON, O. (2007). Improved precision and accuracy for microarrays using updated probe set definitions. *BMC Bioinformatics* **8**, 48. [Cited on page 142].

[Sarmah & Samarasinghe (2010)] SARMAH, C. K. & SAMARASINGHE, S. (2010). Microarray data integration: frameworks and a list of underlying issues. *Curr Bioinf* **5**, 280–9. [Cited on pages 63, 66, and 68].

[Schena et al. (1995)] SCHENA, M., SHALON, D., DAVIS, R. W. & BROWN, P. O. (1995). Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science* **270**(5235), 467–70. [Cited on page 14].

[Scherer (2009)] SCHERER, A. (ed.) (2009). *Batch Effects and Noise in Microarray Experiments: Sources and Solutions*. John Wiley and Sons. [Cited on pages 24, 69, 89, 91, 92, and 93].

[Schulz et al. (2012)] SCHULZ, M. H., ZERBINO, D. R., VINGRON, M. & BIRNEY, E. (2012). Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels. *Bioinformatics* **28**(8), 1086–92. [Cited on page 17].

[Severin et al. (2010)] SEVERIN, J., BEAL, K., VILELLA, A. J., FITZGERALD, S., SCHUSTER, M., GORDON, L., URETA-VIDAL, A., FLICEK, P. & HERRERO, J. (2010). eHive: an artificial intelligence workflow system for genomic analysis. *BMC Bioinformatics* **11**, 240. [Cited on page 54].

[Shabalin et al. (2008)] SHABALIN, A. A., TJELMELAND, H., FAN, C., PEROU, C. M. & NOBEL, A. B. (2008). Merging two gene-expression studies via cross-platform normalization. *Bioinformatics* **24**(9), 1154–60. [Cited on pages 100, 112, 113, 115, and 126].

[She et al. (2009)] SHE, X., ROHL, C. A., CASTLE, J. C., KULKARNI, A. V., JOHNSON, J. M. & CHEN, R. (2009). Definition, conservation and epigenetics of housekeeping and tissue-enriched genes. *BMC Genomics* **10**, 269. [Cited on page 77].

[Sherlock et al. (2001)] SHERLOCK, G., HERNANDEZ-BOUSSARD, T., KASARSKIS, A., BINKLEY, G., MATESE, J. C., DWIGHT, S. S., KALOPER, M., WENG, S., JIN, H., BALL, C. A. et al. (2001). The Stanford Microarray Database. *Nucleic Acids Res* **29**(1), 152–5. [Cited on page 35].

[Shi et al. (2005)] SHI, L., TONG, W., FANG, H., SCHERF, U., HAN, J., PURI, R. K., FRUEH, F. W., GOODSAID, F. M., GUO, L., SU, Z. et al. (2005). Cross-platform comparability of microarray technology: intra-platform consistency and appropriate data analysis procedures are essential. *BMC Bioinformatics* **6 Suppl 2**, S12. [Cited on page 20].

[Sims et al. (2008)] SIMS, A. H., SMETHURST, G. J., HEY, Y., OKONIEWSKI, M. J., PEPPER, S. D., HOWELL, A., MILLER, C. J. & CLARKE, R. B. (2008). 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**, 42. [Cited on pages 97, 114, 115, and 126].

[Smyth (2004)] SMYTH, G. K. (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Stat Appl Genet Mol Biol* **3**, Article3. [Cited on pages 28, 115, and 124].

[Somorjai et al. (2003)] SOMORJAI, R. L., DOLENKO, B. & BAUMGARTNER, R. (2003). Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions. *Bioinformatics* **19**(12), 1484–91. [Cited on pages 21 and 66].

[Spellman et al. (2002)] SPELLMAN, P. T., MILLER, M., STEWART, J., TROUP, C., SARKANS, U., CHERVITZ, S., BERNHART, D., SHERLOCK, G., BALL, C., LEPAGE, M. et al. (2002). Design and implementation of microarray gene expression markup language (MAGE-ML). *Genome Biol* **3**(9), RESEARCH0046. [Cited on page 38].

[Steen (2011)] STEEN, R. G. (2011). Retractions in the medical literature: how many patients are put at risk by flawed research? *Journal of Medical Ethics* **37**(11), 688–692. [Cited on page 37].

[Strieter et al. (1995)] STRIETER, R. M., POLVERINI, P. J., ARENBERG, D. A., WALZ, A., OPDENAKKER, G., DAMME, J. V. & KUNKEL, S. L. (1995). Role of C-X-C chemokines as regulators of angiogenesis in lung cancer. *J Leukoc Biol* **57**(5), 752–62. [Cited on page 133].

[Su et al. (2007)] SU, L.-J., CHANG, C.-W., WU, Y.-C., CHEN, K.-C., LIN, C.-J., LIANG, S.-C., LIN, C.-H., WHANG-PENG, J., HSU, S.-L., CHEN, C.-H. & HUANG, C.-Y. F. (2007). Selection of DDX5 as a novel internal control for Q-RT-PCR from microarray data using a block bootstrap re-sampling scheme. *BMC Genomics* **8**, 140. [Cited on page 123].

[Suárez-Fariñas et al. (2005)] SUÁREZ-FARIÑAS, M., PELLEGRINO, M., WITTKOWSKI, K. M. & MAGNASCO, M. O. (2005). Harshlight:

a "corrective make-up" program for microarray chips. *BMC Bioinformatics* **6**, 294. [Cited on page 92].

[Sunaga et al. (2004)] SUNAGA, N., MIYAJIMA, K., SUZUKI, M., SATO, M., WHITE, M. A., RAMIREZ, R. D., SHAY, J. W., GAZDAR, A. F. & MINNA, J. D. (2004). Different roles for caveolin-1 in the development of non-small cell lung cancer versus small cell lung cancer. *Cancer Res* **64**(12), 4277–85. [Cited on page 128].

[Szabo et al. (2004)] SZABO, A., PEROU, C. M., KARACA, M., PERREARD, L., PALAIS, R., QUACKENBUSH, J. F. & BERNARD, P. S. (2004). Statistical modeling for selecting housekeeper genes. *Genome Biol* **5**(8), R59. [Cited on page 77].

[Tamayo et al. (2011)] TAMAYO, P., STEINHARDT, G., LIBERZON, A. & MESIROV, J. (2011). Gene Set Enrichment Analysis Made Right. *Arxiv preprint arXiv:1110.4128* . [Cited on pages 6 and 61].

[Taminau et al. (2009)a] TAMINAU, J., HILLEWAERE, R., MEGANCK, S., CONKLIN, D., NOWÉ, A. & MANDERICK, B. (2009a). Descriptive Mining of Folk Music: A Testcase. In: *Proceedings of the 21st Benelux Conference on Artificial Intelligence*. Eindhoven, The Netherlands. [Cited on page 8].

[Taminau et al. (2009)b] TAMINAU, J., HILLEWAERE, R., MEGANCK, S., CONKLIN, D., NOWÉ, A. & MANDERICK, B. (2009b). Descriptive Subgroup Mining of Folk Music. In: *Second International Workshop on Machine Learning and Music*. Bled, Slovenia. [Cited on page 8].

[Taminau et al. (2010)a] TAMINAU, J., HILLEWAERE, R., MEGANCK, S., CONKLIN, D., NOWÉ, A. & MANDERICK, B. (2010a). Applying Subgroup Discovery for the Analysis of String Quartet Movements. In: *Third International Workshop on Machine Learning and Music*. Firenze, Italy. [Cited on page 8].

[Taminau et al. (2011)a] TAMINAU, J., HILLEWAERE, R., MEGANCK, S., CONKLIN, D., NOWÉ, A. & MANDERICK, B. (2011a). Applying

Subgroup Discovery for the Analysis of String Quartet Movements. In: *Proceedings of the 23rd Benelux Conference on Artificial Intelligence*. Ghent, Belgium. [Cited on page 8].

[Taminau et al. (Subm)] TAMINAU, J., MEGANCK, S., LAZAR, C., STEENHOFF, D., COLETTA, A., MOLTER, C., DUQUE, R., DE SCHAETZEN, V., WEISS SOLÍS, D. Y., BERSINI, H. & NOWÉ, A. (Subm). Unlocking the potential of publicly available microarray data using inSilicoDb and inSilicoMerging R/Bioconductor packages. *BMC Bioinformatics* . [Cited on pages 7, 86, and 119].

[Taminau et al. (2010)b] TAMINAU, J., MEGANCK, S., LAZAR, C., WEISS-SOLIS, D. Y., COLETTA, A., WALKER, N., BERSINI, H. & NOWÉ, A. (2010b). Sequential Application of Feature Selection and Extraction for Predicting Breast Cancer Aggressiveness. In: *Computational Systems-Biology and Bioinformatics*, vol. 115 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg, pp. 46–57. [Cited on page 8].

[Taminau et al. (2010)c] TAMINAU, J., MEGANCK, S. & NOWÉ, A. (2010c). Validation of Merging Techniques for Cancer Microarray Data Sets. In: *Proceedings of the 22nd Benelux Conference on Artificial Intelligence*. Luxembourg, Luxembourg. [Cited on page 7].

[Taminau et al. (2009)c] TAMINAU, J., MEGANCK, S., WEISS-SOLÍS, D. Y., VAN STAVEREN, W. C., DOM, G., VENET, D., BERSINI, H., DETOURS, V. & NOWÉ, A. (2009c). Validation of Merging Techniques for Cancer Microarray Data Sets. *Australian Journal of Intelligent Information Processing Systems* **10**(4), 4–11. [Cited on pages 7, 88, 89, 90, 165, and 166].

[Taminau et al. (2011)b] TAMINAU, J., STEENHOFF, D., COLETTA, A., MEGANCK, S., LAZAR, C., DE SCHAETZEN, V., DUQUE, R., MOLTER, C., BERSINI, H., NOWÉ, A. & WEISS SOLÍS, D. Y. (2011b). inSilicoDb: an R/Bioconductor package for accessing human Affymetrix expert-curated datasets from GEO. *Bioinformatics* **27**(22), 3204–5. [Cited on pages 6, 32, 42, 47, 57, 122, and 164].

[Tan et al. (2003)] TAN, P. K., DOWNEY, T. J., SPITZNAGEL, E. L., XU, P., FU, D., DIMITROV, D. S., LEMPICKI, R. A., RAAKA, B. M. & CAM, M. C. (2003). Evaluation of gene expression measurements from commercial microarray platforms. *Nucleic Acids Res* **31**(19), 5676–84. [Cited on page 20].

[Thorrez et al. (2008)] THORREZ, L., DEUN, K. V., TRANCHEVENT, L.-C., LOMMEL, L. V., ENGELEN, K., MARCHAL, K., MOREAU, Y., MECHELEN, I. V. & SCHUIT, F. (2008). Using ribosomal protein genes as reference: a tale of caution. *PLoS ONE* **3**(3), e1854. [Cited on page 82].

[Tomás et al. (2012)] TOMÁS, G., TARABICHI, M., GACQUER, D., HÉBRANT, A., DOM, G., DUMONT, J. E., KEUTGEN, X., FAHEY, T. J., MAENHAUT, C. & DETOURS, V. (2012). A general method to derive robust organ-specific gene expression-based differentiation indices: application to thyroid cancer diagnostic. *Oncogene* . [Cited on pages 6 and 61].

[Trapnell et al. (2009)] TRAPNELL, C., PACHTER, L. & SALZBERG, S. L. (2009). TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics* **25**(9), 1105–11. [Cited on page 17].

[Trapnell et al. (2012)] TRAPNELL, C., ROBERTS, A., GOFF, L., PERTEA, G., KIM, D., KELLEY, D. R., PIMENTEL, H., SALZBERG, S. L., RINN, J. L. & PACHTER, L. (2012). Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nat Protoc* **7**(3), 562–78. [Cited on page 54].

[Trapnell et al. (2010)] TRAPNELL, C., WILLIAMS, B. A., PERTEA, G., MORTAZAVI, A., KWAN, G., VAN BAREN, M. J., SALZBERG, S. L., WOLD, B. J. & PACHTER, L. (2010). Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat Biotechnol* **28**(5), 511–5. [Cited on page 17].

[Tseng et al. (2012)] TSENG, G. C., GHOSH, D. & FEINGOLD, E. (2012). Comprehensive literature review and statistical considerations for microarray meta-analysis. *Nucleic Acids Res* **40**(9), 3785–99. [Cited on page 74].

[Tu et al. (2002)] TU, Y., STOLOVITZKY, G. & KLEIN, U. (2002). Quantitative noise analysis for gene expression microarray experiments. *Proc Natl Acad Sci USA* **99**(22), 14031–6. [Cited on page 19].

[Urgard et al. (2011)] URGARD, E., VOODER, T., VÕSA, U., VÄLK, K., LIU, M., LUO, C., HOTI, F., ROOSIPUU, R., ANNILO, T., LAINE, J. et al. (2011). Metagenes associated with survival in non-small cell lung cancer. *Cancer Inform* **10**, 175–83. [Cited on page 128].

[Vandesompele et al. (2002)] VANDESOMPELE, J., PRETER, K. D., PATTYN, F., POPPE, B., ROY, N. V., PAEPE, A. D. & SPELEMAN, F. (2002). Accurate normalization of real-time quantitative RT-PCR data by geometric averaging of multiple internal control genes. *Genome Biol* **3**(7), RESEARCH0034. [Cited on page 76].

[Venter et al. (2001)] VENTER, J. C., ADAMS, M. D., MYERS, E. W., LI, P. W., MURAL, R. J., SUTTON, G. G., SMITH, H. O., YANDELL, M., EVANS, C. A., HOLT, R. A. et al. (2001). The sequence of the human genome. *Science* **291**(5507), 1304–51. [Cited on pages 12 and 16].

[Wang et al. (2007)] WANG, J. Z., DU, Z., PAYATTAKOOL, R., YU, P. S. & CHEN, C.-F. (2007). A new method to measure the semantic similarity of GO terms. *Bioinformatics* **23**(10), 1274–81. [Cited on page 80].

[Wang et al. (2009)] WANG, Z., GERSTEIN, M. & SNYDER, M. (2009). RNA-Seq: a revolutionary tool for transcriptomics. *Nat Rev Genet* **10**(1), 57–63. [Cited on pages 17, 18, and 163].

[Warnat et al. (2005)] WARNAT, P., EILS, R. & BRORS, B. (2005). Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes. *BMC Bioinformatics* **6**, 265. [Cited on pages 104 and 105].

[Wikman et al. (2004)] Wikman, H., Seppänen, J. K., Sarhadi, V. K., Kettunen, E., Salmenkivi, K., Kuosma, E., Vainio-Siukola, K., Nagy, B., Karjalainen, A., Sioris, T. et al. (2004). Caveolins as tumour markers in lung cancer detected by combined use of cDNA and tissue microarrays. *J Pathol* **203**(1), 584–93. [Cited on page 133].

[Wu (2009)] Wu, Z. (2009). A review of statistical methods for preprocessing oligonucleotide microarrays. *Stat Methods Med Res* **18**(6), 533–41. [Cited on page 24].

[Wu & Irizarry (2004)] Wu, Z. & Irizarry, R. A. (2004). Preprocessing of oligonucleotide array data. *Nat Biotechnol* **22**(6), 656–8; author reply 658. [Cited on pages 23 and 24].

[Xia et al. (2009)] Xia, X.-Q., McClelland, M., Porwollik, S., Song, W., Cong, X. & Wang, Y. (2009). WebArrayDB: cross-platform microarray data analysis and public data repository. *Bioinformatics* **25**(18), 2425–9. [Cited on page 105].

[Xu et al. (2008)] Xu, L., Tan, A. C., Winslow, R. L. & Geman, D. (2008). Merging microarray data from separate breast cancer studies provides a robust prognostic test. *BMC Bioinformatics* **9**, 125. [Cited on pages 74 and 133].

[Yale Law School Roundtable on Data and Code Sharing (2010)] Yale Law School Roundtable on Data and Code Sharing (2010). Reproducible Research: Addressing the need for data and code sharing in computational science. *Computing in Science and Engineering* **12**(5), 8–13. [Cited on page 36].

[Yasrebi et al. (2009)] Yasrebi, H., Sperisen, P., Praz, V. & Bucher, P. (2009). Can survival prediction be improved by merging gene expression data sets? *PLoS ONE* **4**(10), e7431. [Cited on page 69].

[Yu et al. (2010)] Yu, G., Li, F., Qin, Y., Bo, X., Wu, Y. & Wang, S. (2010). GOSemSim: an R package for measuring semantic similarity among GO terms and gene products. *Bioinformatics* **26**(7), 976–8. [Cited on page 80].

[Zhu et al. (2008)] ZHU, J., HE, F., HU, S. & YU, J. (2008). On the nature of human housekeeping genes. *Trends Genet* **24**(10), 481–4. [Cited on page 77].

[Zoubarev et al. (2012)] ZOUBAREV, A., HAMER, K. M., KESHAV, K. D., MCCARTHY, E. L., SANTOS, J. R. C., ROSSUM, T. V., MCDONALD, C., HALL, A., WAN, X., LIM, R. et al. (2012). Gemma: A resource for the re-use, sharing and meta-analysis of expression profiling data. *Bioinformatics* . [Cited on page 44].

Index