Escola Tècnica Superior d'Enginyeria ETSE-UV

MÁSTER UNIVERSITARIO EN BIOINFORMÁTICA

VNIVERSITAT DE VALÈNCIA

TRABAJO DE FIN DE MÁSTER

Systems Biology Strategies to Study Cancer Metabolism

AUTOR:
Daniel Ortiz Martínez

TUTORES:
Joaquín Dopazo Blázquez
Vicente Arnau Llombart

Septiembre, 2016

VNIVERSITAT DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria ETSE-UV

MÁSTER UNIVERSITARIO EN BIOINFORMÁTICA

TRABAJO DE FIN DE MÁSTER

Systems Biology Strategies to Study Cancer Metabolism

AUTOR:
Daniel Ortiz Martínez

TUTOR:
Joaquín Dopazo Blázquez
Vicente Arnau Llombart

TRIBUNAL:

PRESIDENTE/A:

VOCAL 1:

VOCAL 2:

FECHA DE DEFENSA:

CALIFICACIÓN:

ACKNOWLEDGMENTS

The author of this work wishes to thank his MSc Thesis supervisors Joaquín Dopazo and Vicente Arnau for giving him the opportunity to work in the fascinating area of bioinformatics and systems biology. In addition to this, the author is also grateful to Cankut Çubuk for all his help with some aspects of this thesis, and in particular for his pipeline to obtain the sets of lowly and highly expressed reactions when working with microarray data.

ABSTRACT

The purpose of this research is to study human metabolism, and more specifically the metabolism of cancer, under a systems biology perspective. In particular, we applied the so-called flux balance analysis (FBA) technique to study the differences between healthy and cancerous cell metabolism. FBA is a systems biology technique able to obtain metabolic models for large-scale systems, in contrast to the severe scale limitations of other popular approaches such as that based on ordinary differential equations. Due to the fact that conventional FBA is not able to reflect how metabolism works for each specific tissue, we applied the tissue-specific FBA method proposed by Shlomi and collaborators, which works by integrating gene expression data with metabolic model reconstructions.

Using tissue-specific FBA, we compared the metabolism of sixty samples of normal renal cells with that of another sixty samples coming from cancerous cells extracted from The Cancer Genome Atlas (TCGA) database. To carry out the analysis, RNA-Seq expression data provided by TCGA was integrated with the human metabolic model reconstruction stored in the Recon 2 database. The obtained results were used to conduct a differential reaction expression study based on statistical hypothesis testing. The study produced a p-value for each metabolic reaction. To facilitate the interpretation of the p-values, they were graphically represented in metabolic maps following two different strategies. On the one hand, the pre-generated metabolic maps provided by the Escher visualization tool were used. On the other hand, we designed our own visualization tool that uses Graphviz to generate metabolic maps in an automatic way.

Differential reaction expression results revealed alterations in different metabolic subsystems when comparing healthy and cancerous renal cells. In particular, such alterations were present in specific parts of the amino-acid, carbohydrate and retinol metabolisms.

In addition to the above mentioned work, we also developed an open-source software toolkit called *Flux Capacitor*, which incorporates many features useful to apply FBA techniques. The toolkit was used to compute the results of all of the experiments reported in this thesis.

CONTENTS

List of Figures

List of Tables

List of Algorithms

CHAPTER 1
INTRODUCTION

1.1 Systems Biology

1.1.1 Introduction

The extraordinary successes of the Human Genome Project and ensuing technological advancements, particularly in sequencing, microarray-based gene expression profiling and mass spectrometry, have promoted a deep transformation of the biological and medical sciences. The key aspect of this transformation is a shift toward a more holistic approach to biology and the emergence of the field of systems biology.

Systems biology is the scientific discipline that studies the systemic properties and dynamic interactions in a biological object, be it a cell, an organism, a virus, or an infected host, in a qualitative and quantitative manner and by combining experimental studies with mathematical modeling (Klipp et al. 2016). Systems biology uses biochemical networks as a main concept, investigating its components and interactions with the help of experimental high-throughput techniques and dedicated small-scale investigations. The acquired knowledge is later integrated into networks and dynamical simulation models.

Systems biology heavily relies on mathematical models as a tool to explain biological phenomena. In this context, a model is an abstract representation of biological objects or processes with the purpose of better understand or simulate their properties and behavior. Models can take many different forms, from graphical representations to mathematical formulas. Typically, models are based on well-established physical laws that justify their general form (e.g. thermodynamical laws). The crucial aspect of a given model, no matter how it was defined, is its ability to accurately answer the biological questions under study.

Defining a biological model involves characterizing a whole range of elements, including its scope; its quantitative aspects expressed in terms of variables, parameters and constants; the dynamical behavior of the system being studied; the different configurations or states of the system variables that are relevant for the model scope; the set of stationary (or steady) states, which are those where all variables remain constant in time; etc.

As it has already been mentioned, networks constitute a central concept in systems biology. Systems biology uses networks to study topics such as protein to protein interactions, protein to RNA interactions, metabolism or signaling pathways. Networks are represented by graphs composed of nodes that are connected by edges.

According to Klipp et al. 2016, three different kind of models are used in the context of systems biology: i) network-based models, ii) rule-based models and iii) statistic models. Network-based models describe and analyze properties, states or dynamics of networks. Frequently used models that fall into this category would be systems of ordinary differential equations to model biochemical reaction networks, or Petri nets to study metabolism. Rule-based models are composed of elements whose state is determined by a set of rules. Examples of this kind of models would be cellular automata, composed of a grid of cells whose states are updated based on rules depending on the states of the neighboring cells, or the more complex agent-based models, where each model entity (such as a protein or cell) is considered as an autonomous agent governed by its own rules. The third kind of biological models are those based on statistics. These models are receiving increasing attention in the field of systems biology, due to the massive data production that characterizes the *omic* technologies. Statistical models are useful in this context due to their ability to establish relations between observed data and to guide the analysis necessary to understand the underlying structures of the system under study. Examples of statistical models applied in systems biology are linear regression or analysis of variance (ANOVA).

One important task developed in systems biology is to integrate data derived from the application of high-throughput techniques. These techniques have evolved rapidly during the last few years, generating a huge amount of information. Combining information obtained from different sources and datasets can be useful for different research purposes such as biomarker identification, drug discovery or the study of complex disease conditions. Data combination approaches often involve sophisticated analysis and data handling techniques, including data normalization, quality control, statistical analysis or visualization techniques, just to name a few.

Another important element of systems biology is the study of model organisms. Model organisms are species that have become extremely useful for scientific research. Important factors that differentiate model organisms from the rest are: culture conditions (ease of handling in laboratory environments), cost, size (smaller organisms can be studied in a higher number), lifespan (short lifespans allow an easier study of aging), etc. The main motivation behind the study of a specific model organism is the possibility to export the biological findings made for such organism to other species or even humans. Model organisms range from prokaryotic organisms to single and multicellular eukariotic species, up to mammals.

1.1.2 Systems Biology and Bioinformatics

Systems biology is rapidly gaining widespread interest from the research community during the last years. The emphasis of this discipline on the word *systems* is motivated by the lack of knowledge about how the different components of biochemical systems, that are often known and described in meticulous detail in an isolated manner, interact with each other to produce the spatial and temporal behavior that constitutes the hallmark of biochemical systems (Kitano 2002). On the other hand, bioinformatics originated from the necessity to provide automatic tools to handle larger and larger amounts of biological data. According to Likić et al. 2010, bioinformatics continues to grow as a discipline in this important role, but is also increasingly merging and contributing to systems approaches by creating new tools useful to integrate existing knowledge about individual biological objects. Examples of these contributions would be biological databases, modeling tools, tools for network visualization, file formats for biological information exchange, etc. As a result, the crucial role bioinformatics plays in modern systems biology puts mathematical and computational sciences at the same level as analytical and experimental biology.

1.2 Modeling Metabolism

Metabolism constitutes one important example of the biochemical networks studied by the discipline of systems biology. Metabolism is the whole set of biochemical reactions occurring inside the cells of a living organism. Such biochemical reactions convert molecules or reactants into different products required to run cellular processes. Reactants are transformed into products in a sequence of steps that is referred to as a metabolic pathway. At each individual pathway step, a chemical reaction catalyzed by enzymes takes the products obtained in the previous step and produces new intermediate molecules that will constitute the reactants of the next step. Both the reactants and the products of each step receive the name of metabolites.

Metabolism can be studied by means of the above mentioned network-based models provided by the systems biology discipline. Arguably, the most popular of such models is based on the use of systems of ordinary differential equations (ODEs). ODEs allow to study the dynamic properties of metabolism due to the introduction of a continuous time axis in their formulation. Another well known systems biology framework to study metabolism assumes that system behavior is static and does not depend on time. Under these circumstances, flux balance analysis (FBA) can be applied. FBA is a mathematical approach for analyzing the flow of metabolites through a metabolic network. FBA removes the time axis considered by the ODE framework and replace it by the concept of steady state, where the values of the variables describing the system are fixed but subject to certain constraints.

In spite of the fact that the ODE framework allows us to obtain very accurate metabolic models, this kind of models requires detailed information about quantitative aspects of the involved biochemical reactions, such as kinetic constants, or enzyme intracellular concentration limits (Domach et al. 1984; D. Fell 1996). The lack of this information restricts the applicability of the ODE framework to small-scale systems. In contrast, FBA solves this problem by relying solely on simple physical constraints, enabling the analysis of large-scale metabolic networks (Price et al. 2003).

One challenge that arises when modeling metabolism is how to take into account the external and internal forces that influence the behavior of metabolic networks. One example of this would be the available amount of nutrients. Another example, particularly important from the perspective of this thesis, is gene expression. It is still not well understood how metabolism works for each specific human tissue (Shlomi et al. 2008), and previous studies demonstrate that changes in gene and protein expression levels play a major role in controlling tissue-specific metabolic functions (Son et al. 2005; Yanai et al. 2005; Levine et al. 2006). Therefore, successfully integrating transcriptomic and metabolic information can constitute one step forward to a more accurate modeling of human metabolism.

The study of human metabolism is experiencing an increasing interest from the research community due to the fact that metabolic diseases such as diabetes and obesity have become a major source of morbidity and mortality (Lanpher et al. 2006; Muoio and Newgard 2006). Another example of disease that is being studied under a metabolic perspective is cancer, since it has been demonstrated that the growth and survival of cancerous cells, and even their malignant transformation from healthy ones require specific alterations in normal cell metabolism (Cairns et al. 2011). As a result, accurately studying human metabolism by means of systems biology techniques holds the promise of greatly advancing our understanding of widespread diseases.

1.3 MSc Thesis Scope

This work is devoted to the study, under a systems biology perspective, of human metabolism and in particular, of the metabolism of cancer. For this purpose, we will integrate transcriptomic and metabolic information existing in well known databases by means of FBA techniques, which are used here due to their ability to model complex metabolic networks, as it was explained in the previous section. The resulting information will be statistically analyzed and graphically represented, allowing us to extract conclusions about the differences between normal and cancer metabolism.

The rest of this document is organized as follows: Chapter 2 enumerates the scientific and technologic goals pursued in this work. Chapter 3 describes common techniques for gene expression profiling that will be useful to obtain integrated models of metabolism. Chapter 4 explains the foundations of the FBA framework and other related approaches. Chapter 5 discusses different alternatives to visualize the results produced as a result of the application of FBA. Chapter 6 describes some databases relevant for FBA-based modeling of metabolism. Chapter 7 shows the results of the experiments we carried out. Chapter 8 explains the conclusions and future work. Finally, Appendix A describes the open-source software that has been developed for this MSc Thesis.

Chapter 2
Scientific and Technologic Goals

THIS MSc Thesis is focused on the application of FBA techniques to the study of human metabolism, with a particular emphasis on cancer metabolism. We define the following list of scientific ([SC]) and technologic ([TC]) goals:

- **Integration of transcriptomic and metabolic information using FBA** [SC]

  Data integration constitutes one key task within the discipline of systems biology. When studying metabolism, there are many factors that may be affecting its behavior. As it was explained in Section 1.2, one important example of these factors is gene expression, due to the essential role it plays in controlling tissue-specific metabolic functions. We will study and implement techniques to integrate transcriptomic and metabolic information so as to be able to study human metabolism accounting for the important influence of gene expression.

- **Review of available software for FBA** [TC]

  The application of FBA techniques requires sophisticated mathematical tools, commonly referred to as *solvers*, that calculate the metabolite flows of a metabolic network given a set of constraints. There are numerous software implementations of such solvers, ranging from commercial to free and open-source software, with varied capabilities and limitations. Here we evaluate and apply some of them so as to determine the most appropriate one taking into account the goals of this thesis.

- **Statistical analysis of FBA results** [SC]

  In order to rigorously evaluate the differences between normal and cancer cell metabolism, it will be necessary to perform statistical significance tests over the results derived from the application of FBA techniques. The specific method used for such tests will be carefully selected taking into account the nature of the data under study.

- **Visualization techniques for FBA results** [SC]

  Visualization constitutes an important step required to extract conclusions from FBA data. Typically, the metabolic network under study is graphically represented, showing the fluxes associated to the different reactions. However, creating these graphical representations is not trivial when large-scale metabolic networks are analyzed, due to the huge number of elements that compose such networks. Here we study some existing tools and methods than can be applied to obtain easily interpretable representations of the data derived from FBA.

- **Comparison between normal and cancer metabolism** [SC]

  Statistical analysis and graphical representations of FBA results will be used in combination with previously existing biological knowledge so as to arrive to meaningful conclusions regarding the differences between normal and cancer metabolism.

- **Development of open-source software for FBA** [TC]

  The development of open-source software in a research context is useful to speed up the propagation of new ideas and results throughout the scientific community. For this purpose, we have created a free and open-source software toolkit implementing the different techniques tested in this work.

CHAPTER 3
GENE EXPRESSION PROFILING

GENE expression constitutes one key factor affecting metabolic behavior as it was explained in Section 1.2. In this chapter we describe two well known techniques for gene expression profiling, namely, microarray-based profiling and RNA sequencing. The transcriptomic data derived from these two techniques will be useful to build complex metabolic models, as it is later explained in Chapter 4.

3.1 Microarray-Based Profiling

3.1.1 Introduction

A DNA microarray or DNA chip is a solid surface where there is attached a collection of DNA spots. Each DNA spot contains a microscopic quantity of a specific DNA sequence, known as *probes*. Such probes can be a short section of a gene or other DNA element that are used to *hybridize* a nucleic acid sample. Hybridization is a process by which two single-stranded deoxyribonucleic acid (DNA) or ribonucleic acid (RNA) molecules are combined in a single double-stranded molecule. Typically, microarray technology hybridize the probe content with cDNA, a DNA molecule obtained from messenger RNA (mRNA) using reverse transcriptase.

DNA microarrays constitute a high-throughput method for gene expression profiling (DeRisi et al. 1997). In a gene expression experiment using DNA chips, the expression levels of thousands of genes are simultaneously measured to study how certain diseases, treatments, developmental stages, etc. affect cellular activity.

A typical DNA microarray experiment involves the following sequence of steps:

1. **Construction of the chip**: this step is usually automated by means of robots and it requires a DNA library. A DNA library is a collection of DNA fragments that is stored and propagated in a population of micro-organisms through the process of molecular cloning. Polymerase chain reaction or PCR (Saiki et al. 1988) is then used to amplify individual clones and spotted in a regular pattern on the microarray surface typically made of plastic, glass or silicon. The DNA fragments that compose the library are often selected based on the so-called ESTs (expressed sequence tags) present in public databases. ESTs are short subsequences of cDNAS that are commonly used to identify gene transcripts.

2. **RNA preparation**: DNA chips work with RNA extracted from two different samples to be compared. Hence, the RNA samples should be first extracted before continuing with the experiment.

3. **RNA transcription**: RNA is transcribed into cDNA by means of reverse transcriptase and labeled with a fluorescent dye. The dyes for the two samples emit light at different wavelengths. Typically, red and green dyes are used.

4. **cDNA hybridization**: the cDNAs are later incubated with the chip, where they hybridize to the spot that contains the complementary fragment.

5. **Signal scanning**: after washing, the ratio of the fluorescence intensities for red and green are measured and displayed as false color picture. Spots of pure red or green indicate a large excess of RNA from one or the other sample, while yellow spots show that the amount of this specific RNA was roughly equal in both samples. Very low amounts of both RNA samples result in dark spots.

6. **Signal quantization and analysis**: the ratios obtained during the previous step can be quantified numerically and used for further analyses.

3.1.2 Bioinformatics Pipeline for Microarrays

Once a typical microarray experiment like that described above has been performed, an additional set of bioinformatic processes is executed so as to extract biological information from the microarray data. The most common goal of this is to study differential expression of genes between samples.

Overview

Gentleman, Carey, Huber, Irizarry, et al. 2005 provide a very detailed guide about a whole set of bioinformatic tools and analyses usually applied over microarray data. Here we group these tasks into three different steps:

1. **Data preprocessing**: raw microarray data are the intensities read for each probe. In practice, these data is heavily manipulated before obtaining the genomic-level measurements that are commonly used in gene expression studies. This procedure is commonly referred to as preprocessing.

2. **Differential expression testing**: once the probe intensities have been normalized, the resulting numerical values can be used to study if there are differences in gene expression across the different samples involved in the experimentation.

3. **Systems biology analyses**: the application of systems biology techniques over the microarray data can provide additional insight with respect to differential expression testing. One of the defining aspects of the applied techniques is its integrative character, where the expression data is combined with other sources of biological information.

Pipeline Details and Software Tools

Once the microarray pipeline for DNA chips has been introduced, we explain some additional details of it along with existing software able to solve the required tasks.

Among the available software options to process microarray data, there is one that have gained substantial popularity during the last years. This option is the Bioconductor open-source project for the analysis and comprehension of genomic data (Huber et al.

2015). Bioconductor is rooted in the open-source statistical computing environment R (R Core Team 2015) and offers a wide range of bioinformatic tools, including a whole set of them for microarray data processing.

According to Gentleman, Carey, Huber, Irizarry, et al. 2005, data preprocessing can be divided into 6 tasks: image analysis, data import, background adjustment, normalization, summarization, and quality assessment. Image analysis converts the pixel intensities in the scanned images into probe-level data. Data import methods are needed to collect information that is often scattered across a number of files or database tables. Background adjustment is required due to the fact that part of the measured probe intensities are due to non-specific hybridization and the noise in the optical detection system. Normalization enables direct comparison of measurements from different array hybridizations due to different sources of variation. In some platforms, summarization is needed because transcripts are represented by multiple probes. Finally, Quality assessment is important since it detects divergent measurements beyond the reasonable level of random fluctuations, allowing us to discard such measurements in subsequent analyses.

Bioconductor does not offer image processing software. Instead, it assumes that we start with numeric probe-level data as input. Such data is typically represented as a rectangular matrix. In addition to this, probes are annotated with information selected by the manufacturer of the chip. This information typically consists of a sequence identifier that can be mapped to genomic databases. The chip designers also provide data describing the array layout, which includes the physical position of each probe in the array. On the other hand, Bioconductor also expects information describing the samples involved in the experiment.

The most commonly used DNA chips for gene expression profiling are the Affymetrix GeneChip arrays. These arrays use short oligonucleotides to probe for genes in an RNA sample. Bioconductor provides the *affy* library (Gautier et al. 2004) to deal with data derived from Affymetrix chips, including procedures for data importing, background adjustment, normalization, summarization and quality control. Alternative and well known packages complementing the functionality provided by *affy* are also available, such as the *gcrma* package (Wu and Irizarry 2016) for background adjustment or the *affyPLM* package (Bolstad et al. 2005), which contains procedures useful for quality control.

Differential expression testing typically involves one or two group t-test comparisons, multiple group ANOVA and some additional and more general linear model tests. Despite the fact that the linear assumption is not always accurate, it is often applied because of the easier interpretability of the resulting models. All of the above mentioned tests are parametric. Alternatively, it is possible to use non-parametric tests, such as the Mann-Whitney's U test or permutation tests. Parametric tests usually have a higher power if the underlying model assumptions (e.g. normality in the case of the t-test) are at least approximately fulfilled. Non-parametric tests have the advantage of making less strong assumptions on the underlying data distribution. In many microarray studies however, a small sample size leads to insufficient power for non-parametric tests (Gentleman, Carey, Huber, Irizarry, et al. 2005).

Due to the lack of knowledge regarding coregulation of genes, statistical tests are computed for each gene separately. This is the standard approach due to its straightforward application. However, testing genes separately has important drawbacks, being the most important of them the fact that the large number of hypothesis tests that are carried out potentially leads to an equally large number of falsely significant results. To tackle this problem, multiple testing procedures can be applied to assess the overall significance of the results for a set of hypothesis tests. For this purpose, they focus on specificity by controlling the false positive error rates such as the family-wise error rate or the false discovery rate (Dudoit et al. 2003).

Bioconductor incorporates different packages for differential expression testing. For instance, combining the *genefilter* package (Gentleman, Carey, Huber, and Hahne 2016) with the *multtest* package (Pollard et al. 2005) it is possible to conduct a set of t-tests adjusting the p-values in a way that the false positive error rate is controlled.

Finally, systems biology tools can be used to gain further biological insight over the microarray data. One typically applied resource is Gene Onthology (GO) (Ashburner et al. 2000). GO provides an onthology of defined terms representing gene product properties across all species. One of the main uses of GO is to perform enrichment analysis on specific gene sets, such as those that are up-regulated according to the results of differential expression studies. For this purpose, Bioconductor provides the *GSEA* package (Morgan et al. 2016).

3.2 RNA Sequencing

3.2.1 Introduction

In spite of the great usefulness that microarray technology has demonstrated in the past to measure gene expression, its use is not without important disadvantages, being some of them relevant from the systems biology perspective. One of the most severe limitations of DNA chips is its poor dynamic range: gene expression measurement is limited by background signal at the low end, and by signal saturation at the high end (Wang et al. 2009). This makes difficult to detect rare transcripts or highly abundant ones. Another important limitation is their inability to detect novel transcripts due to the use of transcript-specific probes.

The recent development of the so called next-generation sequencing (NGS) techniques has given birth to a new technology useful for gene expression profiling called RNA sequencing or RNA-Seq (Wang et al. 2009; Corney 2013). NGS is a generic term used to describe a range of modern technologies that allow us to sequence DNA and RNA much more quickly and cheaply than the previously used Sanger technology. RNA-Seq basically consists in the application of NGS techniques to gene expression profiling, effectively tackling some of the limitations that DNA chips present.

The process behind the RNA-Seq technique is quite simple:

1. **RNA preparation**: input RNA is isolated and purified.

2. **RNA transcription**: the RNA molecules are converted to cDNA.

3. **Sequencing**: the cDNA molecules, with or without amplification, are sequenced using NGS technology. The length of the reads depends on the used sequencing method, typically being between 30 and 400 base pairs (bps). This makes necessary to break RNAs into smaller fragments. As a result, the number of counts per transcript is proportional, not only to its expression level, but also to the transcript length.

4. **Count normalization**: to be able to compare the expression levels of different transcripts, and between libraries with different sequencing depth, the expression level should be normalized.

Since RNA-Seq has practically no background, its dynamic range is only limited by the sequencing depth, successfully tackling the above mentioned range limitation problem of DNA chips. Moreover, it is also not necessary to know in advance the genomic sequence of the specific organism to carry out a transcriptomic study, solving another important disadvantage of microarrays.

3.2.2 Bioinformatics Pipeline for RNA-Seq

After introducing the RNA-Seq technique, we devote this section to describe the bioinformatics processes and tools that are usually involved in the analysis of RNA-Seq data. In spite of the fact that RNA-Seq can be used for different purposes, such as the detection of alternative splicing or the study of single nucleotide polymorphisms, the primary objective of many biological studies is gene expression profiling between samples.

Overview

Oshlack et al. 2010 provide a detailed description of the typical RNA-Seq pipeline for differential expression. This pipeline is composed of 5 steps:

1. **Mapping**: this task tries to find the unique location where a short read is identical to a given reference genome.

2. **Summarization**: after obtaining the genomic locations for as many reads as possible, the summarization step aggregates the reads over some biologically meaningful unit, such as exons, transcripts or genes.

3. **Normalization**: normalization allows to accurately compare expression levels between samples. It has been shown that normalization constitutes an essential step in the analysis of differential expression.

4. **Differential expression testing**: genes that have changed significantly in abundance across experimental conditions are highlighted in this step. This usually involves performing statistical testing between samples of interest.

5. **Systems biology analyses**: as it was explained for microarray data in Section 3.1.2, the output of differential expression testing can be used to discover new biological knowledge. Again, this process is typically carried out by integrating information from different sources, since this is the hallmark of systems biology approaches.

Pipeline Details and Software Tools

Each step of the RNA-Seq pipeline described above can be executed by means of previously existing software tools. In this section we explain additional details of the tasks executed in the pipeline as well as some software tools relevant for such tasks.

Mapping can be a computationally demanding task due to the fact that the reference genome is never a perfect representation of the actual biological source of RNA being sequenced. Allowing a greater degree of fuzziness during mapping also increase the computational complexity of the algorithm. Regular mappers execute a first pass to heuristically find a reduced list of candidate locations followed by a more detailed evaluation using a complex alignment algorithm at local level. One way to carry out this initial heuristic pass is the use of the Burrows Wheeler transform, as it is done in the BOWTIE aligner (Langmead, Trapnell, et al. 2009). BOWTIE constitutes an example of a general purpose aligner. However, general aligners are not enough to address some common mapping problems, such as those situations where a given read spans exon boundaries. Under these circumstances, the read will not be mapped against the reference. There exist tools that allow us to solve this problem, such as the TopHat mapper (Trapnell, Pachter, et al. 2009), which relies on the alignment results obtained by means of BOWTIE so as to perform an analysis of the splice junctions between exons.

The output of the aligner tools should be processed so as to quantify the obtained reads. This process can be carried out in different ways, being the simplest one to count the number of reads overlapping the exons in a gene. However, one problem with this approach is that a significant proportion of reads map to regions outside annotated exons. Because of this, a number of alternative summarization techniques have been proposed. One example of this would be to only take into account those reads that map to coding sequences, as it is implemented in the above mentioned Tophat aligner. Another strategy is incorporated in the Cufflinks tool (Trapnell, Williams, et al. 2010), where the junction reads are included in the summarization or used to model the abundance of splicing isoforms.

Summarized counts should be normalized as a previous step to carry out further analyses. Normalization is required due to the fact that, given the same expression level, longer transcripts have higher read counts. To solve this problem, a common normalization method divides the summarized counts by the length of the gene (Mortazavi et al. 2008). This technique can be refined by taking into account the number of mapped reads, obtaining the RPKM (reads per kilobase of exon model per million mapped reads) measure (Mortazavi et al. 2008). These normalization techniques are incorporated in the ERANGE package.

After normalization, the data is ready to perform differential expression studies. The applied techniques differ from those used for microarray data due to their continuous nature in contrast to the discrete counts provided by RNA-Seq. Existing methods typically assume that RNA-Seq counts follow a Poisson distribution. This decision is based on empirical evidence presented in (Marioni et al. 2008). However, it has also been demonstrated that the Poisson assumption fails at accurately capturing biological variability (Robinson and Smyth 2007; Langmead, Hansen, et al. 2010). To address this problem, the Poisson distribution is replaced by a negative binomial distribution in the edgeR package (Robinson, McCarthy, et al. 2010).

Systems biology tools can also be useful when applied to RNA-Seq data in the same way as it was explained for microarray data. Again, one common analysis that can be carried out consists in the integration of gene expression data with the terms contained in GO. This has been standard practice when working with microarrays, but its application to RNA-Seq data is more difficult due to the gene length bias explained above. To solve this problem, the GOseq tool (Young et al. 2010) can be used. GOseq has been specifically developed for RNA-Seq data and is able to incorporate length or total count bias into gene set tests.

CHAPTER 4
FLUX BALANCE ANALYSIS OF METABOLISM

METABOLISM can be modeled as a biochemical network whose structure plays a fundamental role to define the quantitative and qualitative aspects of the phenotype of living organisms. This chapter is devoted to introduce one major application of systems biology useful for the structural analysis of metabolic networks: flux balance analysis (FBA).

4.1 Metabolic Network Reconstructions

Metabolic network reconstructions have become increasingly important for studying the systems biology of metabolism. The number of organisms with available metabolic reconstructions is growing at a similar pace to whole genome sequencing (Thiele and Palsson 2010). Such metabolic reconstructions started to develop during the previous decade as structured knowledge-bases abstracting relevant information of the biochemical processes taking place with given target organisms. After being built, the reconstructions can be transformed into mathematical models for its use in a wide range of computational biological studies.

The quality of existing metabolic reconstructions varies considerably due to the different amounts of available data and also because of the previous lack of a well defined methodology to guide the entire process. To solve this problem, Thiele and Palsson 2010 proposed a protocol composed of 5 stages that are briefly described here:

1. **Creation of a draft reconstruction**: a draft reconstruction based on the genome annotation of the target organism and biochemical databases is generated. This draft reconstruction is created in an automatic manner and contains a collection of genome encoded metabolic functions, some of which may be falsely included while other ones are missing.

2. **Refinement of manual reconstruction**: the second stage focuses on curation and refinement of the network content. Specifically, the metabolic functions and reactions previously collected are individually evaluated against organism-specific literature.

3. **Conversion from reconstruction to mathematical model**: the conversion requires three steps: i) the network reaction list is converted to a stoichiometric matrix S, where the columns represent the network reactions and the rows the network metabolites, the substrates in a reaction have negative coefficients while products have positive value; ii) definition of system boundaries: for all metabolites that can

be consumed or secreted by the target of a cell, a so-called exchange reaction needs to be added; iii) addition of constraints so as to obtain a condition-specific model (this will be further discussed in Sections 4.4, 4.5 and 4.6.

4. **Network evaluation**: this process consists of network verification, evaluation and validation. Common errors in metabolic networks include wrong reaction constraints, missing transport or exchange reactions, cofactors that cannot be consumed or produced, etc. The result of this evaluation allows to identify the so called network gaps, or missing metabolic functions in the reconstruction. Such gaps are removed by partially repeating stages 2 and 3.

5. **Prospective use**: After completing the previous stages, it is possible to start using the reconstruction in a prospective manner.

4.2 Systems Biology Markup Language

4.2.1 Introduction

The necessity of information standards to share systems biology models has greatly increased during the last years within the research community. The lack of such information sharing standards contrasts with the wide variety of computational tools used to carry out systems biology tasks. This software diversity has been the cause of numerous problems (Hucka et al. 2003): different and complementary tools use models in different formats, making multiple re-encodings necessary; discontinued development of existing tools make previously developed models unusable; multiplicity of modeling environments negatively impact the reproducibility of research because examining, testing and reusing such environments may not be straightforward, etc.

To address this problem, a forum titled *Software Platforms for Systems Biology* was formed. This forum initially included representatives from teams developing different software packages, such as *BioSpice*, *Cellerator*, *DBsolve*, etc. (see (Hucka et al. 2003) for an exhaustive list). The forum decided to develop a simple, XML-based language for representing and exchanging models between simulation and analysis tools: the Systems Biology Markup Language (SBML). The basis of SBML is called *SBML Level 1*. SBML Level 1 is the result of analyzing common features in representation languages used by different systems biology simulators, and comprises the minimal information required to describe non-spatial biochemical models. SBML is open to further extensions (termed *levels*) that will add new features requested and prioritized by the SBML community.

4.2.2 Overview of SBML Level 1

As stated by Hucka et al. 2003, a chemical reaction can be broken down into certain conceptual elements, namely, reactant species, product species, reactions, stoichiometries, rate laws and parameters in the rate laws. Analyzing or simulating a network of reactions requires to make explicit additional components, including compartments for the species and units on the various quantities. A model expressed in SBML format consists of lists of one or more of such components (Hucka et al. 2003):

- **Compartment**: represents a container of finite volume where the reactions take place.

- **Species**: a specie is a chemical substance or entity taking part in a reaction.
- **Reaction**: is a statement describing some transformation, transport or binding process affecting to one or more species. Reactions have associated rate laws describing at which pace they take place.
- **Parameter**: a parameter is a quantity that has a symbolic name.
- **Unit definition**: unit definitions are names for units involved in the expression of quantities of a model.
- **Rule**: a rule is a mathematical expression that is added to the model equations built from the set of reactions.

Typically, systems biology tools read models expressed in SBML and translates them into their internal representation. The skeleton of an SBML file as well as a sample model exploiting the capabilities of the format is explained in (Hucka et al. 2003).

4.3 Mathematical Foundations of FBA

FBA is a systems biology method that uses mathematical optimization to study biochemical networks. In this section we briefly introduce two mathematical tools belonging to the operations research discipline that lay the foundations of FBA, namely, linear programming and integer programming[a].

4.3.1 Linear Programming

Linear programming (LP) uses a mathematical model to describe the problem of interest. The word *linear* refers to the fact that, in this model, all the mathematical functions used are linear functions. On the other hand, the word *programming* has no relation with computer programming but it is a synonym for planning. As a result, LP involves the planning of activities to obtain an optimal result.

The most common type of application of LP involves allocating resources to activities. The available amount of each resource is limited, so they must be carefully allocated. The allocation process involves deciding the levels of the activities that achieve the best possible value of the overall measure of performance.

Notation

Certain symbols are commonly used to denote the components of the linear programming model. Such symbols are described below:

Z: value of overall measure of performance.

x_j: level of activity j (for $j \in \{1, 2, ..., n\}$).

c_j: increase in Z that would result from each unit increase in level of activity j.

[a] To create the content of this section we have used the excellent reference book on operations research written by Hillier and Lieberman 1986.

b_i: amount of resource i that is available for allocation to activities (for $i \in \{1, 2, ..., m\}$).

a_{ij}: amount of resource i consumed by each unit of activity j.

The LP problem consists in making decisions about the level of the activities, so $x_1, x_2, ..., x_n$ are called the *decision variables*.

Standard Form of the Model

After introducing the required notation, we are ready to define the standard form of the LP model, where the values of the variables $x_1, x_2, ..., x_n$ are to be decided so as to:

$$\text{Maximize} \quad Z = c_1x_1 + c_2x_2 + ... + c_nx_n$$

subject to the restrictions

$$\begin{aligned} a_{11}x_1 + a_{12}x_1 \; +... + a_{1n}x_1 \; &\leq b_1 \\ a_{21}x_2 + a_{22}x_2 \; +... + a_{2n}x_2 \; &\leq b_2 \\ &\vdots \\ a_{m1}x_n + a_{m2}x_n + ... + a_{mn}x_n &\leq b_m \end{aligned}$$

and

$$x_1 \geq 0, \;\; x_2 \geq 0, \;\; ..., \;\; x_n \geq 0$$

The previous formulation receives the name of *standard form* for the LP problem. From this point it is possible to introduce common LP terminology:

- **Objective function**: the function being maximized (Z).
- **Constraints**: the restrictions mentioned above are commonly referred to as constraints.
- **Functional constraints**: the first m constraints of the standard model.
- **Nonnegativity constraints**: the last row of constraints of the model in its standard form.

Sometimes, a specific problem does not perfectly fit into the standard form (e.g. the objective function should be minimized instead of maximized), but it is possible to make it fit after some specific transformations (see (Hillier and Lieberman 1986) for additional details).

Terminology for Solutions of the Model

In LP, any combination of values of the decision variables is called a solution, no matter if it is desirable or even allowable. Solutions are classified according certain adjectives:

- **Feasible solution**: a solution where all the constraints are satisfied.
- **Infeasible solution**: a solution where at least one of the constraints is violated. It is possible that a problem has no feasible solutions.

- **Optimal solution**: a feasible solution that has the most favorable value of the objective function. The most favorable value will be the largest if Z is to be maximized and the smallest if it is to be minimized. A problem may have multiple optimal solutions. Alternatively, it is possible that a problem has no optimal solutions because there is not any feasible solution or because the constraints do not prevent improving the value of the objective function indefinitely (in this case, we say we have an unbounded Z or an unbounded objective).

Solving Linear Programming Problems

After introducing the standard form of LP problems and some basic terminology, we are prepared to briefly describe how to find solutions for them. The most common technique for this purpose is the so-called *simplex method*.

The simplex method is an algebraic procedure that operates on linear programs in standard form. Such procedure basically consists on solving systems of equations. It can be demonstrated that the system of equations for a given problem determines a polytope as a *feasible region*. A feasible region is the set of points for a given problem that satisfy all of its constraints. The simplex algorithm begins at a starting vertex and moves along the edges of the polytope until the vertex of the optimum solution is reached.

The simplex algorithm is able to obtain the optimal solution very efficiently for a wide range of LP problems. However, there are some situations for which the algorithm may render unusable due to the necessity of executing a number of calculations that grows exponentially with problem size. Under these circumstances, other algorithms such as the so-called *interior point* algorithms can be used. Interior point algorithms differ from the simplex method in that they find their solution by moving through the interior of the polytope representing the feasible region instead of through its vertices.

Assumptions of Linear Programming

As it has been explained, the LP model allows to find the values of the decision variables that maximize a linear objective function subject to linear constraints. These mathematical properties imply that certain assumptions must hold about the activities and data of the problem being modeled. Here we provide a list of such assumptions:

- **Proportionality**: the contribution of each activity to the value of the objective function is proportional to the level of the activity.

- **Additivity**: every function in an LP model is the sum of the individual contributions of the respective activities.

- **Divisibility**: Decision variables in an LP model are allowed to have any values, including noninteger values, that satisfy the functional and nonnegativity constraints.

- **Certainty**: The value assigned to each parameter of a linear programming model is assumed to be a known constant.

4.3.2 Integer Programming

There are several problems that can be successfully framed as linear programming problems. However, one key limitation that hinders the applicability of the LP model is the above mentioned divisibility assumption, since there are also many practical situations where the decision variables only make sense if they have integer values.

Terminology

When the decision variables of the LP problem are forced to be integers, then we need to solve an *integer linear programming* problem, or simply an *integer programming* (IP) problem. To solve this kind of problems, we apply the linear programming model adding the restriction that the decision variables are integer. If only some of them are integer, this model is referred to as *mixed integer programming* (MIP).

One particular case of integer programming occurs when the decision variables correspond to yes-or-no decisions. We can represent this mathematically by means of decision variables x_j that take a zero value to represent *no*, and a value equal to one to represent *yes*. This kind of variables are called *binary variables* and the problems that contain them, *binary integer problems* (BIPs).

Solving Integer Programming Problems

LP problems can be efficiently solved using currently existing techniques, such as the simplex method. Due to the fact that IP problems are basically LP problems with a reduced set of possible solutions, it may seem that IP solutions are easier to find. Unfortunately, this is far from being true. On the one hand, despite the fact that IP problems are guaranteed to have just a finite number of possible solutions, this does not mean that the problem is easily solvable, since a finite number can be astronomically large. For instance, a BIP problem with n variables has 2^n possible solutions, if n is equal to 30, then the problem would have more than 1 billion solutions. On the other hand, the removal of noninteger solutions from the LP problem makes more difficult to guarantee that there is an optimal solution for it. It is precisely the existence of this guarantee what constitutes the key to the remarkable efficiency of the simplex method. Because of that, IP algorithms are typically based on the use of LP solvers, such as the simplex method. These solvers are used to find solutions for portions of the IP problem that can be related to the corresponding LP problem. Given an IP problem, the corresponding LP problem is referred to as the *LP relaxation*.

One main approach to IP problem solving is the use of the so-called *branch-and-bound algorithms*. These algorithms are able to efficiently explore the potentially huge set of feasible solutions. The approach works by partitioning the entire set of feasible solutions into smaller and smaller subsets that are organized in branches of a tree-like data structure. When exploring the tree of possible solutions, for a given solution subset the algorithm calculates bounds for the best solution that can be reached. The subset is discarded if its bounds indicate that it cannot possibly contains an optimal solution for the problem. The bounding process is typically performed by finding the optimal solutions for the LP relaxation of the IP problem.

4.4 Flux Balance Analysis

After describing the basic concepts of linear and integer programming, we are ready to present the FBA technique, which constitutes the main tool used in this thesis to study human metabolism.

4.4.1 Overview

FBA (D. A. Fell and Small 1986) is a widely used approach for studying genome-scale metabolic network reconstructions (for more details about such reconstructions, see Section 4.1). For this purpose, a mathematical model of the reconstruction is built and later used to find answers to specific biological questions. In contrast to the traditional approach to model metabolism based on ordinary differential equations, FBA uses very little information about kinetic parameters and metabolite concentrations. The words *flux balance* refer to one of the two basic assumptions made by the model. In particular, that the flow rates (or fluxes) of any compound being produced must be equal to the total amount being consumed when the system is in a *steady state*. The second assumption is that evolution has operated on the metabolism of the organism being studied, optimizing some biological goal, such as optimal growth or conservation of resources. When these two assumptions are put together, metabolism can be studied by means of mathematical optimization tools (and more specifically, using LP or IP solving methods as it will be explained in the following section).

According to Orth et al. 2010, the formulation of an FBA problem involves 5 different steps:

1. **Obtain metabolic network reconstruction**: the metabolic network reconstruction produces a list of stoichiometrically balanced biochemical reactions. Cell growth is typically incorporated into the reconstruction with a biomass reaction, which simulates metabolites consumed during biomass production. Additionally, the flow of metabolites in and out of the cell is represented as exchange reactions (examples of such metabolites are glucose and oxygen).

2. **Represent reactions and constraints mathematically**: a numerical matrix incorporating the stoichiometric coefficients of each reaction is generated.

3. **Obtain set of linear equations**: given the stoichiometric matrix, the steady state assumption determines a set of linear equations (more on this in the next section).

4. **Define objective function**: the mathematical model of metabolism is completed by adding an objective function related to a certain biological aspect of the cell that it is assumed to have been optimized through evolution. One example of that is the growth rate.

5. **Calculate fluxes**: mathematical optimization techniques are used to identify a flux distribution that optimizes the objective function.

4.4.2 Mathematical Representation of Metabolism

FBA represents metabolic networks as a stoichiometry balanced set of equations. For this purpose, it is necessary to know the stoichiometric coefficients affecting the metabolites involved in each reaction. These coefficients can be stored in an $m \times n$ stoichiometric matrix, S, where m is the number of metabolites and n is the number of reactions:

$$\mathrm{S} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

The goal of FBA is to find the values of the fluxes for each reaction, $v_1, v_2, ..., v_n$, that maximize a certain objective function such as the growth rate.

If the steady state assumption is made and we also consider some additional constraints regarding the upper and lower bounds for flux values, then it is possible to express the maximization problem posed by FBA as an LP problem:

$$\text{Maximize} \;\; Z = c_1v_1 + c_2v_2 + ... + c_nv_n$$

subject to steady state restrictions

$$\begin{aligned} a_{11}v_1 + a_{12}v_1 \; + ... + a_{1n}v_1 \; &= 0 \\ a_{21}v_2 + a_{22}v_2 \; + ... + a_{2n}v_2 \; &= 0 \\ &\vdots \\ a_{m1}v_n + a_{m2}v_n + ... + a_{mn}v_n &= 0 \end{aligned}$$

and flux range restrictions

$$l_1 \leq v_1 \leq u_1, \;\; l_2 \leq v_2 \leq u_2, \;\; ..., \;\; l_n \leq v_n \leq u_n$$

It should be noted that the previous problem is not an LP problem in standard form. However, this difficulty can be easily addressed by making some straightforward transformations (see (Hillier and Lieberman 1986) for more details).

Alternatively, it is possible to express the above problem in a more compact manner using matrix notation:

$$\begin{aligned} &\text{Maximize} && Z = \mathbf{c}^{\mathbf{T}} \cdot \mathbf{v} \\ &\text{Subject to} && \mathbf{S} \cdot \mathbf{v} = \mathbf{0} \\ & && \mathbf{l} \leq \mathbf{v} \leq \mathbf{u} \end{aligned} \tag{4.1}$$

where $\mathbf{S}$ is the stoichiometric matrix, $\mathbf{v}$ is the flux vector, $\mathbf{c}^{\mathbf{T}}$ is the vector of coefficients representing the increase in the objective function per each unit of increase of the fluxes[b], and $\mathbf{l}$ and $\mathbf{u}$ are the lower and upper bounds vectors for $\mathbf{v}$, respectively

4.5 Flux Variability Analysis

The optimal solution to an FBA problem is seldom unique. Moreover, for a given value of the objective function, there may exist different combinations of values for the fluxes that allow to achieve it. Flux variability analysis (FVA) (Mahadevan and Schilling 2003) constitutes an extension of the standard FBA technique, where the range of values that can take the fluxes while producing the optimal value of the objective function is calculated.

FVA is often used to determine the robustness of metabolic models in various simulation conditions. It is assumed that those reactions which can support a low variability of fluxes through them have a higher importance to an organism.

[b] $(\cdot)^{\mathbf{T}}$ represents the transposed matrix.

4.5.1 FVA Formulation

After finding the solution for the LP problem given in Equation 4.1, FVA solves two optimization problems for each flux of interest, v_i:

$$\begin{array}{ll} \text{Maximize/Minimize} & Z = v_i \\ \text{Subject to} & \mathbf{S} \cdot \mathbf{v} = \mathbf{0} \\ & \mathbf{c}^{\mathbf{T}} \cdot \mathbf{v} \geq \gamma Z_0 \\ & \mathbf{l} \leq \mathbf{v} \leq \mathbf{u} \end{array} \tag{4.2}$$

where Z_0 is the optimal value of the objective function for the LP problem expressed by Equation 4.1 and γ is a parameter that takes values between zero and one, allowing to obtain flux ranges for the optimal solution $\gamma = 1$ or for a suboptimal one $0 \leq \gamma < 1$ (working with suboptimal solutions can be useful to speed up calculations).

One disadvantage of FVA is its computational cost when it is to be applied over the whole set of flux variables. Specifically, a total of $2 \cdot n$ LP problems are to be solved if the metabolic network reconstruction contains n reactions. There are some ways to address this issue, as it is explained in the following section.

4.5.2 Improving FVA Computational Efficiency

The most straightforward strategy to reduce the time cost required by FVA is to make use of parallelism. Since each FVA subproblem given in Equation 4.2 is independent from the rest, it can be solved in a separated node of a multiprocessor or computer cluster, allowing us to achieve almost linear speedups when the number of processing nodes is increased.

However, it is also possible to improve the efficiency of FVA when it is executed on a single CPU. Gudmundsson and Thiele 2010 propose the *fastFVA* algorithm, which is based on the idea that the feasible region for the whole set of LP problems given by Equation 4.2 does not change with respect to that of the FBA problem in Equation 4.1. This suggests that solving each FVA LP problem from scratch is highly inefficient. In contrast, the initial FBA problem can be solved from scratch and the $2 \cdot n$ subsequent problems by starting from the last optimum solution found. The details of the technique can be found in Algorithm 4.1.

4.6 Tissue-Specific Modeling of Metabolism Using FBA

One limitation of current genome-scale metabolic network reconstructions is the fact that they are not tissue specific. When metabolism is modeled by means of constraint-based modeling methods such as FBA, finding a solution to this problem is not easy because of two reasons (Shlomi et al. 2008): first, there is not a clear objective function being optimized as it is the case with simple microorganisms, where biomass production is often used. Second, there is a lack of information on tissue-specific metabolite uptake and secretion, which is essential for the application of FBA.

To tackle the challenge of creating tissue specific metabolic network models, Shlomi et al. 2008 propose an FBA-based computational method integrating a genome-scale metabolic reconstruction with tissue-specific gene- and protein-expression data. The motivation behind this is that gene- and protein-expression levels play a major role in controlling tissue-specific metabolic functions, and a strong correlation between gene expression

```
input    : $\mathbf{S}$ (stoichiometric matrix), $\mathbf{v}$ (flux vector), $\mathbf{c}^\mathbf{T}$ (objective function coefficients)
           $\mathbf{l}$ (flux lower bounds), $\mathbf{u}$ (flux upper bounds), $\gamma$ (gamma parameter)
output   : $F$ (set of flux ranges)
auxiliar: $T$ (set of LP constraints), $Z_i$ (optimal objective function value at $i$'th iteration)
           $\mathbf{v}_i$ (optimal flux vector at $i$'th iteration)
begin
    $T := (\mathbf{S} \cdot \mathbf{v} = \mathbf{0}) + (\mathbf{l} \leq \mathbf{v} \leq \mathbf{u})$
    $Z_0, \mathbf{v}_0 :=$ LP_maximize_from_scratch$(\mathbf{c}^\mathbf{T} \cdot \mathbf{v},\ T)$
    $T := T + (\mathbf{c}^\mathbf{T} \cdot \mathbf{v} \geq \gamma Z_0)$
    $F := \emptyset$
    for $i := 1$ to $n$ do
        $Z_i^{\min}, \mathbf{v}_i^{\min} :=$ LP_minimize$(v_i,\ T,\ \mathbf{v}_{i-1}^{\min})$
        $Z_i^{\max}, \mathbf{v}_i^{\max} :=$ LP_maximize$(v_i,\ T,\ \mathbf{v}_{i-1}^{\max})$
        $F := F + (\mathbf{v}_i, Z_i^{\min}, Z_i^{\max})$
```

Algorithm 4.1: Pseudocode for the *fastFVA* algorithm.

and metabolic fluxes has been found when studying microorganisms. One interesting feature of the method is that observed expression levels are not considered as the final determinants of enzyme activity, but as a hint for the likelihood that the specific enzyme carries metabolic flux in its associated reactions. These hints are integrated into a global network, allowing the model to account for metabolic flux activity that is not reflected in the expression data (i.e. post-transcriptional regulatory effects). According to the paper authors, this strategy eliminates the necessity of having a priori knowledge about tissue-specific objective functions and metabolites exchanged.

4.6.1 Tissue-Specific FBA Formulation

The tissue-specific FBA formulation proposed in (Shlomi et al. 2008) starts from metabolic network information identical to that needed for standard FBA. In particular, it requires a stoichiometric matrix $\mathbf{S}$ with m metabolites and n reactions as well as lower and upper bounds vectors, $\mathbf{l}$ and $\mathbf{u}$, respectively, for the vector of fluxes $\mathbf{v}$. The method also relies on the definition of a set of *highly expressed reactions*, R_H and a set of *lowly expressed reactions*, R_L. Using this information, the following MIP model is built:

$$
\begin{aligned}
\text{Maximize} \quad & Z = \left(\sum_{i \in R_H} (y_i^+ + y_i^-) + \sum_{i \in R_L} y_i^+ \right) \\
\text{Subject to} \quad & \mathbf{S} \cdot \mathbf{v} = \mathbf{0} \\
& \mathbf{l} \leq \mathbf{v} \leq \mathbf{u} \\
& \mathbf{v}_i + y_i^+ (\mathbf{l}_i - \epsilon) \geq \mathbf{l}_i, \ i \in R_H \\
& \mathbf{v}_i + y_i^- (\mathbf{u}_i + \epsilon) \leq \mathbf{u}_i, \ i \in R_H \\
& \mathbf{l}_i (1 - y_i^+) \leq \mathbf{v}_i \leq \mathbf{u}_i (1 - y_i^+), \ i \in R_L \\
& y_i^+, y_i^- \in \{0, 1\}
\end{aligned}
\tag{4.3}
$$

where boolean variables y^+ and y^- represent whether a given reaction is active or not for highly expressed reactions. Specifically, a highly expressed reaction is considered to be active if it carries a significant positive flux that is greater than a positive threshold ϵ, or lower than a significant negative flux $-\epsilon$ for reversible reactions (typically $\epsilon = 1$). On the other hand, for lowly expressed reactions, y^+ represent if the reaction is inactive. The optimization maximizes the number of highly expressed reactions that are active and the number of lowly expressed reactions that are inactive.

The tissue-specific FBA method described above requires the definition of lists of lowly and highly expressed reactions. For this purpose, the first thing to be done is to decide if the different genes are absent or present. The strategy followed to take such decisions varies depending on whether the expression data comes from a microarray or an RNA-Seq experiment. It is worthy of note that the problem to be solved here is not exactly the same as that of studying differential gene expression mentioned in Chapter 3, since the goal is not the comparison of gene expression between samples but to decide if a gene is expressed or not.

Finally, after identifying absent and present genes, it is necessary to associate them to protein reactions, allowing us to obtain the R_H and R_L sets.

4.6.2 Microarray Lowly and Highly Expressed Reactions

As it was explained in Section 3.1, microarray data consists of a set of intensities read for the probes that compose the chip. Since on common gene expression microarrays, a given gene may be detected by multiple probesets, obtaining inconsistent or even contradictory measurements constitutes a potential problem that should be taken into account.

Assuming that the probe intensities have been preprocessed, the following sequence of steps can be executed to obtain the lists of lowly and highly expressed reactions:

1. **Select optimal microarray probeset to represent each gene**: Li et al. 2011 introduce a method called *Jetset* able to obtain unambiguous expression estimates of specified genes. The proposed technique defines scoring metrics to assess different aspects of each probeset, including specificity, splice isoform coverage and robustness against transcript degradation. Afterwards, each gene is assigned to a single representative probeset based on the computed scores.

2. **Identify absent/present probesets**: when using DNA chips, methods to determine whether a gene is expressed or not require the application of certain statistical analyses over the probeset data. Moreover, common available techniques impose the use of specific probeset data preprocessing algorithms as a pre-condition to generate results. In contrast to this, Warren 2016 proposes the so-called *Presence-Absence calls with negative Probeset* (PANP) method which does not have this limitation. PANP is based on the use of certain probesets present in common DNA chips, as a per-sample control of non-specific hybridization. In particular, such probesets are referred to as *Negative Strand Matching Probesets* (NSMPs) and they are present in some DNA chips due to mistakes in the strand direction of ESTs (expressed sequence tags, see more details in Section 3.1.1) annotated in public databases. As a result, some probesets of specific DNA chip models were designed in the reverse complement and cannot hybridize to their true ESTs. Warren 2016 proposes a technique that takes advantage of this fact to build a statistically representative set of negative controls, allowing to identify active and inactive genes.

3. **Convert probeset identifiers into gene identifiers**: after executing the previous steps, we already have a list of absent and present genes. However, such a list

represents genes as DNA chip probesets. This representation is not appropriate for our purposes, since in typical metabolic network reconstructions, genes are not identified by their probesets. To solve this problem, it is necessary to map the probeset identifiers to gene identifiers. One possibility is to use the gene identifiers provided in the Entrez Gene database (Maglott et al. 2011).

4. **Determine lowly and highly expressed reactions**: the final step consists in mapping information about absent or present genes to lowly or highly expressed reactions. Genome annotations often detail the association between gene and protein-reactions. In particular, a set of logical rules applied over the absent/present status of the genes determine whether the reactions are expressed or not (see (Thiele and Palsson 2010) for additional details).

4.6.3 RNA-Seq Lowly and Highly Expressed Reactions

When gene expression is measured by means of an RNA-Seq experiment, Hebenstreit et al. 2011 provide a technique useful to identify absent and present genes. In particular, the method is based on the empirical observation of two major messenger RNA abundance classes in different organisms, including human, mouse and Drosophila.

After normalizing the RNA-Seq data, a kernel density estimation procedure (Rosenblatt 1956) applied over the logarithm of RPKMs revealed that the majority of genes follows a normal distribution centered around 4, whereas the remaining ones formed a shoulder to the left of this main distribution[c] (see the plot in (Hebenstreit et al. 2011)). Further experiments demonstrated that the group of genes centered around 4 corresponded to present genes, and the other group to absent genes producing non-functional transcripts.

The two groups of genes observed in the kernel density estimate can be characterized in more detail by estimating the parameters of a gaussian mixture model with two components. For this purpose, the EM algorithm can be used (Dempster et al. 1977). After estimating the mixture model, it can be used to classify the genes as absent or present.

Once the genes have been classified as absent or present, we use the gene-protein-reaction logical rules present in gene annotations to obtain the lists of lowly and highly expressed reactions, in the same way as it was explained for microarray data.

4.7 Solvers for Mathematical Optimization

FBA of human metabolism involves solving LP problems composed of thousands of variables and constraints, since there is one variable and constraint per reaction, and current network reconstructions of human metabolism contain more than 7 000 of them. Tissue-specific FBA is even more demanding, since it introduces integer variables, forcing us to solve the more computationally expensive IP problems.

Solving the LP and IP problems that arise when studying human metabolism by means of FBA is possible using one of the excellent solvers that are currently available. These solvers differ in things such as the license, capabilities and efficiency. Here we have considered the use of three different packages (although there are other good alternatives):

- **GNU Linear Programming Kit (GLPK)**: GLPK[d] is an open-source software

[c] Genes with zero counts are excluded since they cannot be represented on the log scale.

[d] https://www.gnu.org/software/glpk/

package for solving large scale LP and MIP problems written in C. It provides a callable library and a standalone solver. The package is distributed under the GNU General Public License[e] (GPL) so it can be freely used, copied and modified, only requiring that the same rights are maintained if the software is copylefted.

- **Computational Infrastructure for Operations Research (COIN-OR)**: The COIN-OR project[f] aims at creating mathematical software for operations research. Among the different tools currently promoted by the COIN-OR project, we find CLP (COIN-OR Linear Programming), an open source LP solver written in C++, and CBC (COIN-OR Branch-and-Cut), a MIP solver written in the same language. Both tools provide callable libraries as well as standalone solvers. On the other hand, they are published under the Common Public License[g] that permits it use within proprietary software in contrast to the GPL.

- **CPLEX**: CPLEX[h] is a commercial software package for mathematical optimization currently maintained and developed by IBM. CPLEX initially offered a C implementation of the simplex method and later has included tools and solvers for other problems, including MIP. Similarly to GLPK and COIN-OR, CPLEX also provides a callable library as well as a standalone solver. CPLEX is particularly well suited to tackle very large LP and MIP problems very efficiently. However, since it is a commercial application, its functionality is restricted for free versions unless the prospective user opts for an academic license.

To obtain the empirical results that will be shown in Chapter 7, we will make use of the standalone solvers provided by the three packages described above. The three of them are able to read LP and IP problem definitions written in files. One supported file format is called *lp*. lp files are given in plain text and use a very simple syntax that allows to define the objective function and the constraints of LP or MIP problems. Figure 4.1 shows a very simple example extracted from the package's website. The example defines a MIP problem with 4 variables and 5 constraints, being `x4` an integer variable.

```
Maximize
 obj: x1 + 2 x2 + 3 x3 + x4
Subject To
 c1: - x1 + x2 + x3 + 10 x4 <= 20
 c2: x1 - 3 x2 + x3 <= 30
 c3: x2 - 3.5 x4 = 0
Bounds
 0 <= x1 <= 40
 2 <= x4 <= 3
General
 x4
End
```

Figure 4.1: Sample lp file format.

[e] https://www.gnu.org/licenses/gpl-3.0.html

[f] http://www.coin-or.org/index.html

[g] https://opensource.org/licenses/cpl1.0.php

[h] http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html

CHAPTER 5
VISUALIZATION OF METABOLIC NETWORKS

UNDER a systems biology perspective, network visualization constitutes an essential task due to the emphasis on data integration that characterizes this discipline. In this chapter we describe different resources and techniques useful for metabolic network visualization. In particular, the Escher visualization tool is presented in Section 5.1. The open-source network visualization project called Graphviz and its application to this study is explained in Section 5.2. Finally, an algorithm useful to reduce the complexity and size of metabolic networks is detailed in Section 5.3.

5.1 The Escher Visualization Tool

5.1.1 Introduction

With the advent of next-generation sequencing, it is now easy to comprehensively measure the complex interactions between genes, proteins, reactions and metabolites from which the phenotypic behavior of living organisms emerge. In spite of the fact that data acquisition has become substantially easier, data analysis is increasingly evolving into the primary bottleneck to discovery.

Data visualization plays an important role in addressing the data analysis bottleneck due to its ability to complement the information provided by statistical and modeling methods. Typically, visualization tools can be classified by the biological information being represented and also by the visualization style. For instance, three-dimensional objects are used to represent protein structures. In contrast, only one dimension is sufficient to represent a phylogenetic tree. On the other hand, metabolic pathways and other biological pathways have been typically represented as two-dimensional maps.

As stated by King et al. 2015, an appropriate visualization tool for metabolic pathways should satisfy six different core features: (i) biologic data should be clearly represented and in biochemically correct way, (ii) users should be able to navigate and search through the visualization, (iii) it should allow users to design and customize pathway maps, (iv) different data types can be represented using visual cues such as size and color, (v) maps can be imported and exported enabling interoperability with other tools and (vi) provides an application program interface (API) useful to integrate the tool within data analysis pipelines.

There are some examples of desktop applications able to satisfy all of the above mentioned core features. By contrast, the same is not true when we focus on existing web applications. Desktop applications have many advantages over web applications, including a greater speed or stability. On the other hand, web applications offer quicker deployment times, greater cross-compatibility, etc. However, recent performance improvements are making web tools competitive with desktop applications for many applications.

Escher (King et al. 2015) is a web application tool for visualizing data on biological pathways, designed to incorporate the six core features mentioned above for pathway visualization tools.

5.1.2 Main Functionality

Escher allows to build pathway maps provided that there is the necessary information about the names, stoichiometries, and associated genes for the biochemical reactions in an organism. More specifically, a constraint-based reconstruction and analysis (COBRA) model (Bordbar et al. 2014) is used. COBRA models are generally focused on metabolism, but can be applied to any biochemical reaction network.

Escher provides a graphic interface that allows to build pathways maps, adding new metabolites, reactions and genes. The building process can start from scratch or from previously built maps. Additionally, Escher assists the user during the definition of the map layout. Pathway maps can be imported and exported using the functionality of the interface (for additional details, see the next section).

On the other hand, Escher allows to visualize three types of data: reaction, metabolite and gene data. Using the Escher interface, it is possible to visualize the comparison of two datasets using different comparison functions. In addition to this, it is also possible to alter sizing and coloring options for map elements.

5.1.3 Design and Implementation

Programming Languages

Escher is a web application written primarily in JavaScript. The Escher code can be compiled into a single JavaScript file, and a JavaScript API is available for interacting with and extending an Escher visualization. The Escher website is built using the same API, and other web applications can be built on top of this library.

Escher also provides a Python package with extra features, such as access to Escher maps from Python terminals, offline access to Escher, a Python API for application development, etc.

Map and Model Database

One particularly interesting feature from the perspective of this work are the different pre-generated maps incorporated by the tool. Such maps are available for different model organisms, including the Recon 1 model of human metabolism. Recon 1 is the first version of the Recon X family of metabolic models (see more details in Section 6.1). Escher provides 5 different maps related to Recon 1 that will be used in this thesis:

- **Amino acid metabolism**: describes the various biochemical processes responsible for the synthesis of amino acids.
- **Carbohydrate metabolism**: the set of metabolic reactions related to the formation, breakdown and interconversion of carbohydrates.
- **Glycolisis**: the metabolic pathway that converts glucose into pyruvate.

- **Inositol retinol metabolism**: contains maps of the inositol phosphate metabolism and the pathway responsible of protein retinoylation. Inositol phosphate metabolism is especially interesting for cancer research since it is involved in cellular functions relevant for this disease, such as cell growth, apoptosis, cell migration, cell differentiation, etc.

- **Tryptophan metabolism**: describes the processes required to synthesize tryptophan, an amino acid used in protein biosynthesis and also a precursor to certain neurotransmitters.

File Formats

Escher input data can be provided in two different formats: comma separated values (CSV) and JavaScript object notation (JSON). With respect to the available output formats, graphic information can be exported to well known image formats, such as the Scalable Vector Graphics (SVG) or the Portable Network Graphics (PNG) formats.

License

Escher is open-source software hosted on GitHub[a], with a public bug tracker and tools for community contribution to the codebase. Documentation for Escher describing its features and providing detailed information on the JavaScript and Python APIs is also available.

5.2 Using Graphviz for Metabolic Network Visualization

Graphviz (E. R. Gansner and North 2000) is a general-purpose, open-source graph visualization tool. Graphviz has successfully been applied to create and manipulate graphs in a wide range of fields, including networking, bioinformatics, software engineering, database design, etc.

From the perspective of this work, the main difference between Graphviz and specific-purpose drawing tools such as Escher is the fact that Graphviz can be adapted to the user's specific needs, offering a much greater versatility. However, this increased versatility often comes at the cost of a substantial effort required from the user to obtain the desired results. Some of the experiments reported in Chapter 7 explore the advantages offered by the functionality incorporated in Graphviz. Below we provide a general description of such functionality as well as some additional details about the design and implementation of the tool.

5.2.1 Main Functionality

As stated by E. R. Gansner and North 2000, much work in graph manipulation and visualization has focused either on high-level interactive editors or on low-level graph libraries, whose usefulness is well demonstrated. A middle approach is offered by filters, which read and process an input stream, and produce an output stream. Filters have successfully been used in areas such as text processing or program compilation, due to their focus on

[a] https://github.com/zakandrewking/escher

symbolic computation and their ability to automate repetitive tasks. In contrast, manual interactive editors are not useful in these areas.

Graphviz is a toolkit of libraries and programs for creating, filtering and interacting with graphs, where filtering is just as important as interactive tools. Graphviz functionality is provided through four different components: libraries, layout tools, graphical tools and graph filters.

Libraries

The toolkit uses two libraries, Libgraph and Dynagraph, that provide low-level functionality. Libgraph allows to read, write and manipulate graph abstractions, whereas Dynagraph is built on top of Libgraph and provides a framework for displaying incrementally changing graphs.

Libgraph uses a specific data language called Dot, which is shared by the graph manipulation tools of Graphviz. The Dot language provides syntax for defining graphs, nodes and edges, plus the ability to attach string-valued name-attribute pairs to graph components. Sets of objects are modeled as subgraphs. The details of the Dot language can be found in (E. Gansner et al. 2010). Figure 5.1 provides a sample file in Dot format extracted from (E. Gansner et al. 2010) exploiting some basic drawing features (different colors, node shapes, arc labels, etc.) The result of plotting this graph file is shown in Figure 5.2.

```
digraph G {
size ="4,4";
main [shape=box];
main -> parse [weight=8];
parse -> execute;
main -> init [style=dotted];
main -> cleanup;
execute -> { make_string; printf}
init -> make_string;
edge [color=red];
main -> printf [style=bold,label="100 times"];
make_string [label="make a\nstring"];
node [shape=box,style=filled,color=".7 .3 1.0"];
execute -> compare;
}
```

Figure 5.1: Sample graph file in Dot format.

Layout Tools

Graph layout tools constitute the core of Graphviz. The primary goal of the layout tools is to provide good diagrams of reasonable size graphs, scaling well to large graphs. Here, a reasonable size means no larger than what fits on a single screen or printed page with readable labels, which means around 50 or 100 nodes. For larger graphs, additional user interaction is often required. Graphviz layout tools are implemented as stream-oriented programs that read graphs, compute layouts and write the graphs in a graphics language. Graphviz currently provides 6 layout tools:

- **dot**: make hierarchical layouts of directed graphs, providing an assortment of shapes, styles and colors.
- **neato**: create layouts of undirected graphs that emphasize path distance and symmetry.
- **fdp**: provides an alternative layout to neato for undirected graphs.
- **sfdp**: a version of fdp that scales to very large graphs.
- **twopi**: creates a layout for radial graphs.
- **circo**: a tool appropriate to create circular graph layouts.

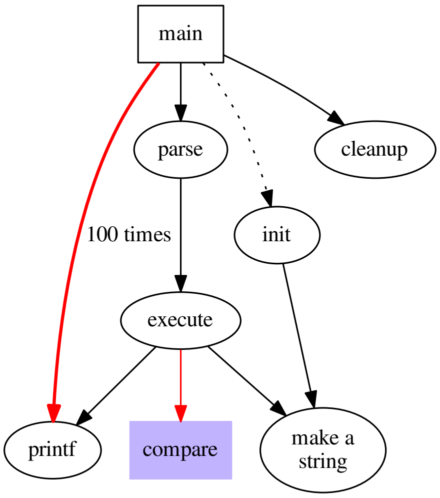


Figure 5.2: Drawing of sample graph file.

Graphical Tools

The primary interactive tool offered by Graphviz is Dotty, a browser that can display layouts and incorporate them in user interfaces for external programs. Dotty provides a WYSIWYG interface as well as a procedural one convenient for algorithmic operations.

Dotty is an application written in the Lefty graphical editor (Dobkin and Koutsofios 1991). Lefty programs are written in a scripting language similar to other conventional scripting languages (such as the UNIX shell). Lefty has string variables with runtime conversion for arithmetic, associative arrays, hierarchical namespaces for organizing code and data, and functions with arguments and local variables.

Graph Filters

As it was explained above, Graphviz makes strong emphasis on graph filters, providing different tools for this purpose (the following is not an exhaustive list):

- **`unflatten`**: unflatten is a preprocessor to `dot` that is used to improve the aspect ratio of graphs having many leaves or disconnected nodes. The usual layout for such a graph is generally very wide or tall.
- **`sccmap`**: decomposes directed graphs into strongly connected components and an auxiliary map of the relationship between components. In this map, each component is collapsed into a node.
- **`gvcolor`**: is a filter that sets node colors from initial seed values. Colors flow along edges from tail to head, and are averaged at nodes. The graph must already have been processed by `dot`. Appropriate choice of initial colors yields drawings in which node colors help to emphasize logical relationships between nodes.

5.2.2 Design and Implementation

Programming Languages

Graphviz is implemented in C language. Some code aspects of the core of Graphviz try to engineer around language features that C lacks but are found in more modern languages, in particular those related to object-oriented programming, such as inheritance, polymorphism and object initialization and finalization.

File Formats

As it was explained above, Graphviz defines its own file format for graph definition called Dot, which is used by the different applications provided in the package. From this Dot files, graph drawings can be generated in different image formats, such as PNG, PostScript (PS) or Portable Document Format (PDF).

License

Graphviz is distributed under the Eclipse Public License (EPL)[b]. EPL is an open-source software license designed to be business-friendly. The receiver of EPL-licensed programs can use, modify, copy and distribute the work and modified versions.

5.2.3 Application to Metabolic Network Visualization

In this thesis, we will apply Graphviz to plot metabolic networks. As it can be seen in Figure 5.1, a graph in Dot file format basically consists of a list of arcs between node pairs. In typical metabolic network representations, such as those used in Escher, nodes represent metabolites, and arcs represent biochemical reactions. Therefore, Dot file generation only requires the lists of arcs and metabolites to be represented and a simple script file to appropriately create the graph with the desired format, including node and arc shapes, sizes, colors, etc.

Appendix A describes the open-source software created for this thesis, including a brief description of the main available tools. Some of these tools are useful to create graphs in Dot format.

[b] http://www.eclipse.org/legal/epl-v10.html

5.3 Metabolic Network Reducing Algorithm

The size and complexity of metabolic network reconstructions have become larger and larger during the last years. Many stoichiometric methods cannot be applied to large networks, containing several thousand reactions. On the other hand, it is easier to study the basic principles of the metabolism of a given organism by focusing on smaller models representing their core parts (Erdrich et al. 2015). From the perspective of this study, this can also be useful to obtain simpler visualizations of metabolic networks. In Section 5.2.1 we explained that a reasonable graph size to ensure an understandable representation using Graphviz was around 50 or 100 nodes. However, human metabolic reconstructions may contain thousands of them. To solve this problem, manually generated network representations can be used. The elaboration of such representations can be assisted by computer programs, as it is the case of Escher. Alternatively, the simplified networks can be obtained in a faster and more systematic manner by means of network reducing algorithms.

Here we propose to apply a network reducing algorithm called *NetworkReducer* (Erdrich et al. 2015), whose goal is to obtain small models capturing central metabolism or other aspects of interest given the whole network model under study and a list of protected elements and functions.

5.3.1 The *NetworkReducer* Algorithm

NetworkReducer operates over metabolic network models such as those described in Chapter 4. For convenience in reading, here we reproduce some of the formulas that were already introduced there. In particular, the metabolic network models are composed of m metabolites, n reactions and a $m \times n$ stoichiometric matrix S. Given the flux vector $\mathbf{v}$, if the steady state assumption is made, we obtain the metabolite balancing equation:

$$\mathbf{S} \cdot \mathbf{v} = \mathbf{0} \tag{5.1}$$

The solutions $\mathbf{v}$ satisfying the previous equation form the null space of S whose dimension is given by the degrees of freedom (df):

$$\mathrm{df} = n - \mathrm{rank}(\mathbf{S}) \tag{5.2}$$

Additionally, the flux vector can be subject to lower and upper bound constraints:

$$\mathbf{l} \leq \mathbf{v} \leq \mathbf{u} \tag{5.3}$$

As it was explained in Chapter 4, FBA can be used to find the flux vectors satisfying the constraints given in Equations 5.1 and 5.3, and at the same time, optimizing a specific linear objective function:

$$\text{Maximize} \quad Z = \mathbf{c}^{\mathbf{T}} \cdot \mathbf{v} \tag{5.4}$$

where $\mathbf{c}^{\mathbf{T}}$ represents a transposed vector of coefficients and Z is typically the amount of biomass production or other product of interest.

NetworkReducer reduces the size of a large metabolic network to a smaller subnetwork while retaining certain desired features of the full network. Such features can be classified in five groups:

- **Protected metabolites** (P^M): P^M defines a list of metabolites that should be present in the resulting subnetwork.

- **Protected reactions** (P^R): the set of reactions P^R should be kept in the final subnetwork.
- **Protected phenotypes** (P^P): a set of inequalities describing specific phenotypes, $\mathbf{D}_k\mathbf{v} \leq \mathbf{d}_k$, where $k = 1...s$ and s is the number of phenotypes, see (Erdrich et al. 2015) for more details.
- **Minimum degrees of freedom** ($\mathrm{df}_{\mathrm{min}}$): the number of degrees of freedom given by Equation 5.2 of the final subnetwork should be above $\mathrm{df}_{\mathrm{min}}$.
- **Minimum number of reactions** (n_{min}): the algorithm ensures that the final number of reactions is not below n_{min}.

NetworkReducer starts with a preprocessing step that removes blocked reactions (those with zero flux) and checks feasibility of protected phenotypes. After that, the algorithm executes an iterative process where reactions are removed, checking that the resulting subnetwork retains the protected parts explained above. At each iteration, the algorithm applies FVA (see Section 4.5) to decide which reaction is to be removed. Erdrich et al. 2015 suggest to remove first those reactions with smallest fluxes, although alternative criteria can be applied. When the network pruning iterations finish, a final (and optional) step of network compression is executed, where reaction sets are represented as single overall reactions with collapsed stoichiometries. Algorithm 5.1 shows the pseudocode of the algorithm.

```
input    : $N$ (stoichiometric network), $P^R$ (protected reactions)
           $P^M$ (protected metabolites), $P^P$ (protected phenotypes)
           $\mathrm{df}_{\mathrm{min}}$ (minimum degrees of freedom), $n_{\mathrm{min}}$ (minimum number of reactions)
output   : $N'$ (reduced stoichiometric network)
auxiliar: $R$ (set of removable reactions), $F$ (set of flux ranges), success (boolean value)
begin
    $N'$ := network_preprocessing($N, P^M, P^R, P^P$)
    $R$ := get_reactions($N'$) $- P^R$
    while $\mathrm{df}(n) < \mathrm{df}_{\mathrm{min}}$ and length($R$) $\neq 0$ and get_num_react($N'$) $> n_{\mathrm{min}}$ do
        $F$ := FVA($N', R$)
        success := false
        while success = false and length($R$) $\neq 0$ do
            $r$ := get_candidate_reaction_for_removal($N', F, R$)
            $N'$ := remove_reaction($N', r$)
            $R := R - r$
            success := check_protected_functions($N', P^M, P^R, P^P$)
            if success = false then
                $N'$ := insert_reaction($N', r$)
    $N'$ := remove_unconnected_metabolites($N'$)
    $N'$ := compress_network($N'$)
```

Algorithm 5.1: Pseudocode for the *NetworkReducer* algorithm.

5.3.2 Application to Human Metabolism

Erdrich et al. 2015 applied their *NetworkReducer* algorithm to the iAF1260 model of Escherichia Coli presented in (Feist et al. 2007). The model is composed of 2 382 reactions and 1 682 metabolites. However, in this thesis we will work with genome-scale models of human metabolism, which are much more complex. Model size can be problematic when executing the algorithm due to the necessity of performing an FVA procedure at each iteration. Specifically, if the current metabolic network is composed of n reactions, then FVA should solve n optimizations. Moreover, if the optimization is formulated as an integer problem instead of a linear one, then the time complexity becomes even greater (see Section 4.3.2 for a more detailed explanation).

To address this issue, we propose a slight modification of *NetworkReducer* where the time cost of the FVA step is greatly decreased. For this purpose, at each iteration FVA is not applied over the whole set of removable reactions but over a randomly generated subset. The size of the subset is provided as an input parameter. The greater the subset, the closer the algorithm gets to the results of *NetworkReducer* but also the greater the time cost. We will refer to the new algorithm as the *SimplifiedNetworkReducer* algorithm, its pseudocode is shown in Algorithm 5.2. The key aspect of the algorithm is the `extract_subset_of_removable_reactions` function, that randomly selects a subset of the set of removable reactions at each iteration.

```
input   : $N$ (stoichiometric network), $P^R$ (protected reactions)
          $P^M$ (protected metabolites), $P^P$ (protected phenotypes)
          $\mathrm{df}_{\mathrm{min}}$ (minimum degrees of freedom), $n_{\mathrm{min}}$ (minimum number of reactions)
          $s$ (size of removable variable subset to be analyzed with FVA)
output  : $N'$ (reduced stoichiometric network)
auxiliar: $R$ (set of removable reactions), $R'$ (partial set of removable reactions)
          $F$ (set of flux ranges), success (boolean value)
begin
    $N' :=$ network_preprocessing$(N, P^M, P^R, P^P)$
    $R :=$ get_reactions$(N') - P^R$
    while $\mathrm{df}(n) < \mathrm{df}_{\mathrm{min}}$ and $length(R) \neq 0$ and $get\_num\_react(N') > n_{\mathrm{min}}$ do
        $R' =$ extract_subset_of_removable_reactions$(R, s)$
        $F :=$ FVA$(N', R')$
        success := false
        while success = false and $length(R') \neq 0$ do
            $r :=$ get_candidate_reaction_for_removal$(N', F, R')$
            $N' :=$ remove_reaction$(N', r)$
            $R := R - r$
            $R' := R' - r$
            success := check_protected_functions$(N', P^M, P^R, P^P)$
            if success = false then
                $N' :=$ insert_reaction$(N', r)$
    $N' :=$ remove_unconnected_metabolites$(N')$
    $N' :=$ compress_network$(N')$
```

Algorithm 5.2: Pseudocode for the *SimplifiedNetworkReducer* algorithm.

CHAPTER 6
PUBLIC DATABASES

THIS chapter is devoted to introduce two public databases relevant for the studies carried out in this MSc Thesis. On one hand, the Recon X database on human metabolism is described in Section 6.1. On the other hand, The Cancer Genome Atlas database containing diverse information related to cancer is presented in Section 6.2.

6.1 Recon X

Understanding the phenotypic behavior of biological organisms is an overly complex goal where the modeling of metabolic pathways plays a central role. Because of this, a high quality reconstruction of metabolism is greatly interesting for addressing a wide variety of scientific and applied questions about target living organisms, including contextualization of high-throughput data or the optimization of metabolic engineering procedures just to name a few.

Biochemistry has long been occupied with the reconstruction of metabolic pathways. However, it has been in the last decade when, by means of modern genome-sequencing capabilities, these pathway reconstructions have been increasingly integrated into genome-scale metabolic models. As a result, the field of genome-scale metabolic network analysis has expanded rapidly, and today more than fifty genome-scale metabolic reconstructions have been published.

The Recon X database provides a reconstruction of human metabolism using currently existing knowledge on metabolites and chemical reactions. In the following sections we provide a brief overview about this database and mention possible applications developed in the context of genome-scale metabolic network analysis.

6.1.1 Overview

The Recon X database[a] is an effort to reconstruct human metabolism. Recon X contains the metabolic information of Recon 2 (Thiele, Swainston, et al. 2013), the most comprehensive biochemical knowledge-base on human metabolism currently available.

Recon 2 is a consensus metabolic reconstruction integrating metabolic information from 5 different sources:

- Recon 1, a global human metabolic reconstruction (Duarte et al. 2007).
- EHMN, Edinburgh Human Metabolic Network (Hao et al. 2010).

[a] http://humanmetabolism.org/

- HepatoNet1, a liver metabolic reconstruction (Gille et al. 2010).
- Ac/FAO, a module on acylcarnitine/fatty acid oxidation metabolism (Sahoo, Franzson, et al. 2012).
- A human small intestinal enterocytes reconstruction (Sahoo and Thiele 2013).

In addition to this, Recon 2 has the following interesting features:

- Full semantic annotation, with references to persistent and publicly available chemical and gene databases.
- Access to database content both using the Recon X's webpage or using a downloadable file in SBML format.

Access from Webpage

Recon X webpage can be used to access information about metabolites and chemical reactions. Specifically, the database contains a total of 2 626 metabolites and 7 440 reactions. Information about metabolites comprises a list of metabolite names along with links to extended data for each one. If the user clicks on one of such links, it is possible to inspect the list of reactions where the given metabolite is involved, links to other databases, such as PubChem[b], or the Human Metabolome Database[c], etc.

Regarding the information about chemical reactions, again the webpage provides a list of reaction names with links to extended information, which in this case includes the stoichiometric formula with the involved metabolites, as well as other important fields, such as whether a gene to protein association exists for a given reaction.

Finally, the webpage also offers the possibility to search for specific metabolites or reactions by providing their identifiers or other special fields.

Access from File in SBML Format

Recon X also offers the possibility of downloading the entire database. Specifically, the database is provided in a single file in SBML format.

6.1.2 Applications

Since the first genome-scale reconstruction was published a decade ago, the availability and utility of genome-scale metabolic reconstructions have made a qualitative leap forward. Metabolic reconstruction have now been built for a wide variety of organisms and have been used toward five major ends (Oberhardt et al. 2009):

1. **Contextualization of high-throughput data**: with biology increasingly becoming a data-rich field, an emerging challenge has been determining how to organize, sort, interrelate, and contextualize all of the high-throughput datasets now available. This challenge has motivated the field of top-down systems biology, wherein statistical analyses of high-throughput data are used to infer biochemical network structures and functions. In top-down modeling, determination of network structure poses a major technological and computational hurdle. However, many of the

[b] http://pubchem.ncbi.nlm.nih.gov/

[c] http://www.hmdb.ca/

weaknesses of top-down modeling, such as lower accuracy and confidence in the resulting models, can be alleviated by comparison or merging with carefully built bottom-up models, such as genome-scale metabolic reconstructions.

2. **Guidance of metabolic engineering**: metabolic engineering involves the use of recombinant DNA technology to selectively alter cell metabolism and improve a targeted cellular function. Traditionally, metabolic engineering has been performed on a small scale through manipulation of a few genes to affect yield of a target metabolite. The use of genome-scale metabolic reconstructions represents a major evolution for the field, wherein whole-cell networks and systems-level analyses are for the first time being leveled to determine optimal engineering strategies on a whole-cell basis.

3. **Directing hypothesis-driven discovery**: Much of what is known in biology today is the result of meticulous, hypothesis-driven discovery. With the tremendous expansion of biological data in recent years, the need has arisen for new method development to integrate high-throughput data with the biological discovery process. Genome-scale metabolic reconstructions represent concise collections of existing hypotheses, and taken together as a broad context they enable systematic identification of new hypotheses that can be tested and resolved.

4. **Interrogation of multi-species relationships**: metagenomics studies particularly have shown most ecosystems to be extremely diverse, including up to thousands of distinct taxa. Genome-scale metabolic reconstructions are increasingly being applied to these multi-cell problems, as well as to the study of functional differences between species.

5. **Network property discovery**: genome-scale metabolic reconstructions have enabled analysis of emergent phenomena through a focus on whole networks rather than individual pathways or genes, and many computational techniques have been developed to probe network properties. These types of network-level analyses will be critical to fully unravel the complex genotype-phenotype relationships in cells.

6.2 The Cancer Genome Atlas

Cancer is considered the most complex disease that mankind has to face. There are at least 200 forms of cancer and many more subtypes. Cancer is caused by errors in DNA that make cells grow uncontrolled. Identifying the genomic changes for each type of cancer and understanding the interactions between these changes will provide the foundation for improving cancer prevention, early detection and treatment.

In 2005, The Cancer Genome Atlas (TCGA) (NCI and NHGRI 2005a) was launched as a major effort to accelerate the comprehensive understanding of the genetics of cancer using innovative genome analysis technologies. TCGA is a public funded project launched by the National Institute of Health (NIH) that aims to catalogue and discover major cancer-causing alterations in large cohorts of human tumors through large-scale genome sequencing and integrated multi-dimensional analyses. According to (Tomczak et al. 2015), Phase I of the project (a 3-year pilot study) aimed to develop and test the research infrastructure based on the characterization of chosen tumors having poor prognosis: brain, lung, and ovarian cancers. Phase II started in 2009 expanding previous analyses to additional types reaching 30 different tumor types analyzed by 2014. The TCGA project engaged scientists and managers from NIH's National Cancer Institute (NCI) and National Human Genome Research Institute (NHGRI) funded by the US government, as well as cooperating with institutions across the USA and Europe.

6.2.1 Overview

The main goal of TCGA is to improve our ability to diagnose, treat and prevent cancer. As reported in (NCI and NHGRI 2005c), TCGA has generated comprehensive, multi-dimensional maps of the key genomic changes in 33 types of cancer. TCGA dataset currently stores 2.5 petabytes of data describing tumor tissue and matched normal tissues from more than 11,000 patients. This information is publicly available and has been used widely by the research community. In addition to this, it has been estimated that the available data has contributed to more than a thousand cancer studies by independent researchers.

A general description about the main components of TCGA is provided in (NCI and NHGRI 2005b):

1. **Tissue processing**: cancer patients are asked to donate a portion of tumor tissue that has been removed as part of their cancer treatment along with a sample of normal tissue. Both samples are referred to as biospecimens. After biospecimen samples have been extracted, the TCGA Biospecimen Core Resources process the samples to ensure if they meet the stringent set of criteria that is required to enable their use for research purposes. In addition to this, biospecimens are coded so as to remove any information that might connect a sample with private information of a patient.

2. **Research and discovery**: researchers of the TCGA Genome Characterization Center analyze tumor and normal tissue from hundreds of patients for each cancer selected for study, providing the statistical power to produce a complete genomic profile of each cancer type. Some of the aspects studied during sample analysis include how the genome is rearranged or how gene expression changes in tumors compared to normal cells. Because of this, the application of DNA sequencing techniques conducted by the High-throughput TCGA Genome Sequencing Centers plays a major role in the performed research tasks. Overall, TCGA have analyzed thousands of biospecimen samples, integrating the data across the different tumor types.

3. **Data sharing**: all the generated information about biospecimen samples is entered by TCGA Data Coordinating Center into public databases as it becomes available.

4. **Community research and discovery**: scientists from the broader cancer research community are able to search, download and analyze the different datasets created by TCGA, allowing TCGA data to have a multiplier effect on the scope and quality of cancer research.

6.2.2 Applications

TCGA has created a genomic data analysis pipeline able to collect, select, and analyze human tissues for genomic alterations on a very large scale. The main applications and findings of the initiative are related to cancer research (NCI and NHGRI 2005c):

- **Molecular basis of cancer**: TCGA has improved our understanding of the genomic underpinnings of cancer. One example of this can be found in a TCGA study that established a link at the molecular level between a subtype of breast cancer and another subtype of ovarian cancer. Since both subtypes seem to follow a common path of development, they may respond to similar therapeutic strategies.

- **Tumor subtypes**: TCGA has greatly contributed to a new perspective on cancer classification, identifying tumor subtypes with different sets of genomic alterations.

- **Therapeutic targets**: the initiative has identified genomic features of tumors that can be targeted with currently available therapies or can be used as the basis for new drug development. For instance, genomic alterations in lung squamous cell carcinoma were studied and identified by TCGA. The findings of this research will be used to define the specific treatment provided to patients.

Chapter 7
Experiments

THE main goal of this work is to study human metabolism, and in particular the metabolism of cancer, under a systems biology perspective. For this purpose it is necessary to integrate transcriptomic and metabolic data using existing databases and to analyze the resulting information by means of appropriate mathematics and bioinformatics techniques. Previous chapters have been devoted to explain the different elements involved in the process, and at this point we are prepared to present the empirical results of the study. This chapter is organized as follows: Sections 7.1 and 7.2 list the datasets and methods, respectively, used in the experiments. Section 7.3 enumerates the software tools required to obtain the results, including solvers, visualization programs and the open-source software that has been developed for this thesis. Finally, Section 7.4 shows the experiment results.

7.1 Datasets

The basis of this work is the integration of metabolic and transcriptomic data extracted from public databases. In particular, the following two datasets were used:

- **Recon 2**: Recon 2 is the second version of the human metabolic reconstruction provided in the Recon X database (see Section 6.1). Recon 2 contains 7 440 reactions and 5 063 metabolites.

- **Kidney renal clear cell carcinoma (KIRC) data**: the KIRC data collection[a] is part of the TCGA database (see Section 6.2) whose aim is to accelerate the comprehensive understanding of the genetics of cancer using innovative genome analysis technologies. KIRC data contains diverse information about 537 sample of kidney tissue, including RNA-Seq data. From the available data, we selected 60 samples coming from healthy cells and another 60 from cancerous cells so as to obtain a balanced experiment design. In addition to this, all of the samples were sequenced at the same center[b], avoiding possible batch effects.

[a] https://gdc-portal.nci.nih.gov/projects/TCGA-KIRC

[b] This and other relevant information is present in the TCGA barcodes associated to biospecimen data (see https://wiki.nci.nih.gov/display/TCGA/TCGA+barcode for more details).

7.2 Methods

To obtain the experiment results reported in this chapter the following methods were necessary:

- **Flux balance analysis (FBA)**: FBA constitutes the basic technique used in this work to study human metabolism (for more details, see Section 4.4). FBA uses mathematical optimization, and in particular, linear programming (LP), to obtain its results.

- **Flux variability analysis (FVA)**: FVA allow us to assess which fluxes have a higher importance within a metabolic network (additional details are explained in Section 4.5). FVA also plays an important role in the network reducing algorithm applied here.

- **Kernel density estimation (KDE)**: KDE was applied to carry out a descriptive study of the RNA-Seq data. In particular, the binary logarithm of RPKMs was represented. The purpose of this representation was to verify the empirical findings shown in (Hebenstreit et al. 2011), were the data was grouped in two gaussian distribution associated to highly and lowly expressed genes (see Section 4.6.3).

- **Gaussian mixture model estimation**: this method was used to estimate the parameters of a mixture of two gaussians distributions from the binary logarithm of RPKMs observed in the RNA-Seq data. Once the model was estimated, it was used to classify the genes as lowly or highly expressed (refer to Section 4.6.3 for additional details).

- **Tissue-specific FBA**: Tissue-specific FBA allows us to study the metabolism of kidney cells. This technique is based on mathematical optimization as well as conventional FBA. However, in this case, mixed integer programming (MIP) problems are to be solved (see Section 4.6). This method requires lists of lowly and highly expressed genes as input data.

- **Mann-Whitney's U-test**: the Mann-Whitney's U-test (Mann and Whitney 1947) is a non-parametric statistical test that is used to decide whether two independent samples of observations come from the same population. Due to its non-parametric nature, the U-test does not require that the observed data come from a specific distribution, in contrast to the Student's t-test, which is used for the same purpose but assumes data normality.

- **Benjamini-Hochberg procedure**: a multiple testing correction procedure is needed to adjust our statistical confidence measures when performing a large number of statistical tests. The Benjamini-Hochberg procedure (Benjamini and Hochberg 1995) allows to control the false discovery rate (FDR). The FDR focuses on controlling the rate of type I errors that occur when performing a set of hypothesis tests.

- **Network reducer algorithm**: this method is used to obtain smaller metabolic networks where the results of other methods are easier to analyze. The network reducer algorithm used in this work internally uses FVA as a criterion to discard network elements (see Section 5.3 for additional details).

7.3 Software Tools

In this section we enumerate the main software tools used to obtain the empirical results, including mathematical solvers, tools for metabolic network visualization and statistical packages. Additionally, a significant effort has been put in the development of software specific for this work.

7.3.1 Solvers

We used the standalone solvers available for the three software packages described in Section 4.7:

- **GLPSOL**: GLPSOL is the solver incorporated in the GLPK. GLPSOL solves LP and MIP problems (version 4.57 was used).
- **CLP**: it is the LP solver included in the COIN-OR project (version 1.16.6 was used).
- **CBC**: COIN-OR also incorporates CBC, a MIP solver (version 2.9.5 was used).
- **CPLEX**: CPLEX is a commercial LP and MIP solver provided by IBM (version 12.6.2.0 was used).

7.3.2 Visualization Tools

Chapter 5 introduced the two visualization tools used in this work:

- **Escher**: this tool allows to visualize the results of FBA techniques over manually-generated metabolic network representations. In particular, the metabolic maps for Recon 1 will be used (see Section 5.1.3 for more details about the available maps).
- **Graphviz**: Graphviz is a general purpose graph visualization tool, which provides a greater versatility with respect to Escher at the cost of additional effort required from the user to obtain the desired results (version 2.38.0 was used).

7.3.3 Statistical Tools

Statistical techniques were also necessary in the proposed experimentation as it was mentioned in Section 7.2. The following statistical packages were used:

- **scikit-learn**: the scikit-learn Python module[c] provides a set of tools for data analysis. In particular, we used the KDE and gaussian mixture model estimation functionality provided by the package (version 0.17.1 was used).
- **scipy**: scipy[d] is a Python module for mathematics, science and engineering. It provides an implementation of the Mann-Whitney U-test (version 0.17.1 was used).
- **statsmodels**: statsmodel[e] is a Python statistics package that we have used to conduct statistical tests. Specifically, we used its implementation of the Benjamini-Hochberg procedure (version 0.8.0rc1 was used).

[c] http://scikit-learn.org

[d] https://www.scipy.org/

[e] http://statsmodels.sourceforge.net/

7.3.4 Software Developed for this Thesis

The different tools described above not always offer full or even partial implementations of all of the methods used in this thesis (for instance, the FBA method uses mathematical solvers to find solutions for LP problems, however this is only a part of the tasks to be executed when applying FBA). In addition to this, the results reported in this chapter often requires the combination of different tools and methods. To fill these gaps in the available tools, we have developed an open-source software toolkit that is described in Appendix A. The name of the toolkit is *Flux Capacitor* or `fcap`.

7.4 Results

After introducing the datasets, methods and software tools used in the experiments, we are ready to show the obtained results.

In some experiments, the time cost of specific algorithms or processes is reported. In all cases such time cost was measured on a computing cluster composed of nodes integrated by two Intel Xeon E5-2450 processors with 8 cores and 4GB RAM per each core. Typically, computations were done in individual cores except for those situations in which parallelism can be exploited.

7.4.1 Comparison between Solvers

As it was explained in Chapter 4, the methods used in this work to study human metabolism are based on mathematical optimization techniques, and more specifically on the use of LP and MIP solvers. In this section we study the differences in efficiency of some available options. Efficiency is a key aspect from the perspective of this work, because two of the techniques applied here require to intensively solve LP and MIP problems. These two techniques are FVA and the *NetworkReducer* algorithm (indeed, *NetworkReducer* internally executes multiple instances of FVA).

Efficiency Solving LP Problems

As it was previously stated, LP problems can be efficiently solved using the simplex algorithm. FBA involves solving individual LP problems. Table 7.1 shows the time in seconds required to maximize the biomass function defined in the Recon 2 human metabolic network reconstruction for three different solvers: GLPSOL, CLP and CPLEX. Only one core were used to perform the calculations. As it can be seen, the three solvers were able to find the solution in less than a second, being GLPSOL the slowest and CPLEX the fastest one. However, the differences were negligible.

Efficiency Solving MIP Problems

Solving MIP problems is more computationally demanding than solving LP problems (see Section 4.3.2 for more details). The tissue-specific FBA technique described in Section 4.6 involves solving a MIP problem for each RNA-Seq sample. Table 7.2 shows the average cost when solving 120 MIP problems for the TCGA's KIRC samples selected for this study (60 for healthy cells and 60 for cancerous cells). RNA-Seq data was combined with the Recon 2 metabolic network reconstruction. GLPSOL, CBC and CPLEX solvers

Table 7.1: Time in seconds required to maximize the biomass function for Recon 2 metabolic reconstruction when using GLPSOL, CLP and CPLEX. Calculations were executed in 1 core.

	Time (s)
GLPSOL	0.72
CLP	0.23
CPLEX	0.17

Table 7.2: Average time in seconds required by GLPSOL, CBC and CPLEX to solve 120 MIP problems for the TCGA's RNA-Seq KIRC samples chosen for this work. Recon 2 metabolic reconstruction was used. Computations were carried out in 8 cores.

	Avg. Time (s)
GLPSOL	N/A
CBC	810
CPLEX	11

were used. Computations were carried in a whole cluster node with 8 cores, taking advantage of the parallel execution capabilities of those solvers which have them, in particular, CBC[f] and CPLEX.

As it can be seen in Table 7.2, CPLEX was substantially faster than CBC when solving the MIP problems. No data is reported for GLPSOL since it was not able to find the solution in most cases. This is in line with the findings shown in (Meindl and Templ 2013), where the commercial solvers are by far the most efficient and the free ones have difficulties to solve some kinds of problems. However, in our case CBC worked well although slower than CPLEX.

Time cost of MIP solving can be reduced if the optimization is suboptimal. One possibility is to stop the process when the difference (or gap) between the best solution found and the bound for the optimal solution falls below a certain percentage of that bound. This percentage is referred to as the *MIP gap tolerance* parameter by CPLEX, and it is also available for the CBC solver. Table 7.3 shows the average time in seconds required to solve 120 MIP problems resulting from the integration of KIRC and Recon 2 data. In this case, the CBC and CPLEX solvers were used with a gap tolerance equal to 0.01. Additionally, the average percentage of optimality of the solutions is also reported. Optimality is defined here as the best solution obtained by the suboptimal solver divided by the best one when using the solver without any gap tolerance parameter (an optimality of 100% means that the optimal solution was found). As it is shown in the table, both solvers substantially reduced their time cost while retaining almost a 100% optimality.

For the rest of the experiments, we will use CPLEX as our LP and MIP solver due to its greater efficiency. However, it has been demonstrated that CBC is also a good option and has one great advantage with respect to CPLEX. In particular, CBC has a much less restrictive license, which can be interesting for certain uses.

[f] To enable CBC's parallel mode it was necessary to use the `configure` option `--enable-cbc -parallel` when building the package.

Table 7.3: Average execution time in seconds and percentage of optimality obtained by CBC and CPLEX when solving 120 MIP problems for the integration of KIRC RNA-Seq data the Recon 2 metabolic reconstruction. Gap tolerance was equal to 0.01. Computations were made in 8 cores.

	Avg. Time (s)	**Optimality (%)**
CBC	30	99.8
CPLEX	2	99.3

7.4.2 Recon 2 Biomass Optimization

Recon 2 metabolic reconstruction used in this work provides a biomass function that can be used to apply FBA to human metabolism. After this, it is possible to study the network robustness by means of FVA.

The objective function obtained when applying FBA to the Recon 2 metabolic model was equal to 3.20. Regarding the FVA results, we computed the flux ranges using 0.9 as the value of the γ parameter and generated a box plot for them that is shown in Figure 7.1. The first quartile of the flux range data was equal to zero, which means that a 25% of the fluxes cannot change their value in optimal biomass production mode. However, the median value for the flux ranges was greater than 500 (the exact value was 635.9). Taking into account that the maximum value for the flux ranges is $2\,000$[g], this observation suggests that the network has a high robustness degree.

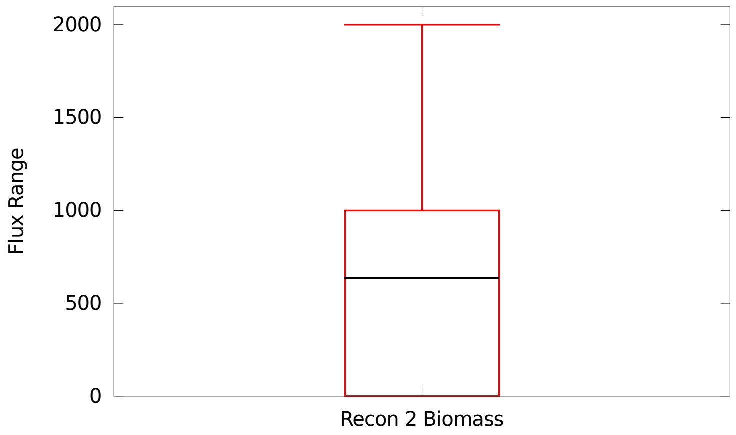


Figure 7.1: Boxplot for Recon 2 flux ranges when optimizing the biomass function. The γ parameter was equal to 0.9.

Due to the great computational cost of FVA, we also measured the time in seconds required to carry out the computations. Due to the fact that FVA is an embarrassingly parallel problem, the measurements were made using different numbers of computing nodes. Table 7.4 shows the time cost in seconds and the speedup achieved when applying FVA to the Recon 2 metabolic model with biomass production as the objective function. The process was executed in 1, 2 and 4 computer nodes composed of 8 cores. As it can be seen, FVA required more than two hours when executed in one computing node. Parallelism

[g] The maximum value for the flux upper bounds in Recon 2 is $1\,000$. and the minimum value for the lower bounds is $-1\,000$.

Table 7.4: Execution time in seconds and speedup when applying FVA to the Recon 2 metabolic network model using biomass production as the objective function. The γ parameter was equal to 0.9. Computing nodes composed of 8 cores were used.

	Time (s)	**Speedup**
FVA (1 node)	9 480	1.0
FVA (2 nodes)	7 140	1.3
FVA (4 nodes)	4 140	2.3

allowed us to reduce the time cost, but the speedup was always below the number of computing nodes. We think that this is due to the time spent by our implementation reading and writing files (files in lp format should be loaded in memory and solution files are stored on disk). This part of the process cannot be parallelized and had a non-negligible time cost when compared to that of solving the LP problems using CPLEX.

7.4.3 Determining Absent and Present Genes

As it was explained in Section 4.6.3, obtaining the set of lowly and highly expressed reactions when working with RNA-Seq data first requires to decide whether each gene is absent or present according to its abundance. For this purpose, we follow the technique proposed in (Hebenstreit et al. 2011), which is based on the empirical observation of two main RNA abundance classes. These two abundance classes can be characterized by means of a gaussian mixture model of two components. After estimating the parameters of the mixture model, it can be used to classify the genes as absent or present.

The `fcap` toolkit developed for this thesis implements the required code to determine whether each gene is absent or present. However, it is important to first verify that the two RNA abundance classes reported in (Hebenstreit et al. 2011) can also be observed here.

Figure 7.2 shows a KDE obtained from the binary logarithm of the RPKMs for the TCGA's KIRC data. Zero counts were excluded from the representation. The results are very similar to those reported in (Hebenstreit et al. 2011), where two overlapping components were identified: one associated to present genes centered at 4 approximately, and another one at the left of it associated to absent genes. For our data, the estimation of a two component gaussian mixture model determined that the group of present genes was centered at 3.0, and that of absent genes at -1.6.

7.4.4 Tissue-Specific FBA of an Individual Sample

After verifying the correctness of the technique to identify absent and present genes, we conducted an exploratory tissue-specific FBA on a single sample. The barcode of such sample was `TCGA.A3.3324.01A.02R.1325.07`. However, it is important to stress out that experiments with other samples produced very similar results to those reported below.

Statistics about Genes and Reactions

First, we obtained the set of absent and present genes. Figure 7.3 shows some statistics about the number of genes contained in the KIRC and Recon 2 data. As it can be seen, Recon 2 contains much less genes than KIRC. On the other hand, there were more present genes than absent. This is in line with the densities shown in Figure 7.2 for the two RNA abundance classes.

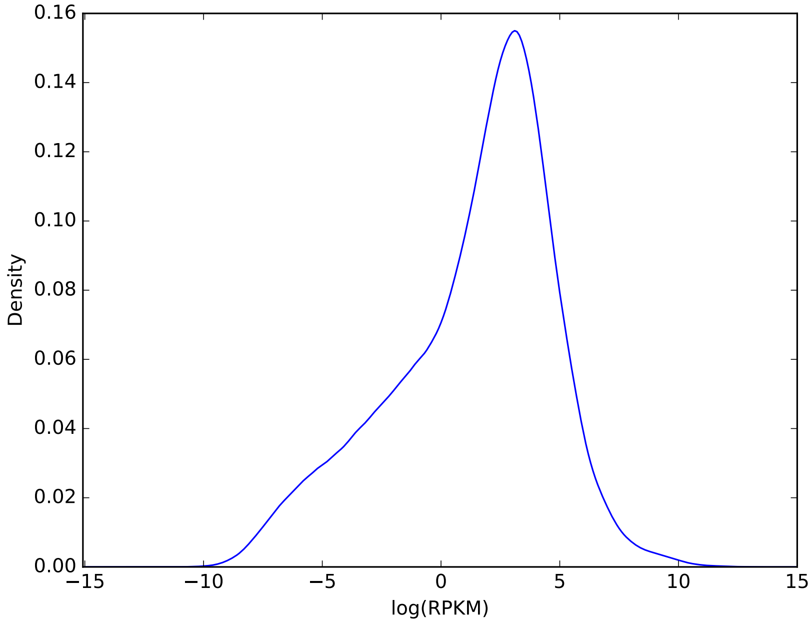


Figure 7.2: KDE of the binary logarithm of the RPKMs for the KIRC data. Genes with zero counts were excluded.

Table 7.5: Solution statistics after the application of tissue-specific FBA to KIRC's sample `TCGA.A3.3324.01A.02R.1325.07`. The gap tolerance was set to 0.01.

Objective function value	3 127
Solver effectiveness (%)	73.7

After obtaining the absent/present genes, we applied the gene to protein-reaction rules contained in Recon 2, determining the sets of lowly and highly expressed reactions for the specific sample. Figure 7.4 shows the main reaction statistics. As it can be seen, the highly expressed reactions outnumbered the lowly expressed ones.

Tissue-Specific FBA Results

We used the lists of lowly and highly expressed reactions to perform tissue-specific FBA. Table 7.5 shows the main statistics about the solution. As it can be seen, the value of the objective function was equal to 3 127. This means that 3 127 of the reactions identified as lowly or highly expressed retained this condition in the solution found by CPLEX. This number divided by the sum of the sizes of the lower and highly expressed reactions sets was used to compute a measure of the effectiveness of the solver. In this case, CPLEX achieved an effectiveness equal to 73.7%. The remaining 26.3% of the reactions changed its expression status in the final solution. Different explanations can be proposed for this observation. Shlomi et al. 2008 suggest that this kind of changes can be due to post-transcriptional regulatory effects. However, it is also possible that some reactions were incorrectly classified as lowly or highly expressed and the solver changed their status in the solution so as to avoid model inconsistencies.

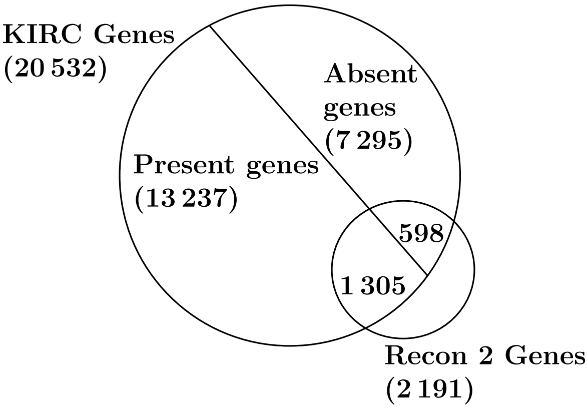


Figure 7.3: Basic statistics about genes in Recon 2 metabolic network and TCGA KIRC's sample `TCGA.A3.3324.01A.02R.1325.07`.

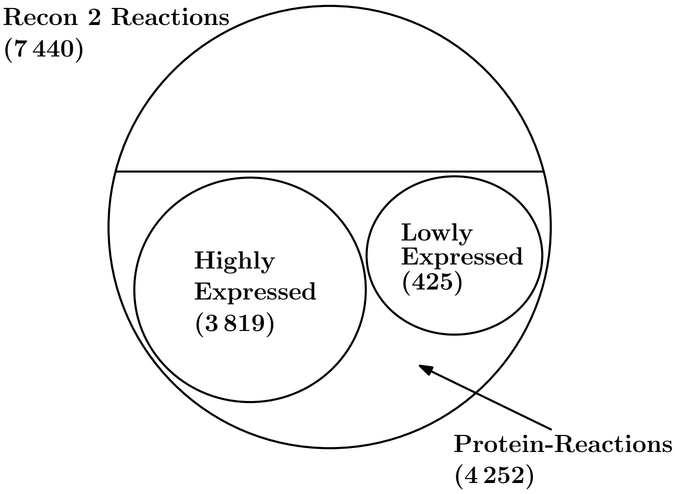


Figure 7.4: Reaction statistics about genes regarding the application of tissue-specific FBA to KIRC's sample `TCGA.A3.3324.01A.02R.1325.07`.

The fluxes found by CPLEX when obtaining the optimal solution can be visualized by means of the Escher tool. For this purpose, it was necessary to convert the fluxes calculated by CPLEX into JSON format. After that, the JSON file can be loaded with Escher after selecting the appropriate map (see Section 5.1 for additional details). Figure 7.5 shows a representation of the fluxes associated to the tricarboxylic acid (TCA) cycle (also known as the Krebs cycle) for the KIRC sample under study. Positive fluxes are shown in red color, while negative ones are shown in blue.

Network Robustness

After performing the tissue-specific FBA procedure, we studied the robustness of the network by means of FVA. Figure 7.6 shows a boxplot for the flux ranges. The γ parameter was set to 0.9, and the gap tolerance to 0.01. As it can be seen, the results and conclusions to be extracted are quite similar to those obtained when optimizing the Recon 2 biomass function (see Figure 7.1). Specifically, the network showed a high degree of robustness.

On the other hand, FVA was even more computationally demanding in this case than it was when optimizing the Recon 2 biomass function, since it involves solving thousands of MIP problems. To accelerate the calculations, we used previously obtained solutions

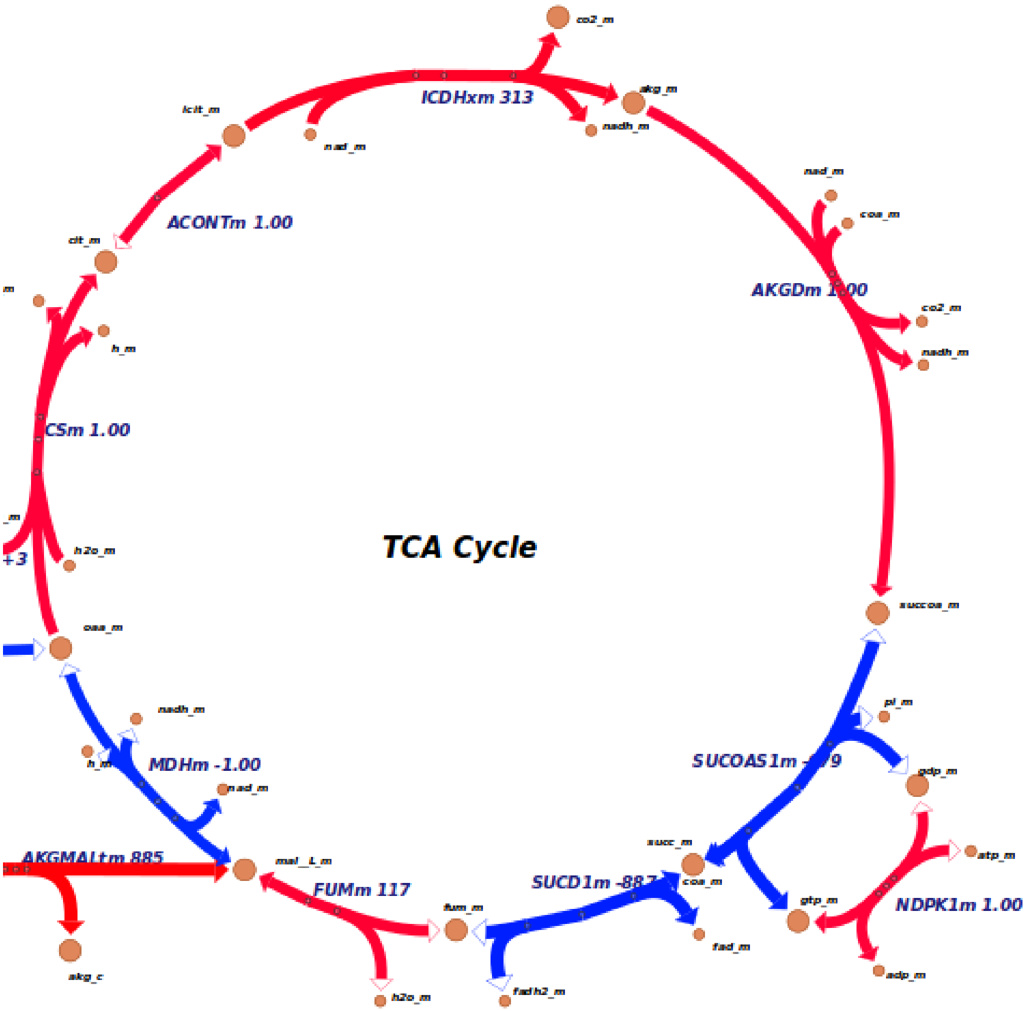


Figure 7.5: Escher representation of the metabolic fluxes for KIRC sample with code TCGA.A3.3324.01A.02R.1325.07. The TCA cycle is shown. Positive fluxes are shown in red and negative ones in blue.

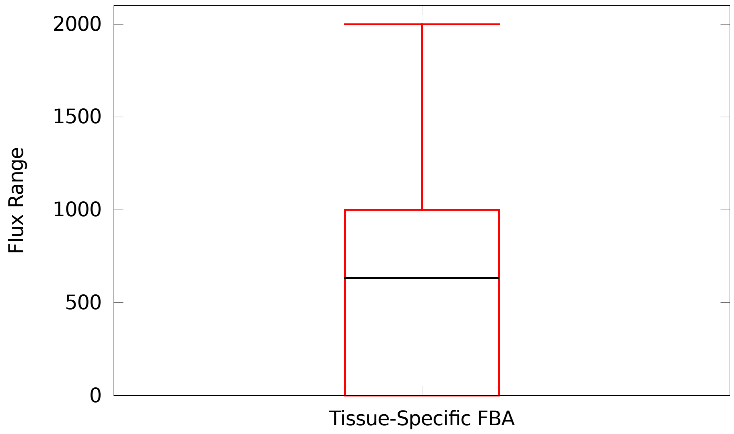


Figure 7.6: Boxplot for flux ranges when applying tissue-specific FBA over KIRC sample TCGA.A3.3324.01A.02R.1325.07 and Recon 2 metabolic model. The γ parameter was equal to 0.9 and the gap tolerance was set to 0.01.

to initialize the solver[h] as it is proposed in (Gudmundsson and Thiele 2010) (see Section 4.5.2 for more details). Table 7.6 shows the time cost in seconds of the algorithm with $\gamma = 0.9$ and gap tolerance equal to 0.01. The table also shows the speedup for the execution in 1, 2 and 4 computing nodes. For this experiment, the nodes were composed of only 2 cores instead of 8, since the memory requirements of the solver were greater for MIP problems. As it can be observed in the table, the time cost was substantially higher than that required to perform FVA with the Recon 2 biomass function (see Table 7.4). By contrast, the use of parallelism allowed us to obtain higher speedups. The reason for this is that the time required for reading and writing files was lower in relation to the cost of solving the MIP problems.

Table 7.6: Execution time in seconds and speedup when applying FVA tissue-specific FBA over KIRC sample `TCGA.A3.3324.01A.02R.1325.07` and Recon 2 metabolic model. The value of the γ parameter was 0.9 and the gap tolerance was set to 0.01. Computations were made in computer nodes composed of 2 cores.

	Time (s)	**Speedup**
FVA (1 node)	29 823	1.0
FVA (2 nodes)	18 233	1.6
FVA (4 nodes)	9 352	3.2

7.4.5 Tissue-Specific FBA of Multiple Samples

After reporting the tissue-specific FBA results for a single sample, we applied the technique to the whole set of samples. The goal of the analysis was to study the differences in metabolic behavior for healthy and cancerous cells. For this purpose, we designed a differential reaction expression experiment.

Experiment Design

There are two questions that should be answered so as to conduct the differential reaction expression experiment. The first one is how do we measure reaction expression for each sample. This decision is directly related to the second question. In particular, we should also decide which hypothesis test will be appropriate to discover whether there are statistically significant differences between the measurements associated to the healthy cells and those of the cancerous cells. The election of the hypothesis test will be influenced by the nature of the measure used to represent reaction expression.

One straightforward approach to measure reaction expression for each sample is to simply use the values of the fluxes that maximize the objective function of the tissue-specific FBA procedure. However, the individual sample results reported above, and more specifically the distribution of flux ranges shown in Figure 7.6, advise against this strategy. As it can be observed in the figure, the median flux range while maintaining the same value of the objective function is higher than 600, given a theoretical maximum of $2\,000$. This moves us to think that the flux magnitude is not an appropriate aspect to pay

[h] For this purpose, it is necessary to write the so-called *MIP start file* using the CPLEX option `write`. Such file is later read by the solver before processing the new MIP problem. In addition to this, enabling the so-called *polishing* heuristic provided by CPLEX was also useful to reduce the time cost.

attention to, due to its great variability. Instead, we propose to classify the fluxes into three categories: inactive, active in the direct sense and active in the reverse sense. This categorization allows us to work with more stable values while still capturing the most essential information about the reactions. The categorization strategy can be based on the thresholds imposed to the fluxes by the tissue-specific FBA technique (see Section 4.6.1). In particular, a reaction was considered active if it carries a flux above a certain positive threshold ϵ (active in the direct sense) or below a negative threshold $-\epsilon$ (active in the reverse sense), otherwise it is considered inactive. Following this criterion, we propose to represent reverse active, inactive and direct active reactions using the numbers -1, 0 and 1, respectively.

As it was explained in Chapter 3, the well known Student's t-test is typically used when analyzing differential gene expression. The t-test assumes that the observed variable is a normally distributed interval variable[i]. This assumption cannot be justified in our case, since the representation of the metabolic information we propose is ordinal, i.e. it is a categorical representation where the categories can be ordered. One hypothesis test that can be applied under these circumstances is the Mann-Whitney U-test, which is a non-parametric alternative to the t-test where the only assumption about the observed variable is that it is at least ordinal.

Differential Reaction Expression Analysis

The 120 samples of the TCGA's KIRC dataset selected for this work (60 cases and 60 controls) were integrated with the Recon 2 metabolic reconstruction so as to perform tissue-specific FBA (again, a gap tolerance equal to 0.01 was used to solve the MIP problems). After calculating the flux values for all of the reactions, they were transformed following the ordinal representation described above. The ordinal values were used to conduct a Mann-Whitney U-test for each reaction, testing the null hypothesis that there is no difference in reaction expression between healthy and cancerous cells. Afterwards, the p-values returned by the U-test were corrected using the Benjamini-Hochberg procedure.

Table 7.7 shows the number of null hypotheses rejected by the hypothesis tests, with and without p-value correction. As it can be seen, the Benjamini-Hochberg procedure produces a significant reduction in the number of null hypothesis rejections. On the other hand, Table 7.8 shows the five reactions with the lowest corrected p-value.

Table 7.7: Number of rejected null hypotheses (H_0) for the differential reaction expression experiment. Mann-Whitney U-test with and without Benjamini-Hochberg's (BH) p-value correction was used. The value of α was 0.05.

	H_0 **Rejections**
U**-test**	835
U**-test + BH**	428

Analyzing lists of differentially expressed reactions can be tedious due to the lack of contextual information. One way to tackle this problem is by means of the graphical representations of metabolism provided by Escher. In spite of the fact that Escher is intended to be used for metabolic flux visualization, it is straightforward to adapt it for visualizing p-values instead of fluxes, using specific colors to highlight differentially expressed

[i] An interval variable is a measurement where the difference between two values is meaningful.

Table 7.8: List of the five reactions with lowest corrected p-value for the differential reaction expression experiment.

Reaction	**p-value**
NCCt	$7.8 \cdot 10^{-21}$
DCT	$3.7 \cdot 10^{-20}$
DM_melanin_c_	$3.7 \cdot 10^{-20}$
DOPAQNISO1	$3.7 \cdot 10^{-20}$
TYRASE	$3.7 \cdot 10^{-20}$

reactions. To do this, flux values are replaced by p-values in the JSON file loaded by Escher. After loading the file, the tool can be configured to display a specific color for those reactions with small p-values. Figure 7.7 shows an example where the Escher's retinol metabolism diagram highlights the differentially expressed reactions using red color.

As it was explained in Section 5.1.3, Escher provides five different metabolic maps related to human metabolism. These maps can be used to carry out a brief analysis of metabolic subsystems presenting differentially expressed reactions:

- **Amino acid metabolism**: when displaying corrected p-values with Escher, differentially expressed reactions can be found in the following subunits: glycine, serine and theronine metabolism (e.g. SERHL, p-value=$1.8 \cdot 10^{-19}$), methionine and cysteine metabolism (e.g. 2AMACSULT, p-value=$9.5 \cdot 10^{-12}$) and finally, the urea cycle (e.g. CBPSAM, p-value=$1.6 \cdot 10^{-5}$). By contrast, the following subunits showed no statistically significant differences: valine, leucine and isoleucine degradation, collagen degradation and lysine degradation.

- **Carbohydrate metabolism**: for this map, the following subsystems showed differentially expressed reactions: glyoxilate metabolism (e.g. HPYRtp, pvalue=p-value=$1.0 \cdot 10^{-9}$), pentose-phosphate pathway (e.g. PPM, p-value=$9.4 \cdot 10^{-5}$) ketogenesis (e.g. BDHm, p-value=$1.6 \cdot 10^{-5}$), TCA cycle (e.g. SUCOAS1m, p-value=$3.3 \cdot 10^{-10}$), polysaccharide degradation (e.g. TREHe, p-value=$3.6 \cdot 10^{-6}$), and itaconate and mesaconate metabolism (e.g. ITCOAL1m, p-value=0.029). Other parts did not show statistically significant differences in reaction expression, including metabolism of galactose, ascorbate, fructose, mannose, fucose, aminosugars, nucleotide sugar, propanoate, inositol phosphate, glycogen and starch. Pentose and glucoronate interconversions did not show differentially expressed reactions either.

- **Glycolisis**: this pathway is a subset of the carbohydrate metabolism.

- **Inositol retinol metabolism**: retinol metabolism presented differentially expressed reactions (e.g. LRAT, p-value=0.0057) while inositol phosphate metabolism had not any.

- **Tryptophan metabolism**: there were no differentially expressed reactions in tryptophan metabolism.

A detailed study about the whole set of differentially expressed reactions that were identified is beyond the scope of this work. However, it can be interesting to discuss a bit more the results related to retinol metabolism, due its relationship with cell differentiation and carcinogenesis.

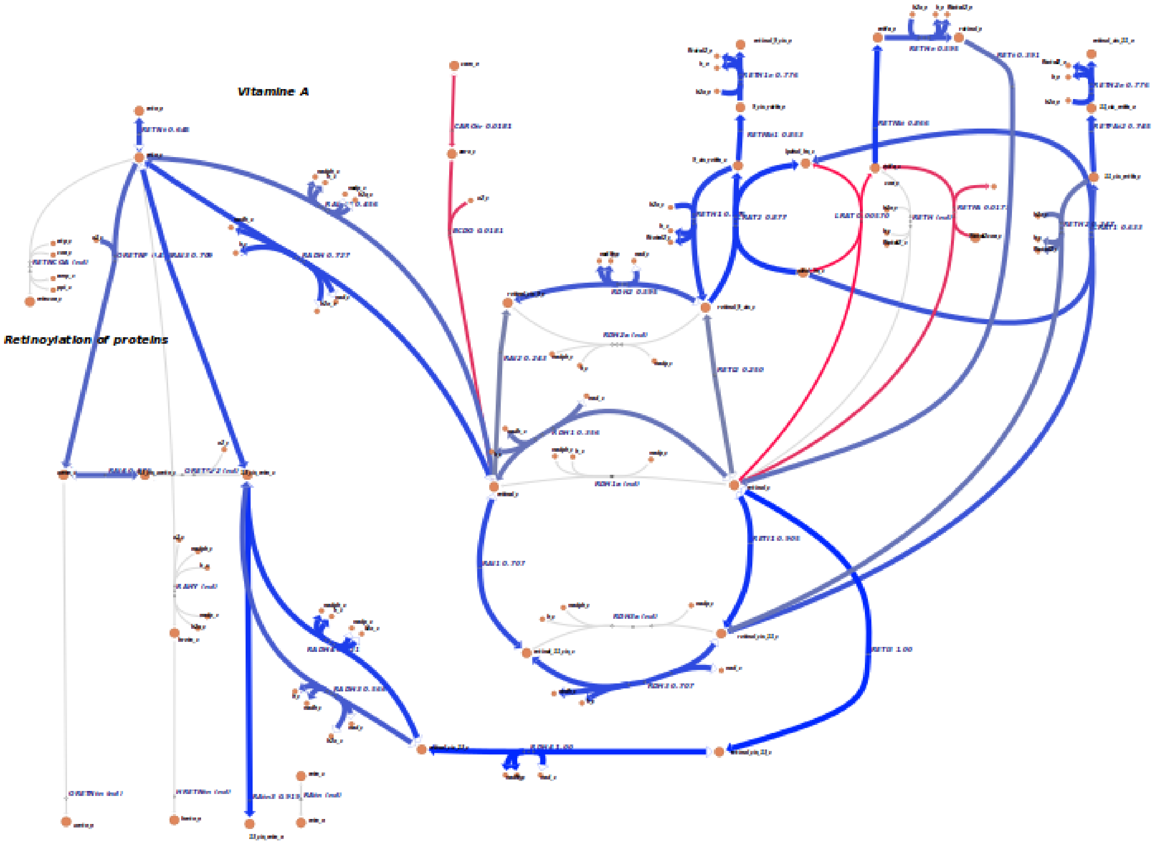


Figure 7.7: Escher's plot for retinol metabolism. Differentially expressed reactions are colored in red.

It is well known that retinoids, the family of molecules comprising the analogues of retinol, play a major role in the control of both cellular differentiation and proliferation (R. Blomhoff and H. K. Blomhoff 2006). Moreover, it has been shown that the metabolic dysregulation of retinoic acid (a metabolite of retinol) is implicated in tumorigenesis (R. Blomhoff and H. K. Blomhoff 2006; Osanai and Petkovich 2005). In our analysis, retinol metabolism presented differentially expressed reactions. Specifically, there were four of them: CAROtr, BCDO, LRAT and RETFA. Recent literature documents cases that link the activity of LRAT to certain tumor types such as colorectal cancer (Brown et al. 2014).

7.4.6 Visualizing Metabolic Networks with Graphviz

Escher provides some pre-generated metabolic maps useful to study the results obtained by means of FBA. However, as it was explained in Section 5.2, if we want a greater versatility the use of interactive editors such as Escher may not be the best alternative.

fcap, the open-source toolkit we have developed for this work incorporates one tool to automate the generation of metabolic maps by means of Graphviz. In particular, it processes the information contained in a metabolic network reconstruction in SBML format, such as the Recon 2 reconstruction, and generates a graph representing the different reactions and metabolites contained in it. The tool is also able to display data associated to the reactions, such as fluxes or p-values, using different colors depending on their magnitude.

Due to the fact that regular metabolic reconstructions have thousands of reactions and metabolites, representing them in a single diagram would not produce intelligible results.

To tackle this problem, the tool also accepts as input a list of the specific reactions that we want to see in the diagram. In addition to this, the final output can also be improved by providing a list identifying which metabolites are external (e.g. H_2O).

Figure 7.8 shows a diagram of the TCA cycle using our tool. In particular, the flux values for the KIRC sample with code `TCGA.A3.3324.01A.02R.1325.07` are depicted (the result can be compared with the corresponding Escher diagram shown in Figure 7.5). Positive fluxes are shown in red and negative ones in blue. Arrows always show reaction senses in direct order according to the stoichiometric coefficients. Therefore, when a given flux is negative in the picture, it should be taken into account that its associated reaction is working in reverse mode. A list of external metabolites were used to obtain a more clear diagram. As it can be seen, the result was fairly acceptable despite the fact that no human intervention was required to generate the plot.

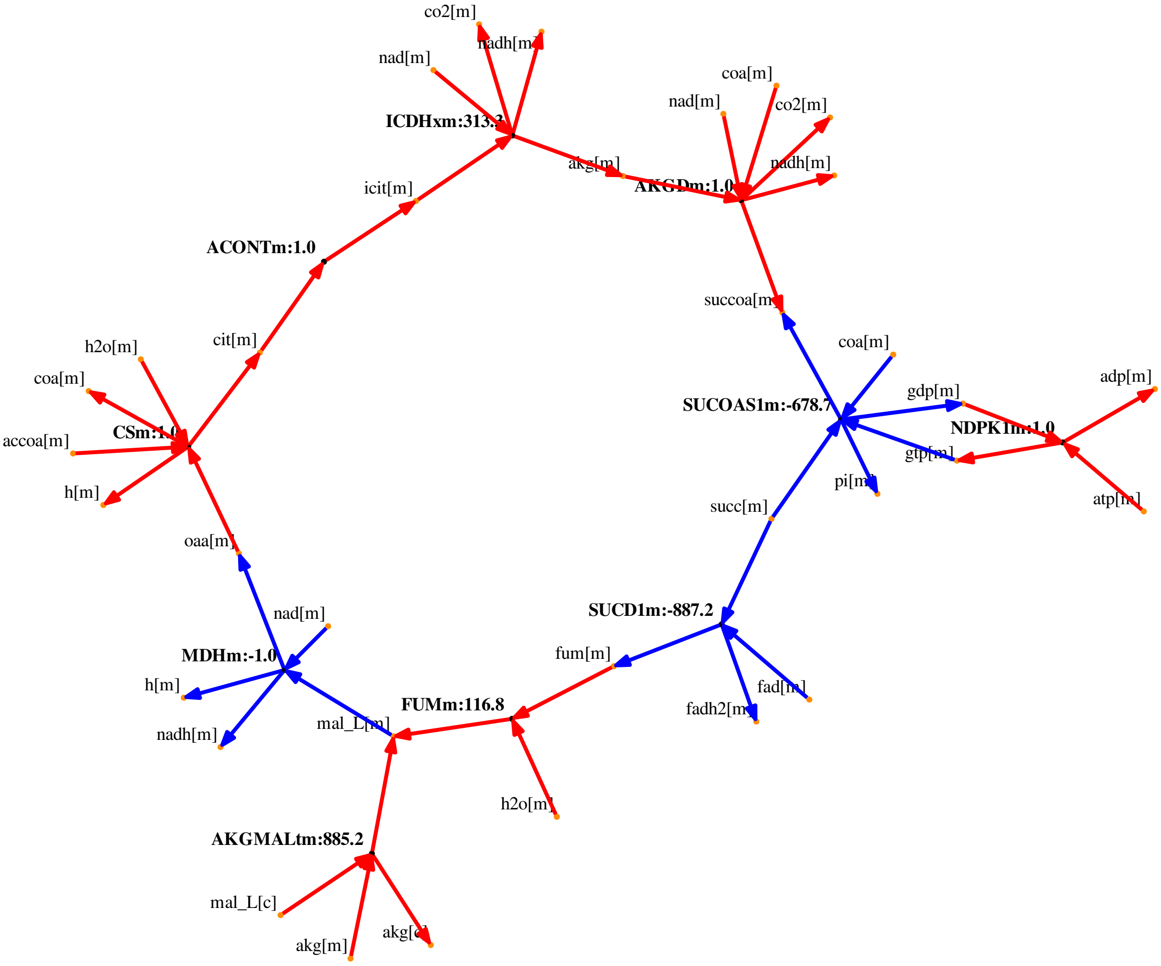


Figure 7.8: `fcap` representation using Graphviz of the metabolic fluxes for KIRC sample with code `TCGA.A3.3324.01A.02R.1325.07`. The picture shows the TCA cycle. Positive fluxes are displayed in red and negative ones in blue. Arrows shows the sense of the reactions in direct order as given by stoichiometric coefficients. Reactions with a negative flux are working in reverse mode.

Obtaining good representation results in an automated manner becomes difficult when the number of network elements is increased. To test this circumstance, we represented the p-values obtained when applying tissue-specific FBA to the KIRC samples, focusing on retinol metabolism. Figure 7.9 shows the resulting diagram, which can be compared with that provided by Escher (see Figure 7.7). In this case, the result is still intelligible, but could be improved by manually editing the diagram.

7.4.7 Reducing Metabolic Networks

In the previous section, we have used Graphviz to obtain automated representations of small metabolic networks. Such small networks can be seen as biological subsystems that are easier to be studied in an isolated manner. However, it could also be interesting to see additional context information for the subsystems by adding a limited and controlled number of network elements. For this purpose, it can be useful to apply a network reducing algorithm such as *NetworkReducer*, which was described in Section 5.3.

`fcap`, the software package created for this thesis implements the fast version of *NetworkReducer* we have proposed (see Algorithm 5.2), which is more appropriate to handle the large metabolic network contained in Recon 2. The algorithm works by iteratively removing reactions from the network while ensuring that some protected elements are retained. At each iteration, the reaction with lowest flux range from a randomly selected subset according to the FVA procedure is removed. *NetworkReducer* can be used to generate representations of biological subsystems incorporating additional context information. In particular, we can provide the whole Recon 2 network as input for the algorithm, defining the reactions that compose the biological subsystem of interest (e.g. the retinol metabolism) as protected reactions. The algorithm can be executed until the resulting network only contains the reactions of the protected subsystem. Intermediate results can be saved to disk and graphically represented, so as to gain knowledge about additional biological subsystems that may be related to that being studied.

Figure 7.10 shows different reduced networks obtained when applying *NetworkReducer* to Recon 2 using the set of reactions that compose retinol metabolism as protected reactions. Specifically, the networks were represented every 200 algorithm iterations ranging from 6 200 to 7 200. The reaction with lowest flux range when maximizing the biomass function was removed at each iteration. The plots display reactions outside retinol metabolism in gray color. On the other hand, differentially and non-differentially expressed retinol metabolism reactions were represented using red and blue colors, respectively. As it can be observed, reactions appear concentrated in a few dense areas. The reduction algorithm tends to retain such areas while removing isolated reactions. This is due to the fact that those reactions with lowest flux ranges are eliminated first. Isolated reactions typically have lower flux ranges since there are no other reactions in their vicinity able to balance the flow of metabolites.

To finish the analysis, we also inspected in more detail the network obtained after executing 7 200 iterations of the *NetworkReducer* algorithm. In particular, we added corrected p-values and reaction names to the diagram shown in Figure 7.10f, looking for differentially expressed reactions outside (but in the vicinity of) retinol metabolism. One example of such reactions was CHOLtu (p-value=$4.2 \cdot 10^{-6}$), which is related to the extracellular transport of choline. Choline is a water-soluble nutrient that is involved in the biosynthesis of cell membranes. Abnormalities in choline processing have been identified as tumor biomarkers (Gillies and Morse 2005).

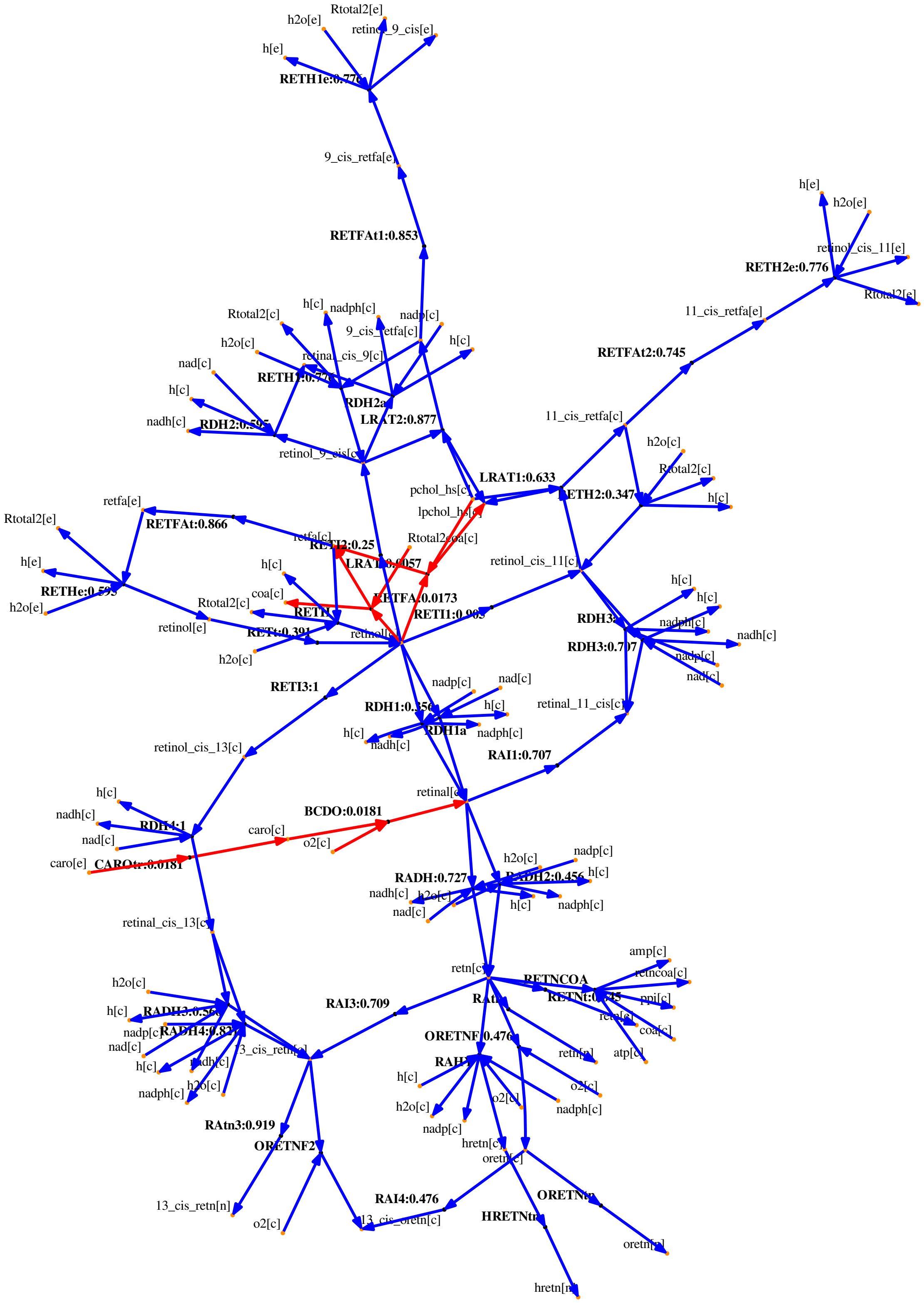
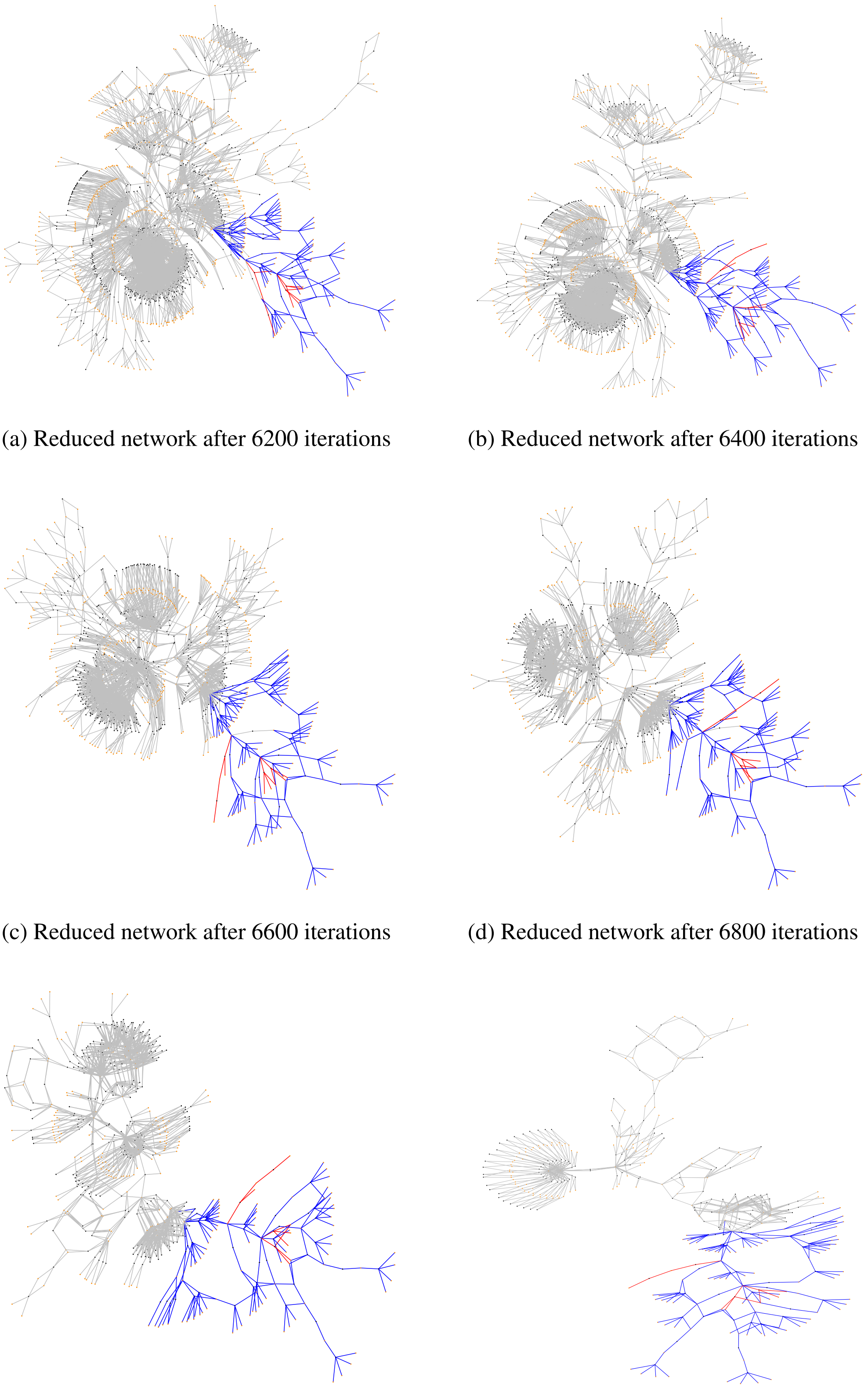


Figure 7.9: `fcap`'s plot for retinol metabolism showing the corrected p-values for each reaction. Differentially expressed reactions are colored in red (α was equal to 0.05).



(a) Reduced network after 6200 iterations

(b) Reduced network after 6400 iterations

(c) Reduced network after 6600 iterations

(d) Reduced network after 6800 iterations

(e) Reduced network after 7000 iterations

(f) Reduced network after 7200 iterations

Figure 7.10: Result of applying the *NetworkReducer* algorithm to the Recon 2 metabolic model, using the reactions contained in the Escher's retinol metabolism map as protected reactions. Reduced networks after executing a varying number of algorithm iterations are shown. Gray color is used to represent reactions not belonging to retinol metabolism.

CHAPTER 8
CONCLUSIONS AND FUTURE WORK

THIS chapter summarizes the main findings of our systems biology study on cancer metabolism. Such findings are used to draw general conclusions as well as to identify further research lines and improvements in current work that we plan to develop in the future.

8.1 Conclusions

Chapter 2 was devoted to describe the goals pursued in this work. These goals were classified into scientific ([SC]) and technologic ([TC]) goals. Below we describe the main results obtained for each of them:

- **Integration of transcriptomic and metabolic information using FBA** [SC]

  At the beginning of this document, we mentioned the great importance of data integration within the discipline of systems biology. In our experiments we have successfully integrated RNA-Seq data of healthy and cancerous kidney cells coming from TCGA database with the human metabolic model contained in Recon 2. For this purpose, first we determined the list of absent and present genes for each sample by identifying two different messenger RNA abundance classes. Second, we obtained lists of lowly and highly expressed reactions by applying gene to protein-reactions rules. Third, tissue-specific FBA was applied to compute a mathematical model of human metabolism taking as input the lists of lowly and highly expressed reactions as well as the steady state and flux range constraints stored in Recon 2. In addition to this, after obtaining the solution for the FBA problem, FVA was used to study the robustness of the metabolic model.

- **Review of available software for FBA** [TC]

  Sophisticated mathematical optimization tools called solvers are required to find solutions for the linear and integer programming problems that arise when FBA is applied. In this work we have tested four of them: GLPSOL, CLP, CBC and CPLEX. GLPSOL, CLP and CPLEX were applied to solve the linear programming problem derived from maximizing the Recon 2's biomass function. The three solvers were able to find the optimal solution in fractions of a second. On the other hand, GLPSOL, CBC and CPLEX were used to deal with the integer programming problems posed by tissue-specific FBA. In this case, CPLEX was clearly the fastest alternative requiring around 10 seconds to solve each problem, followed by CBC, which spent 800 seconds. By contrast GLPSOL was not able to find the solutions in reasonable time. In addition to this, it was possible to greatly reduce the time cost of CBC by allowing the algorithm to obtain suboptimal solutions. This finding was particularly relevant due to the fact that CPLEX is a commercial tool providing restrictive license conditions while CBC is part of an open-source software initiative.

- **Statistical analysis of FBA results** [SC]

  To be able to compare metabolism of normal and cancerous cells we devised a differential reaction expression experiment. In particular, flux values obtained by means of tissue-specific FBA for the different samples were converted into ordinal values representing reactions that are inactive, active in the direct sense or active in the reverse sense. Afterwards, the transformed values were analyzed by means of the Mann-Whitney's U-test, obtaining a set of p-values that were corrected using the Benjamini-Hochberg procedure. The ordinal representation was used due to the great variability shown by the network fluxes. Since common hypothesis tests such as the t-test are not designed to work with ordinal variables, we chose the U-test instead, which does not have this restriction.

- **Visualization techniques for FBA results** [SC]

  Visualization plays an important role in the analysis of systems biology results, making them easier to interpret. In this thesis we have tackled the visualization problem by means of the pre-generated maps provided by Escher and also with maps created in an automatic manner using Graphviz. Escher was applied to represent FBA fluxes as well as the p-values of our differential reaction expression experiment. Escher proved to be very useful to analyze the biological subsystems contained in its maps. On the other hand, Graphviz was fairly effective to represent small networks such as the TCA cycle, generating diagrams comparable to those of Escher. In contrast to this, the results were less intelligible for greater networks like that of retinol metabolism. However, a very interesting application of Graphviz was the representation of reduced networks obtained with the *NetworkReducer* algorithm. Specifically, *NetworkReducer* was used to generate networks containing the reactions related to retinol metabolism and also additional context information in the form of reactions belonging to other metabolic subsystems in the vicinity. The resulting networks were represented in an automated manner by means of Graphviz. This functionality could not be obtained when using Escher.

- **Comparison between normal and cancer metabolism** [SC]

  The differential reaction expression experiment we designed for this thesis allowed us to discover alterations in different metabolic subsystems when comparing healthy and cancerous samples. Such alterations were present in specific parts of the amino-acid, carbohydrate and retinol metabolisms. Retinol metabolism is particularly interesting from the perspective of this work due to its well known relationship with cell growth and proliferation. The LRAT reaction belonging to retinol metabolism was identified as differentially expressed, in line with certain cancer studies presented in the literature. The study of additional reactions in the vicinity of retinol metabolism using Graphviz revealed another differentially expressed reaction, CHOLtu, related to extracellular transport of choline. This also coincides with existing works in the literature that establish a link between cancer and abnormalities in choline processing.

- **Development of open-source software for FBA** [TC]

  For this thesis we have developed an open-source software package called Flux Capacitor or `fcap`. Flux Capacitor includes many features useful to carry out FBA, network visualization and statistical testing. A detailed description of the toolkit is provided in Appendix A.

8.2 Future Work

After enumerating the main achievements of this thesis, we identify the following directions for future developments:

- **More detailed analysis of tissue-specific FBA results**:

  The tissue-specific FBA study comparing metabolism of healthy and cancerous cells presented here can be extended in many ways. First, the whole list of differentially expressed reactions as well as its representation by means of Graphviz and Escher could be the subject of a detailed analysis under a strictly biological point of view, aspect that was beyond the scope of this work. Second, it would be important to perform a differential gene expression analysis complementing the results presented in this thesis, which were strictly focused on studying metabolic reactions. Third, due to the crucial role played by the lists of lowly and highly expressed reactions in the results obtained by tissue-specific FBA, it would also be interesting to conduct a systematic analysis for the different reactions, obtaining detailed statistics about the process by which they were considered as active or inactive. This process depends on things such as whether a given reaction is affected by a gene-protein-reaction association or if the solver needed to change the initial status assigned to a reaction to satisfy the problem constraints.

- **Better network representations using Graphviz**:

  Network representations generated with Graphviz were not as good as those provided by Escher when the number of network elements was increased. Graphviz includes some features not exploited here that could be useful to improve the quality of the diagrams. One of these features is to use *subgraphs* as a way to cluster sets of related reactions and metabolites. When drawing a specific network, the visualization tool could receive as input parameter the different sets in which the reactions to be represented are clustered, and use this information to structure the diagram.

- **Further experiments with *NetworkReducer***:

  In this work we have presented preliminary experiments using the *NetworkReducer* algorithm. These experiments can be extended to other metabolic maps, such as those provided by Escher. In addition to this, it could be interesting to change the criterion used by the algorithm to remove reactions at each iteration. Here we have selected those reactions with lowest flux range as candidates to be removed, resulting in an algorithm that tends to preserve areas with a high density of reactions instead of areas with isolated ones. If the criterion was reversed, that is, removing reactions with highest flux ranges first, then the resulting network would contain reactions whose flux could not easily change. In other words, it would be composed of essential reactions.

- **Improvements and extensions in Flux Capacitor**:

  The software developed for this thesis can be improved and extended in many different ways. One negative aspect of the package are its multiple dependencies with existing software, including several R and Python modules. It would be interesting to reduce such dependencies, or even to simplify the package design by totally removing the use of one of the two above mentioned languages. Using only one programming language would also allow to develop an interactive mode to access the toolkit functionality in parallel to the batch-oriented processing currently implemented. Finally, Flux Capacitor is strongly focused on the use of CPLEX as mathematical solver, which is distributed in a commercial package. CPLEX could be replaced by the freely available solvers CLP and CBC. In spite of the fact that CPLEX is the fastest option, both CLP and CBC have demonstrated to work fast enough when the appropriate parameters are used.

Bibliography

Ashburner, M., C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock (2000). "Gene ontology: tool for the unification of biology. The Gene Ontology Consortium". In: *Nat. Genet.* 25.1, pp. 25–29.

Benjamini, Y. and Y. Hochberg (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 57.1, pp. 289–300. ISSN: 00359246. DOI: 10.2307/2346101.

Blomhoff, R. and H. K. Blomhoff (2006). "Overview of retinoid metabolism and function". In: *J. Neurobiol.* 66.7, pp. 606–630.

Bolstad, B. M., F. Collin, J. Brettschneider, K. Simpson, L. Cope, R. A. Irizarry, and T.P. Speed (2005). "Quality Assessment of Affymetrix GeneChip Data". In: *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Ed. by Robert Gentleman, Vincent J. Carey, Wolfgang Huber, Rafael A. Irizarry, and Sandrine Dudoit. New York, NY: Springer New York, pp. 33–47. ISBN: 978-0-387-29362-2. DOI: 10.1007/0-387-29362-0_3.

Bordbar, A., J. M. Monk, Z. A. King, and B. Ø. Palsson (2014). "Constraint-based models predict metabolic and associated cellular functions". In: *Nat. Rev. Genet.* 15.2, pp. 107–120.

Brown, G. T., B. G. Cash, D. Blihoghe, P. Johansson, A. Alnabulsi, and G. I. Murray (2014). "The expression and prognostic significance of retinoic acid metabolising enzymes in colorectal cancer". In: *PLoS ONE* 9.3, e90776.

Cairns, R. A., I. S. Harris, and T. W. Mak (2011). "Regulation of cancer cell metabolism". In: *Nat. Rev. Cancer* 11, pp. 85–95.

Corney, D. C. (2013). "RNA-Seq using next generation sequencing". In: *Mat. Methods* 3.213.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). "Maximum likelihood from incomplete data via the EM algorithm". In: *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B* 39.1, pp. 1–38.

DeRisi, J. L., V. R. Iyer, and P. O. Brown (1997). "Exploring the metabolic and genetic control of gene expression on a genomic scale." In: *Science (New York, N.Y.)* 278.5338, pp. 680–686. ISSN: 0036-8075. DOI: 10.1126/science.278.5338.680.

Dobkin, D. and E. Koutsofios (1991). "LEFTY: A Two-view Editor for Technical Pictures". In: *Graphics Interface '91*, pp. 68–76.

Domach, M. M., Leung S. K., Cahn R. E., Cocks G. G., and Shuler M. L. (1984). "Computer model for glucose-limited growth of a single cell of Escherichia coli B/r-A". In: *Biotechnology and Bioengineering* 26, pp. 203–216.

Duarte, N. C., S. A. Becker, N. Jamshidi, I. Thiele, M. L. Mo, T. D. Vo, R. Srivas, and B. Ø. Palsson (2007). "Global reconstruction of the human metabolic network based on genomic and bibliomic data". In: *Proc Natl Acad Sci U S A* 104.6, pp. 1777–1782. ISSN: 0027-8424.

Dudoit, S., J. P. Shaffer, and J. C. Boldrick (2003). “Multiple Hypothesis Testing in Microarray Experiments”. In: *Statist. Sci.* 18.1, pp. 71–103. DOI: 10.1214/ss/1056397487.

Erdrich, P., R. Steuer, and S. Klamt (2015). “An algorithm for the reduction of genome-scale metabolic network models to meaningful core models”. In: *BMC Syst Biol* 9, p. 48.

Feist, A. M., C. S. Henry, J. L. Reed, M. Krummenacker, A. R. Joyce, P. D. Karp, L. J. Broadbelt, V. Hatzimanikatis, and B. Ø. Palsson (2007). “A genome-scale metabolic reconstruction for Escherichia coli K-12 MG1655 that accounts for 1260 ORFs and thermodynamic information”. In: *Mol. Syst. Biol.* 3, p. 121.

Fell, D. (1996). *Understanding the Control of Metabolism (Frontiers in Metabolism)*. Portland Press, London. ISBN: 185578047X.

Fell, D. A. and J. R. Small (1986). “Fat synthesis in adipose tissue. An examination of stoichiometric constraints”. In: *Biochem. J.* 238.3, pp. 781–786.

Gansner, E. R. and S. C. North (2000). “An open graph visualization system and its applications to software engineering”. In: *SOFTWARE - PRACTICE AND EXPERIENCE* 30.11, pp. 1203–1233.

Gansner, E., E. Koutsofios, and S. North (2010). *Drawing graphs with dot*. Tech. rep.

Gautier, L., L. Cope, B. M. Bolstad, and R. A. Irizarry (2004). “affy—analysis of Affymetrix GeneChip data at the probe level”. In: *Bioinformatics* 20.3, pp. 307–315. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btg405.

Gentleman, R., V. Carey, W. Huber, and F. Hahne (2016). *genefilter: methods for filtering genes from high-throughput experiments*. R package version 1.52.1.

Gentleman, R., V. Carey, W. Huber, R. Irizarry, and S. (eds) Dudoit (2005). *Bioinformatics and computational biology solutions using R and Bioconductor*. Statistics for biology and health. Springer Science+Business Media.

Gille, C, C Bölling, A Hoppe, S Bulik, S Hoffmann, K Hübner, A Karlstädt, R Ganeshan, M König, K Rother, M Weidlich, J Behre, and H. G. Holzhütter (2010). “HepatoNet1: a comprehensive metabolic reconstruction of the human hepatocyte for the analysis of liver physiology”. In: *Mol. Syst. Biol.* 6.411.

Gillies, R. J. and D. L. Morse (2005). “In vivo magnetic resonance spectroscopy in cancer”. In: *Annu Rev Biomed Eng* 7, pp. 287–326.

Gudmundsson, S. and I. Thiele (2010). “Computationally efficient flux variability analysis”. In: *BMC Bioinformatics* 11, p. 489.

Hao, T., H. W. Ma, X. M. Zhao, and I. Goryanin (2010). “Compartmentalization of the Edinburgh Human Metabolic Network”. In: *BMC Bioinformatics* 11.1. ISSN: 1471-2105. DOI: 10.1186/1471-2105-11-393.

Hebenstreit, D., M. Fang, M. Gu, V. Charoensawan, A. van Oudenaarden, and S. A. Teichmann (2011). “RNA sequencing reveals two major classes of gene expression levels in metazoan cells”. In: *Mol. Syst. Biol.* 7, p. 497.

Hillier, F. S. and G. J. Lieberman (1986). *Introduction to Operations Research, 4th Ed.* San Francisco, CA, USA: Holden-Day, Inc. ISBN: 0816238715.

Huber, W., V. J. Carey, R. Gentleman, S. Anders, M. Carlson, B. S. Carvalho, H. C. Bravo, S. Davis, L. Gatto, T. Girke, R. Gottardo, F. Hahne, K. D. Hansen, R. A. Irizarry, M. Lawrence, M. I. Love, J. MacDonald, V. Obenchain, A. K. Oleś, H. Pagès, A. Reyes, P. Shannon, G. K. Smyth, D. Tenenbaum, L. Waldron, and M. Morgan (2015). “Orchestrating high-throughput genomic analysis with Bioconductor”. In: *Nat. Methods* 12.2, pp. 115–121.

Hucka, M., A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J. H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. Le Novere, L. M. Loew, D. Lucio, P. Mendes, E. Minch, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang (2003). "The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models". In: *Bioinformatics* 19.4, pp. 524–531.

King, Z. A., A. Drager, A. Ebrahim, N. Sonnenschein, N. E. Lewis, and B. Ø. Palsson (2015). "Escher: A Web Application for Building, Sharing, and Embedding Data-Rich Visualizations of Biological Pathways". In: *PLoS Comput. Biol.* 11.8, e1004321.

Kitano, H. (2002). "Computational systems biology". In: *Nature* 420.6912, pp. 206–210.

Klipp, E., W. Liebermeister, C. Wierling, and A. Kowald (2016). *Systems Biology: A Textbook (Second Edition)*. Weinheim: Wiley-VCH. ISBN: 978-3-527-33636-4.

Langmead, B., K. D. Hansen, and J. T. Leek (2010). "Cloud-scale RNA-sequencing differential expression analysis with Myrna". In: *Genome Biol.* 11.8, R83.

Langmead, B., C. Trapnell, M. Pop, and S. L. Salzberg (2009). "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome". In: *Genome Biol.* 10.3, R25.

Lanpher, B., N. Brunett-Pierri, and B. Lee (2006). "Inborn errors of metabolism: the flux from Mendelian to complex diseases". In: *Nat. Rev. Genet.* 7, pp. 449–460.

Levine, David M., David R. Haynor, John C. Castle, Sergey B. Stepaniants, Matteo Pellegrini, Mao Mao, and Jason M. Johnson (2006). "Pathway and gene-set activation measurement from mRNA expression data: the tissue distribution of human pathways". In: *Genome Biology* 7.10, pp. 1–17. ISSN: 1474-760X. DOI: 10.1186/gb-2006-7-10-r93.

Li, Q., N. J. Birkbak, B. Gyorffy, Z. Szallasi, and A. C. Eklund (2011). "Jetset: selecting the optimal microarray probe set to represent a gene". In: *BMC Bioinformatics* 12, p. 474.

Likić, V. A., M. J. McConville, T. Lithgow, and A. Bacic (2010). "Systems biology: the next frontier for bioinformatics". In: *Adv Bioinformatics*. DOI: 10.1155/2010/268925.

Maglott, D., J. Ostell, K. D. Pruitt, and T. Tatusova (2011). "Entrez Gene: gene-centered information at NCBI". In: *Nucleic Acids Res.* 39.Database issue, pp. D52–57.

Mahadevan, R. and C. H. Schilling (2003). "The effects of alternate optimal solutions in constraint-based genome-scale metabolic models". In: *Metab. Eng.* 5.4, pp. 264–276.

Mann, H. B. and D. R. Whitney (1947). "On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other". In: *Ann. Math. Statist.* 18.1, pp. 50–60. DOI: 10.1214/aoms/1177730491.

Marioni, J. C., C. E. Mason, S. M. Mane, M. Stephens, and Y. Gilad (2008). "RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays". In: *Genome Res.* 18.9, pp. 1509–1517.

Meindl, B. and M. Templ (2013). "Analysis of Commercial and Free and Open Source Solvers for the Cell Suppression Problem". In: *Trans. Data Privacy* 6.2, pp. 147–159. ISSN: 1888-5063.

Morgan, M., R. Gentleman, and V. Carey (2016). *GSEABase: Gene set enrichment data structures and methods*. R package version 1.34.0.

Mortazavi, A., B. A. Williams, K. McCue, L. Schaeffer, and B. Wold (2008). "Mapping and quantifying mammalian transcriptomes by RNA-Seq". In: *Nat. Methods* 5.7, pp. 621–628.

Muoio, D. M. and C. B. Newgard (2006). "Obesity-related derangements in metabolic regulation". In: *Annu. Rev. Biochem.* 75, pp. 367–401.

NCI and the NHGRI (2005a). *The Cancer Genome Atlas*. URL: http://cancergenome.nih.gov/ (visited on 07/17/2016).

NCI and the NHGRI (2005b). *The Cancer Genome Atlas, How It Works*. URL: http://cancergenome.nih.gov/newsevents/multimedialibrary/interactives/howitworks (visited on 07/17/2016).

NCI and the NHGRI (2005c). *The Cancer Genome Atlas, Program Overview*. URL: http://cancergenome.nih.gov/abouttcga/overview (visited on 07/17/2016).

Oberhardt, M. A., B. Ø. Palsson, and Jason A. Papin (2009). "Applications of genome-scale metabolic reconstructions". In: *Molecular systems biology* 5.1. ISSN: 1744-4292. DOI: 10.1038/msb.2009.77.

Orth, J. D., I. Thiele, and B. Ø. Palsson (2010). "What is flux balance analysis?" In: *Nat. Biotechnol.* 28.3, pp. 245–248.

Osanai, M. and M. Petkovich (2005). "Expression of the retinoic acid-metabolizing enzyme CYP26A1 limits programmed cell death". In: *Mol. Pharmacol.* 67.5, pp. 1808–1817.

Oshlack, A., M. D. Robinson, and M. D. Young (2010). "From RNA-seq reads to differential expression results". In: *Genome Biology* 11.12, pp. 1–10. ISSN: 1474-760X. DOI: 10.1186/gb-2010-11-12-220.

Pollard, Katherine S., Sandrine Dudoit, and Mark J. van der Laan (2005). *Multiple Testing Procedures: R multtest Package and Applications to Genomics, in Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Springer.

Price, N. D., J. A. Papin, Schilling C. H., and Palsson B. Ø. (2003). "Genome-scale microbial in silico models: the constraints-based approach". In: *Trends Biotechnol.* 21, pp. 162–169.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. URL: https://www.R-project.org/.

Robinson, M. D., D. J. McCarthy, and G. K. Smyth (2010). "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data". In: *Bioinformatics* 26.1, pp. 139–140.

Robinson, M. D. and G. K. Smyth (2007). "Moderated statistical tests for assessing differences in tag abundance". In: *Bioinformatics* 23.21, pp. 2881–2887.

Rosenblatt, M. (1956). "Remarks on Some Nonparametric Estimates of a Density Function". In: *Ann. Math. Statist.* 27.3, pp. 832–837. DOI: 10.1214/aoms/1177728190.

Sahoo, S., L. Franzson, J. J. Jonsson, and I. Thiele (2012). "A compendium of inborn errors of metabolism mapped onto the human metabolic network". In: *Mol. BioSyst.* 8 (10), pp. 2545–2558. DOI: 10.1039/C2MB25075F.

Sahoo, S. and I. Thiele (2013). "Predicting the impact of diet and enzymopathies on human small intestinal epithelial cells". In: *Human Mol. Genet.* 13.22. DOI: 10.1093/hmg/ddt119.

Saiki, R. K., D. H. Gelfand, S. Stoffel, S. J. Scharf, R. Higuchi, G. T. Horn, K. B. Mullis, and H. A. Erlich (1988). "Primer-directed enzymatic amplification of DNA with a thermostable DNA polymerase". In: *Science* 239.4839, pp. 487–491.

Shlomi, T., M. N. Cabili, M. J. Herrgård, B. Ø. Palsson, and Eytan Ruppin (2008). “Network-based prediction of human tissue-specific metabolism”. In: *Nat. Biotechnol.* 26(9), pp. 1003–10.

Son, C. G., S Bilke, S Davis, B. T. Greer, J. S. Wei, C. C. Whiteford, Q. R. Chen, N. Cenachi, and J. Kahn (2005). “Database of mRNA gene expression profiles of multiple human organs”. In: *Genome Res.* 15(3), pp. 443–50.

Thiele, I. and B. Ø. Palsson (2010). “A protocol for generating a high-quality genome-scale metabolic reconstruction”. In: *Nat Protoc* 5.1, pp. 93–121.

Thiele, I., N. Swainston, R. M. Fleming, A. Hoppe, S. Sahoo, M. K. Aurich, H. Haraldsdottir, M. L. Mo, O. Rolfsson, M. D. Stobbe, S. G. Thorleifsson, R. Agren, C. Bolling, S. Bordel, A. K. Chavali, P. Dobson, W. B. Dunn, L. Endler, D. Hala, M. Hucka, D. Hull, D. Jameson, N. Jamshidi, J. J. Jonsson, N. Juty, S. Keating, I. Nookaew, N. Le Novere, N. Malys, A. Mazein, J. A. Papin, N. D. Price, E. Selkov, M. I. Sigurdsson, E. Simeonidis, N. Sonnenschein, K. Smallbone, A. Sorokin, J. H. van Beek, D. Weichart, I. Goryanin, J. Nielsen, H. V. Westerhoff, D. B. Kell, P. Mendes, and B. Ø. Palsson (2013). “A community-driven global reconstruction of human metabolism”. In: *Nat. Biotechnol.* 31.5, pp. 419–425.

Tomczak, K., P. Czerwińska, and M. Wiznerowicz (2015). “The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge”. In: *Contemp Oncol (Pozn)* 19.1A, pp. 68–77.

Trapnell, C., L. Pachter, and S. L. Salzberg (2009). “TopHat: discovering splice junctions with RNA-Seq”. In: *Bioinformatics* 25.9, pp. 1105–1111.

Trapnell, C., B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter (2010). “Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation”. In: *Nat. Biotechnol.* 28.5, pp. 511–515.

Wang, Z., M. Gerstein, and M. Snyder (2009). “RNA-Seq: a revolutionary tool for transcriptomics”. In: *Nat. Rev. Genet.* 10.1, pp. 57–63.

Warren, Peter (2016). *panp: Presence-Absence Calls from Negative Strand Matching Probesets*. R package version 1.40.0.

Wu, J. and R. Irizarry (2016). *gcrma: Background Adjustment Using Sequence Information*. With contributions from James MacDonald and Jeff Gentry. R package version 2.42.0.

Yanai, I., H. Benjamin, M. Shmoish, V. Chalifa-Caspi, M. Shklar, R. Ophir, A. Bar-Even, S. Horn-Saban, M. Safran, E. Domany, D. Lancet, and O. Shmueli (2005). “Genome-wide midrange transcription profiles reveal expression level relationships in human tissue specification”. In: *Bioinformatics* 21.5, pp. 650–659. DOI: 10.1093/bioinformatics/bti042.

Young, M. D., M. J. Wakefield, G. K. Smyth, and A. Oshlack (2010). “Gene ontology analysis for RNA-seq: accounting for selection bias”. In: *Genome Biol.* 11.2, R14.

Appendix A
Open-Source Software

OPEN-SOURCE software allows the rapid propagation of new ideas in a research context. For this thesis we have developed Flux Capacitor or `fcap`, an open-source software package for systems biology.

A.1 Introduction

`fcap` is focused on the application of FBA to study metabolism. The toolkit is currently under development. Below there is a list of its main functionalities:

- **FBA**: the toolkit implements FBA (see Section 4.4) for metabolic models given in SBML format, maximizing the biomass function and returning the flux values that correspond to the optimal solution.
- **FVA**: a parallel version of the FVA procedure (see Section 4.5) is included. `fcap` also incorporates the techniques proposed in (Gudmundsson and Thiele 2010) to accelerate the calculations (see Section 4.5.2 for more details).
- **Tissue-specific FBA**: the package provides an implementation of the tissue-specific FBA procedure proposed by Shlomi et al. 2008 (see Section 4.6). The application of tissue-specific FBA requires the generation of lists of lowly and highly expressed reactions. As it was explained, the procedure required to obtain such lists depends on whether the gene expression data comes from a microarray experiment (see Section 4.6.2) or from an RNA-Seq experiment (see Section 4.6.3). `fcap` implements both procedures.
- **Statistical testing**: `fcap` allows to apply statistical hypothesis tests for case/control samples. In particular, the t-test and the Mann-Whitney's U-test can be executed.
- **Network visualization**: the package allows to generate automated graphical representations of metabolic networks in SBML format. For this purpose, the open-source graph visualization tool called Graphviz is used (see Section 5.2).
- **Network reduction**: `fcap` includes an implementation of the *NetworkReducer* algorithm proposed by Erdrich et al. 2015 (see Section 5.3.1) as well as a fast version of it specifically proposed in this work (see Section 5.3.2).

A.2 Distribution Details

`fcap` has been coded in Python, R, shell-scripting and C++. It can be compiled on UNIX-like and Windows (using Cygwin) systems.

`fcap` is released under the GNU Lesser General Public License[a] (LGPL), allowing other developers and companies to use and integrate the package into their own software without requiring that such software is also LGPL-licensed.

A.3 Installation

The code of the `fcap` toolkit is hosted on github[b]. To install `fcap`, first it is necessary to install the autotools (autoconf, autoconf-archive, automake and libtool packages in Ubuntu). If `fcap` is to be used on a Windows platform, the Cygwin environment[c] should be installed.

Once the autotools are available (as well as other required software such as Cygwin), the user can proceed with the installation of `fcap` by following the next sequence of steps:

1. Obtain the package using git:

   ```
   $ git clone https://github.com/daormar/flux-capacitor.git
   ```

   Additionally, `fcap` can be downloaded in a zip file[d].

2. `cd` to the directory containing the package's source code and type `./reconf`.

3. Type `./configure` to configure the package.

4. Type `make` to compile the package.

5. Type `make install` to install the programs and any data files and documentation.

6. You can remove the program binaries and object files from the source code directory by typing `make clean`.

By default the files are installed under the `/usr/local` directory (or similar, depending of the OS you use); however, since Step 5 requires root privileges, another directory can be specified during Step 3 by typing:

```
$ configure --prefix=<absolute-installation-path>
```

Additionally, `fcap` internally uses CPLEX as a mathematical solver to obtain the solutions required by FBA and FVA procedures. Therefore, users also need to install this package to be able to access most of the functionality of the toolkit.

[a] https://www.gnu.org/copyleft/lgpl.html

[b] https://github.com/daormar/flux-capacitor/

[c] https://www.cygwin.com/

[d] https://github.com/daormar/flux-capacitor/archive/master.zip

A.4 Main Tools

The functionality of fcap is provided by means of a set of tools executing modular tasks. Next, we provide a list of the most important of such tools, briefly describing the input parameters they expect as well as their dependencies with other software (if any):

- **extract_sbml_model_info**: extracts information from a metabolic model in SBML format. The program takes as input a file in SBML format and generates a list of text files with varied information (reaction and metabolite names, stoichiometric matrix, etc.). It is implemented in R and requires the sybilSBML library.

- **auto_fba**: automates an FBA procedure. The tool receives as input the name of the SBML file containing the metabolic model and the type of optimization to be computed: biomass function or tissue-specific. If tissue-specific FBA is to be applied, then the program requires transcriptomic information, that can be provided as a set of CEL files for microarray data or as a file with RNA-Seq counts. auto_fba is implemented as a UNIX shell script.

- **auto_fva**: automates a whole FVA procedure. The program takes as input the prefix of the files in lp format representing the initial FBA problem to be solved (they are obtained by means of the auto_fba tool). In addition to this, auto_fva also takes additional parameters to control process efficiency. This tool is implemented as a UNIX shell script.

- **test_samples**: performs statistical tests for a set of samples classified into cases and controls. The tool expects as input a CSV file with the sample data and another one with the phenotype data. test_samples is a Python program using the scipy and the statsmodels modules.

- **correct_pvalues**: corrects a set of p-values using the Benjamini-Hochberg procedure. It receives as input a file with p-values generated by means of test_samples and the value of α. The tool is written in Python and uses the statsmodels module.

- **plot_metab_network**: generates files in Graphviz format representing metabolic networks. Such files can later be converted to graphics files in different formats. The tool takes as input the plot type to be generated, the prefix of a series of files representing the metabolic network generated with the extract_sbml_model_info tool, a file containing the identifiers of the reactions to be included in the plot, another file with data about the reactions (e.g. flux values, p-values) and optionally, a list of identifiers of external metabolites. plot_metab_network is written in Python.

- **network_reducer**: reduces the number of elements of a metabolic network. It is designed to work with the output of the auto_fba tool. Its basic input parameters are those enumerated in Algorithm 5.2. network_reducer is a UNIX shell script.

All of the tools included in the package can display help messages describing their expected input parameters.