TUCS

Johannes Tuikkala

Algorithmic Techniques in Gene Expression Processing

From Imputation to Visualization

Turku Centre *for* Computer Science

TUCS Dissertations
No 185, November 2014

Algorithmic Techniques in Gene Expression Processing

From Imputation to Visualization

Johannes Tuikkala

To be presented, with the permission of the Faculty of Mathematics and Natural Sciences of the University of Turku, for public criticism in Auditorium of Mikro T31 on November 20, 2014, at 12 noon.

University of Turku
Department of Information Technology
Joukahaisenkatu 3-5, 20014 Turku 2014

2014

Supervisors

Professor Olli Nevalainen, PhD
Department of Information Technology
University of Turku
Finland

Docent Tero Aittokallio, PhD
Institute for Molecular Medicine Finland
University of Helsinki
Finland

Reviewers

Professor Ion Petre, PhD
Department of Information Technologies
Åbo Akademi University
Finland

Professor Mauno Vihinen, PhD
Department of Experimental Medical Science
Lund University
Sweden

Opponent

Professor Pasi Fränti, PhD
School of Computing
University of Eastern Finland
Finland

ISBN 978-952-12-3126-1
ISSN 1239-1883

Abstract

The amount of biological data has grown exponentially in recent decades. Modern biotechnologies, such as microarrays and next-generation sequencing, are capable to produce massive amounts of biomedical data in a single experiment. As the amount of the data is rapidly growing there is an urgent need for reliable computational methods for analyzing and visualizing it. This thesis addresses this need by studying how to efficiently and reliably analyze and visualize high-dimensional data, especially that obtained from gene expression microarray experiments.

First, we will study the ways to improve the quality of microarray data by replacing (imputing) the missing data entries with the estimated values for these entries. Missing value imputation is a method which is commonly used to make the original incomplete data complete, thus making it easier to be analyzed with statistical and computational methods. Our novel approach was to use curated external biological information as a guide for the missing value imputation.

Secondly, we studied the effect of missing value imputation on the downstream data analysis methods like clustering. We compared multiple recent imputation algorithms against 8 publicly available microarray data sets. It was observed that the missing value imputation indeed is a rational way to improve the quality of biological data. The research revealed differences between the clustering results obtained with different imputation methods. On most data sets, the simple and fast k-NN imputation was good enough, but there were also needs for more advanced imputation methods, such as Bayesian Principal Component Algorithm (BPCA).

Finally, we studied the visualization of biological network data. Biological interaction networks are examples of the outcome of multiple biological experiments such as using the gene microarray techniques. Such networks are typically very large and highly connected, thus there is a need for fast algorithms for producing visually pleasant layouts. A computationally efficient way to produce layouts of large biological interaction networks was developed. The algorithm uses multilevel optimization within the regular force directed graph layout algorithm.

Tiivistelmä

Biologisen tiedon määrä on kasvanut räjähdysmäisesti viime vuosikymmeninä. Modernin bioteknologian keinot, kuten geenimikrosirut ja uudet sekvensointimenetelmät tuottavat valtavia määriä biolääketieteellistä dataa yhdellä koesarjalla. Tiedon määrän kasvaessa yhä nopeammin, enenee myös tarve luotettaville tietoteknisille tiedon analysointi- ja visualisointiratkaisuille. Tässä väitöskirjastyössä pyritään löytämään ratkaisuja näihin tarpeisiin. Tutkimuksessa etsitään nopeita ja luotettavia tapoja analysoida ja visualisoida moniulotteista dataa, jota saadaan esimerkiksi geenimikrosirukokeiden tuloksena.

Työssä kehitettiin aluksi algoritmisia keinoja parantaa geenimikrosirudatan laatua korvaamalla (imputoimalla) datan puuttuvat arvot estimaattiarvoilla. Puuttuvien arvojen imputointi on menetelmä, jolla voidaan tuottaa alkuperäisestä epätäydellisestä datamatriisista täydellinen. Täydellisen datan etuna on mm. sen helpompi analysoitavuus tilastollisilla ja algoritmisilla menetelmillä. Tutkimuksessa kehitettiin uusi lähestymistapa imputointiin. Ideana oli ohjata imputointia käyttämällä apuna luotettavaa ulkoista biologista tietolähdettä.

Lisäksi tutkittiin tarkemmin sitä, kuinka imputaatio vaikuttaa jatkoanalysointimenetelmien, kuten ryvästämisen (klusteroinnin) tuloksiin. Työssä vertailtiin useaa imputaatioalgoritmia. Vertailu suoritettiin imputoimalla ja klusteroimalla kahdeksan erilaista mikrosirukokeilla tuotettua datamatriisia. Tutkimuksen tulokset vahvistivat oletusta siitä, että puuttuvien arvojen estimointi parantaa biologisen datan laatua klusteroinnin onnistumisella mitattuna. Eri imputaatiomenetelmillä estimoitujen datajoukkojen klusteroinneissa oli selkeitä eroja. Useimmilla mikrosirudatoilla nopea ja yleisesti käytetty k-NN-imputaatio menetelmä tuotti riittävän hyvälaatuisen datan klusteroitavaksi. Toisaalta tietyissä tapauksissa osoitettiin, että paras tulos saavutetaan vasta edistyksellisemmillä imputaatiomenetelmillä. Esimerkki tällaisesta oli bayeslaiseen pääkomponenttianalyysiin perustuva BPCA-algoritmi.

Lopuksi tutkittiin biologisten vuorovaikutusverkkojen visualisointia. Biologiset vuorovaikutusverkot ovat tyypillisesti lopputuloksia lukuisista biologisista koesarjoista, kuten mikrosirukokeista. Nämä verkot ovat usein hyvin isoja ja tiheitä. Tämän takia tarvitaan nopeita algoritmeja esteettisesti ja biologisesti hyvän visualisoinnin tuottamiseen. Tässä tutkimuksessa kehitettiin laskennallisesti tehokas tapa piirtää tällainen visualisointi. Kehitetty menetelmä hyödyntää monitasoista optimointia tavallisessa jousivoimaohjatussa verkon piirtoalgoritmissa.

Acknowledgements

First of all I want to thank my supervisors Professor Olli Nevalainen, PhD, and Docent Tero Aittokallio, PhD, for the guidance, support, and encouragement given in all phases of this work. I would like to thank the University of Turku and Turku Centre for Computer Science (TUCS) for making this work possible. I thank professor Esa Tyystjärvi for giving feedback on the text of Chapter 2. I wish to extend my thanks to Professor Jukka Teuhola for providing a valuable feedback on text of the thesis.

I want to thank my former colleagues Mikael Laine, MSc, and Olli Luoma, PhD, who provided many good laughs and inspiring discussions. I thank colleagues who were co-authors of our publications; from them I would especially like to thank Kati Talvinen, PhD and Laura Elo-Uhlgren, PhD. I am also grateful for my current employer Vaadin, for providing me the flexible working time; without that this thesis would have required much more sweat and teardrops. Thanks belong also to all of my colleagues in Vaadin.

I thank the reviewers of this thesis Professor Ion Petre, PhD, and Professor Mauno Vihinen, PhD, for their efforts and feedback. I thank Professor Pasi Fränti for accepting to act as opponent at the disputation of this thesis.

I sincerely thank my wife Kaisa and my daughters Saara and Mirjami for their endless love, and support during the time I spent working with the thesis. My special thanks belong to my parents, Hannele and Mauri, and deceased grandparents, Aino-mummo, Sylvi-mummu, and Matti-ukko, for their love and their encouragement. I'm grateful to all my 14 siblings and to my godmother Mirja and godfather Kalevi for having them in my life.

Finally, I would like to express my thanks to all people close to me for their support, encouragement and interest to my research work.

List of original publications

P1 Johannes Tuikkala, Laura Elo, Olli S. Nevalainen, Tero Aittokallio, Improving Missing Value Estimation in Microarray Data with Gene Ontology. *Bioinformatics* 22(5):566-572, 2006.

P2 Johannes Tuikkala, Laura L. Elo, Olli S. Nevalainen, Tero Aittokallio, Missing Value Imputation Improves Clustering and Interpretation of Gene Expression Microarray Data. *BMC Bioinformatics* 9(1):202, 2008.

P3 Johannes Tuikkala, Heidi Vähämaa, Pekka Salmela, Olli S. Nevalainen and Tero Aittokallio, A multilevel layout algorithm for visualizing physical and genetic interaction networks, with emphasis on their modular organization. *BioData Mining*, 5:2, 2012.

P4 Olli Luoma, Johannes Tuikkala, Olli S. Nevalainen, Accelerating GLA with an M-Tree. In *Proceedings of The Second World Enformatika Conference (WEC 2005)* (2):196-199, 2005.

P5 Kati Talvinen, Johannes Tuikkala, Marjukka Nykänen, Anssi Nieminen, Jorma Anttinen, Olli S. Nevalainen, Saija Hurme, Teijo Kuopio, Pauliina Kronqvist, Altered expression of p120 catenin predicts poor outcome in invasive breast cancer. *Journal of Cancer Research and Clinical Oncology* 136(9):1377-1387, 2010

Contents

Chapter 1

Introduction

Biological research has made important breakthroughs recently at an accelerating pace. The characteristic feature of the recent biological discoveries has been the massively increasing amount of new biological data produced through this research. One of the modern data collection methods of molecular biology has been the DNA sequencing technique. In the year 1977 researchers were able to read the DNA sequence of the bacteriophage ϕX174 containing 5,375 nucleotides [69]. The first complete genome sequencing was finished by 1995 resulting in about 1.8 million base pairs of nucleotide sequences of *Haemophilus influenzae* bacterium [29]. In the year 2003, the genome sequencing achieved its most anticipated goal when the whole human genome of about 3.3 billion base pairs was finally sequenced [20]. Today we have an easy access to over 190 billion base pairs of more than 300 000 organisms trough the GenBank database [11].

Concurrently with the development of the DNA sequencing, other high-throughput biological data acquisition methods were developed. Maybe one of the most important tools in these methods have been the gene microchips. The first commercial microchips were originally produced by Affymetrix in 1996 [48, 50]. They were able to screen the expression levels of about 1000 to 2000 genes in one experimental condition. Modern microchips are nowadays used to screen expression of thousands to tens of thousands of genes.

The very large amount of biological data yields a need for high performance data analysis and visualization methods. Thus, methods for analyzing and visualizing large-scale biological data have been objects of very active research from the late 70s to the present day.

Although, the data acquisition methods have become more and more accurate, there are limitations on the reliability of the gathered data. A typical limitation is caused by missing data values i.e. the data that have not been acquired or that are for some reason considered unreliable. There are numerous reasons for the existence of incomplete data in a biological research. For instance to keep the biological experiments at an affordable cost level, research instruments such as microchips have to by small in physical size. This is very analogical to digital photography where photons are gathered with an imaging sensor (typically CMOS type). The smaller the sensor the cheaper it is, but the acquired data become consequently less reliable.

Secondly, abundance of data and its typically very high dimensions cause logical problems since the human comprehension is generally limited to the first three dimensions. For instance it is not unusual to have a biological data set consisting of thousands of data vectors of a dimension greater than ten. In addition to human comprehension, the high dimensional data is a big challenge for the most statistical analyzing methods available. It is also easy to see that visualizing even three-dimensional data provides challenges, thus, it is no wonder that higher dimensions are still much harder to visualize.

Moreover, although we have currently large amounts of computational power available even in personal computers, most of the computationally complex algorithms are still much too slow to be used when the problem size is big enough.

1.1 Aim of the Thesis

This thesis considers ways to address the seemingly different needs on the analysis path from the raw biological data from the experiments to biologically interesting results. Data from genome-wide gene expression experiments are used as case studies. The missing data imputation belongs to the first analysis steps after the raw data has been acquired. One of the subsequent steps consists of clustering and visualization of the gene expression data. Finally, biologically relevant conclusions will be drawn from the data.

Objectives of the thesis are to study: (P1) how to improve the imputation accuracy with biological *a priori* information, (P2) how to evaluate the imputation accuracy by considering the stability of the clustering results (P3), how very large biological interaction networks can be visualized in a biologically relevant and computationally effective way, (P4) how to improve the computational complexity of a clustering algorithm with use of the M-tree indexing method, and (P5) how to carry out data analysis of the real clinical gene expression data. We aim to provide solutions to these topics such that the developed methods could be applicable also in the other research areas. While the application area belongs to the biological research, our approach is algorithmic research.

1.2 Structure of the Thesis

The rest of this thesis is organized as follows. We start from the biological background in Chapter 2, going shortly through the basic terms of cell biology and gene expression research. In Chapter 3 we present how gene expression is measured with the microarray techniques. We study the cDNA and oligonucleotide microarray techniques since these were the two most commonly used ones at the moment of writing the publications of this thesis. Chapter 4 presents approaches to dealing with the missing data. The imputation methods for missing data developed here aim to provide as complete and accurate data as possible for downstream analyses. Biological visualization methods are examined in Chapter 5. We show how to develop an efficient graph drawing tool suited for very big biological networks. The tool is designed to take care of the strength of the connections between nodes as described by external biological information that is available from public sources such as the Gene Ontology. In Chapter 6, the summary of each publication included in the thesis is presented. The conclusions are presented in Chapter 7. We briefly discuss also other related publications of the author at the end of the thesis.

Chapter 2

Biological background

The need for efficient computational tools for analyzing biological data is the basis of this work. Knowledge of the biological domain is needed when developing methods for data analysis. Therefore, in this chapter, we will go through some of the fundamental concepts of the biological background domain. The discussion is very brief, and it only lists the concepts used later in this study. First, in Section 2.1, we will explore the biological processes that are taking place in the cell. We will shortly explain how the cell produces proteins and other gene products through gene expression. Also, mechanisms, such as gene regulation and interaction, that may trigger these processes, are shortly described. The need to measure and quantify the products of these processes presupposes a basic knowledge about the fundamentals of the biotechnology and bioinformatics (Section 2.2). The main source for the basic concepts and definitions discussed here is the textbook of Watson et al. [90].

2.1 Biology of the cell

2.1.1 Structure of the cell

The *cell* is a basic structural and functional unit of all known living organisms: It is described as the smallest unit of life [2, 90]. There are organisms, such as bacteria, that consist of a single cell, and organisms, like human, that contain trillions of cells of which each one has its own specific function.

Cells are traditionally divided into *eukaryotic* cells of animals, plants, fungi etc. and *prokaryotic* cells of bacteria and archaes. Eukaryotic cells have a *nucleus* while *prokaryotic* cells do not have it. In this study we focus only on the organisms that consist of eukaryotic cells (*Eukaryotes*).

Almost all cells share some similar structures in their anatomy. We will focus here on the cells that contain genetic material called *DNA* (*Deoxyribonucleic acid*). Most eukaryotic cells contain the DNA in the cell's nucleus.

An example of a cell that does not contain a nucleus nor DNA is the red blood cell of human.

DNA is a long polymer of *nucleotides* containing all genetic information needed by the organism. DNA consists of a number of nucleotide pairs that form a structure called *double helix*. DNA's role is to preserve the genetic information of the organism. DNA is accessed when the double helix is opened totally or partially. When the double helix is opened the cell can produce an identical copy of itself, that is needed when the cell is *proliferating*. Partially opened DNA is needed when a sequence of DNA is transcribed to *messenger RNA* (*mRNA*) (*Ribonucleic acid*).

There are four different nucleotides in DNA: adenine (A), guanine (G), thymine (T), and cytosine (C). The double helix consists of two strands of the nucleotides such that a nucleotide in the first strand (such as C) has its opposite nucleotide (G) in the other strand.

A set of rules how nucleotides are arranged in the DNA, is called the *genetic code* [2]. Nucleotides are arranged to a sequence of three nucleotides triplets called *codons*. A codon represents the instruction according to which an amino acid is selected when a protein molecule is formed from a gene [2]. For example the codon CCG in a gene will correspond the Proline amino acid in a protein.

Figure 2.1 presents a scale from a single cell (a), to its nucleus containing the *chromosomes* (b). Chromosomes contain tightly packed DNA (c), where the *genes* are located (d). A gene is composed of nucleotides (e).

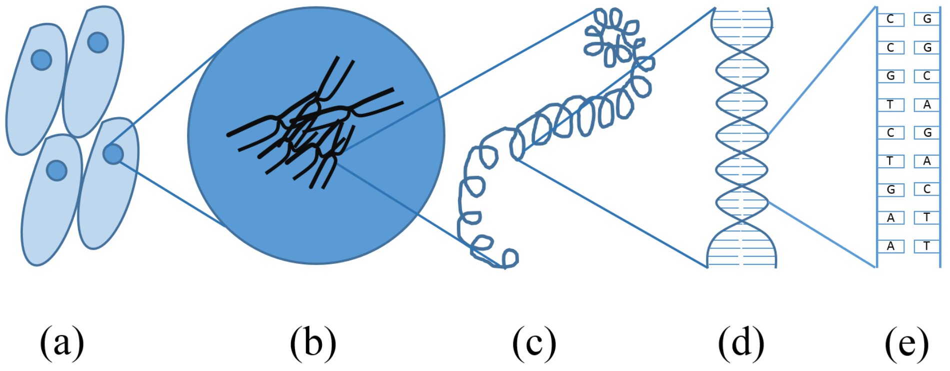

Figure 2.1: View from a cell (a) to a small part of the gene (e) [2].

In Fig. 2.1, the structure of a cell is highly simplified: only the *cell membrane* (i.e. the outer boundary of the cell) and nucleus are shown. A more detailed illustration of an (animal) cell structure is presented in Fig. 2.2. The figure is adapted from to [90]. The function of the *cell membrane* is to separate cell's interior from the outside environment and act as an interface between the cell and its outside [2]. The nucleus contains the DNA and *nucleolus*, which is used to assemble and transcribe *ribosomal RNA* (*rRNA*). The rRNA is the building block of the ribosomes. The *endoplasmic reticulum* is a labyrinth like structure of membrane; its role is to provide a place where most of the protein synthesis occurs with the help of the ribosomes. The *ribosomes* are the active molecular machines whose role is to synthesize the proteins by linking amino acids together according to the instructions given by messenger RNA molecules.

The *cytoplasm* (i.e. cell medium) contains also other organelles such as mitochondria and Golgi apparatus: the *mitochondria* produces molecules called *adenosine triphosphate* (ATP) which function as a source of chemical energy in various processes in the cell. The function of the *Golgi apparatus* is to package the proteins before they can be sent outside of the cell. We suggest the reader to view the textbook of Watson et al. [90] for a more detailed description.

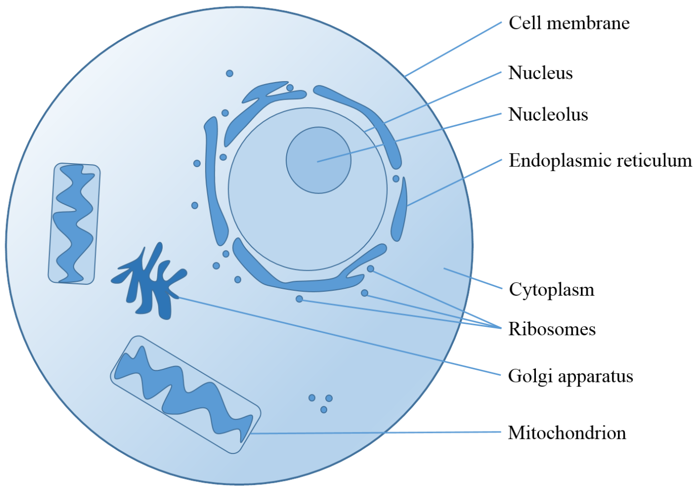

Figure 2.2: General structure of an animal cell [90].

2.1.2 Biological processes

The two important biological processes of the cell, mentioned here, are the gene expression, and DNA replication. Gene *expression* is a process in which a particular region (gene) of the DNA is copied (in a *transcription* sub-process) into a RNA which is then either *translated* to a protein, or for some genes, is the final gene product itself [2]. The produced RNA can be translated to an amino acid sequence in any of three possible *reading frames*, depending on the from which of the first three nucleotides the translation is started [2]. The translation begins with the *start codon* (typically AUG) and ends when the *stop codon* of the RNA is reached [2].

The *expression level* of a gene can be quantified from the abundance of the messenger RNA molecules produced by the gene [51]. *DNA replication* is the process where the original DNA molecule is replicated into two identical copies. DNA replication is one of the key parts of the so-called *cell-division cycle*, where a cell is divided into two cells.

Other important biological processes are regulation of the gene expression and interactions between proteins and/or genes. *Genetic regulation* and *interaction* processes are commonly described by a *genetic regulatory network* (GRN) in which genes interact with each other and other material present in the cell. Genetic regulation is a set of processes that control how, when, and in which amounts genes are expressed in the cell. In these processes *transcription factors* (TF) such as the proteins and other gene products, control the activation and suppression of the expression of other genes. These processes are organized in networks called genetic regulatory networks.

A genetic regulatory network can be initiated for instance by an environmental change of the cell's temperature that triggers the expression of certain genes. Then, expression products of these genes typically will trigger other genes to be expressed. This will continue, and can trigger more gene expression in the cell and the cells nearby it.

2.2 From biochemistry and molecular biology to bioinformatics

Many biological processes that operate on the molecular level, are interrelated to one another. The roots of *bioinformatics* lie in the biological sciences like the biochemistry and molecular biology [37]. The term bioinformatics was introduced in 1970 by Paulien Hogeweg as the study of informatic processes in biotic systems [36]. The definition itself brings us close to mathematical sciences like statistics and information sciences. One can consider the bioinformatics as an example of successful inter-disciplinary research area that is even more important today.

In this study we are interested in the challenges of the abundance of data (so called data deluge) and data incompleteness which are products of biotechnological research devices such as gene microchips. The abundance of data is a reason why computer science is needed in bioinformatics. The data incompleteness is due to the fact that we are quantifying real life biological systems where lots of distractions are present.

We can perceive the amount of numerical data generated by biotechnological research instruments by looking at the products which are available from the biggest industrial bioinformatics companies such as Affymetrix and Illumina. By an Affymetrix gene chip a small research team could easily acquire tens of millions of signal values in one week from a human genome wide study using Human SNP (*single-nucleotide polymorphism*) Array 6.0 [55].

Gene expression profiling is used to find patterns of gene expression. These patterns can be e.g. a temporal pattern of gene expressions during the measured time span of the study in question, or physiological patterns where the expression of the genes is measured in various physiological conditions. Gene expression profiling is a widely used molecular biological technique for studying the functional roles of genes in the cells. The data used in this thesis originate from this kind of profiling studies.

Chapter 3

Gene microarrays

The use of *gene microarrays* has been a popular technique for quantifying the relative expression of thousands of genes in many different experimental conditions [51][P1]. Gene microarrays are widely used for gene expression profiling since they provide very high throughput for analyzing expression profiles of thousands of genes simultaneously.

The first modern uses of gene microarrays for gene expression profiling date back to 1995 when Schena et al. [71] were able to develop a high throughput method for measuring the gene expression of 45 genes of Arabidopsis in parallel. Before that, in 1982 [7], the gene microarray technology was evolved from the Southern blotting method that is used to identify a specific sequence of DNA from a studied sample [75]. For this reason the basic steps of this method will be shortly reviewed below.

Southern blotting is a rather tedious process which provides reliable information of the existence of the studied DNA sequence in the given samples. The method is based on (i) the separation of differently sized DNA fragments (cut by restriction enzymes) of given samples with the use of electric field. This part is called electrophoresis. The *electrophoresis* step produces patterns of DNA groups of the samples.

The patterns of DNA groups are then transferred into a sheet of nitrocellulose paper. Then (ii) plenty of *single-stranded DNA probes* of the studied sequence is allowed to react with the sample groups to see how it will anneal with them. The more annealed DNA is found from groups of a sample, the more certain it is that the studied DNA is found from the sample. Quantification of the annealed DNA is made possible by labeling the probe sequences by a radioactive or (more recently) fluorescent marker [13]. After the specific annealing time the remaining probes are washed away, and the sheet of nitrocellulose paper was placed on an x-ray film in case of radioactive markers. When the x-ray film was developed the annealing profiles are obtained for all the samples.

In this study, we are especially interested in two different types of microarrays: spotted *cDNA microarrays* and in situ synthesized *oligonucleotide arrays*. Both of these can be used for gene expression profiling [25, 40]. It should be mentioned also, that the microarray technology has been used for various other purposes too such as single nucleotide polymorphism (SNP) detection [32] and identification of protein binding sites of DNA-binding proteins [66], but those are not covered in this study.

3.1 cDNA microarrays

The cDNA microarrays are the first generation of gene microarrays. The basic idea is to do multiple simplified and miniaturized Southern blotting experiments in parallel. In a Southern blotting experiment the existence of a single DNA sequence (probe) in the sample is studied, but in a cDNA microarray it is possible to have thousands of probe cDNA sequences of genes in the same experiment and study how these genes are expressed in the studied sample. Thus, with gene microarrays one has a way to obtain a whole gene expression profile of each sample.

3.1.1 Constructing cDNA microarray

A cDNA microarray is constructed (prepared) on a microscope glass slide with an array of tiny *spots*. In our simplified example the array structure consisting of M (commonly from hundreds to thousands) rows and 3 replicate columns for each row is assumed. A small amount of purified and cloned cDNA probes of the studied genes are placed on these spots such that the three spots on a single row contain similar cDNA probes. The spots on the second row respectively contain cDNA of another studied gene. In the end, we have cDNA content of M genes 3 times in the individual spots of the array (see Fig 3.1; Microarray preparation). In practice the ordering of the spots on the microarray might not be so straightforward as in our example.

Now, assume that the researcher wants to study how the gene expression profiles of a tumor sample (such as *Colorectal adenocarcionoma* [77]) differ for different severity of the tumor. The motivation could be to find a set of genes which are participating in the tumor development, such that it is possible to prevent the deathly metastasis of the cancer. In a case such as this the mRNA of the tumor samples will be extracted and purified, and a cDNA is generated from it via reverse transcription. Then, the cDNA of the studied sample (such as metastasis tumor) and the reference sample (non-metastasis tumor) are labeled with different fluorescent colors (such as Cy5 red and Cy3 green).

A small amount of both sample is placed on the columns of the cDNA microarray spots, such that each spot of a different cDNA probe is let to

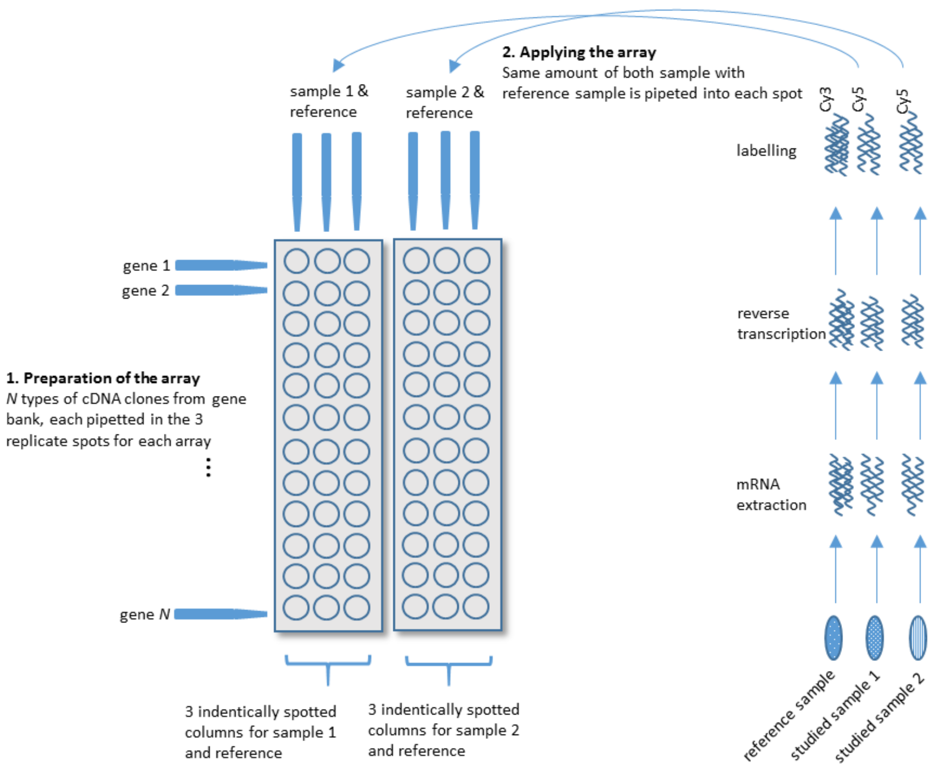
2. Applying the array
Same amount of both sample with reference sample is pipeted into each spot
sample 1 & reference
sample 2 & reference
Cy3
Cy5
Cy5
labelling
gene 1
gene 2
reverse transcription
1. Preparation of the array
N types of cDNA clones from gene bank, each pipetted in the 3 replicate spots for each array
mRNA extraction
gene N
3 indentically spotted columns for sample 1 and reference
3 indentically spotted columns for sample 2 and reference
reference sample
studied sample 1
studied sample 2

Figure 3.1: Steps of the preparation and applying the cDNA microarray [84]

react with the sample's mRNA. See Fig. 3.1 for a graphical representation of the microarray slide applying process. This is a simplified example where two arrays are used to study two different samples (such us two patients); in practice the number N of samples is from 3 to hundreds but could be more, even thousands.

After the mRNA of the studied and reference sample's have been added into spots, they are allowed to hybridize a specific time (e.g. overnight) at about 65 degrees of Celsius. In hybridization two strands of nucleic acids are bound (annealed) together. If the gene corresponding to the probe cDNA is expressed in either of the samples, there will be labeled cDNA that will hybridize with the cDNA probe. After hybridization the remaining unreacted material is washed away and the color of each spot measured using two different wavelengths of a laser beam.

Since the samples are labeled with a different fluorescent color, the level of gene expression can be measured from the resulting color of the hybridized mRNA. If, for instance, the mRNA from a metastasis tumor is labeled with Cy5 and the reference sample is labeled with Cy3. Then, if the resulting color of the spot is red instead of green we could deduce that the corresponding gene is more actively expressed (*overexpressed*) in the studied sample than in the reference.

The color measurement process is called *scanning*. The scanning is done with an automated *microarray scanner*, in which the color of each spot is recorded with a photomultiplier tube [84]. The colors are subsequently converted to numerical values.

The process described above is called *two-channel detection* since each spot contains information about the expression level of the gene in both the reference and the studied sample. In *single-channel detection*, only one sample is allowed to hybridize with the probe cDNA [61]. Here we are mainly studying the two-channel detection if not stated otherwise.

3.1.2 cDNA microarray data acquisition

Numerical microarray data are acquired from the image that is constructed from the output of a microarray scanner. An example of a recorded image of microarray spots is shown in Fig. 3.2 [84]. In the figure the spots are already *gridded* (i.e. separated) from each other by using the vertical and horizontal *projections*. The projections are got by summing the pixel values of each pixel row (column) to a vertical (horizontal) axis and drawing the grid lines between the minima of the projected values [84].

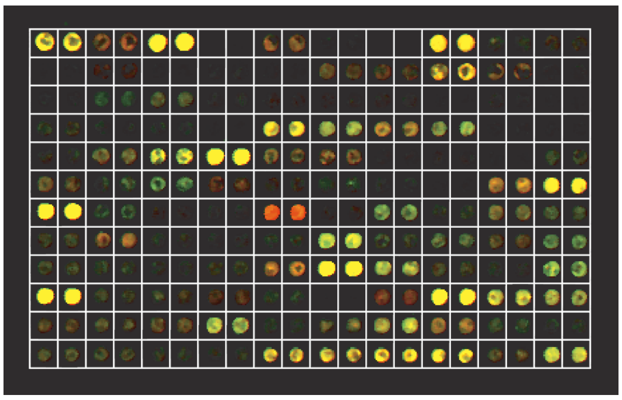

Figure 3.2: Gridded cDNA microarray spots [84]. [1]

After the gridding is done the rough spot locations are detected. The more accurate spot detection is called *segmentation*. The purpose of the segmentation is to detect the exact areas of single spots where the hybridization has happened. The segmentation can be done by drawing fixed or adaptive circles or non-circular contours around each spot [84]. The segmented spot

[1] Copyright All rights reserved by CSC - IT Center for Science Ltd, Finland. (The PDF version of this book or parts of it can be used for academic non-profit purposes, provided that this copyright notice is included. This publication may not be sold or included as part of other publications without permission of CSC.)

is converted to a numerical value I by subtracting the average $\hat{I}$ of the pixel values (*intensities*) inside the segment from the background intensity I_B outside the segment:

$$I = \hat{I} - I_B \tag{3.1}$$

The final data value is calculated as a logarithm of the ratio of the intensity of the studied sample I_R and the intensity of a reference sample I_G. Here R and G denote red (Cy5) and green (Cy3) fluorescent colors typically used for tagging the studied and reference cDNA samples.

To maintain the reliability of the data, a number of separate quality control steps are also applied to segmented spots. In this phase, problematic spots are marked as unreliable, and they will yield missing values. For instance, a problematic spot could be a spot with very high variety in the pixel intensities inside it, or two spots which lie too close to each other [92].

In this thesis, we will represent the data set (matrix) produced with a microarray experiment with the following notation where M is the number of genes (rows) and N is the number of experimental conditions (columns).

$$\mathbf{A} = (a_{ij})_{M \times N} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{MN} \end{bmatrix} \tag{3.2}$$

A single row and column vector of $\mathbf{A}$ are notated as $a_i = [a_{i1}, \cdots, a_{iN}]$ and $a_{\cdot j} = [a_{1j}, \cdots a_{Mj}]^\intercal$ respectively. Row vectors of $\mathbf{A}$ are also called data objects, genes, or gene expression profiles, depending on the context.

3.2 Oligonucleotide microarrays

One main difference between the cDNA microarrays and the oligonucleotide arrays is between the probes used. While in the cDNA array the cloned cDNA of known genes are used as probes, in the oligonucleotide arrays the probes are synthesized nucleotide by nucleotide on the arrays. The synthesized oligonucleotides are short, typically 10-30 base pairs, nucleotide sequences that are known or predicted to match specific regions of the DNA called open reading frames (ORF) [84]. An ORF is a continuous nucleotide sequence without stop codons (see Section 2.1.2) [2].

In situ synthesizing of probe nucleotides provides an advanced way to have quality control in the arrays. For example GeneChip arrays of Affymetrix contains additional test probes called *mismatch* (MM) for each *perfect match* (PM) probe which contain one nucleotide mismatch in the middle of the probe [84]. The function of the MM probe is to provide a way to measure *non-specific hybridization* [9]. Affymetrix GeneChip arrays have 10-25 probes (from various parts of the gene's DNA) for each studied gene. In contrast, Illumina arrays have tens of copies of the same probes randomly placed on the array [9].

The second difference between cDNA and oligonucleotide arrays is in the arrangement of the research protocol. In cDNA microarrays one typically has both the studied sample and the reference sample hybridized simultaneously on the spots. In contrast to that, in oligonucleotide array single RNA is hybridized on the array and the studied samples are compared to the reference sample computationally [84].

The first uses of these in situ synthesized arrays are from 1992 when Southern et al. [76] were able to in situ synthesize and stably attach oligonucleotides to a glass surface. Later, in 1994, Pease et al. developed the photolithographic process for synthesizing hundreds of tightly packed oligonucleotide probes on a small array [62]. Pease et al. used their array for DNA sequence analysis.

Later on in 1996, Lockhart et al. [50] utilized in situ synthesized oligonucleotide arrays for gene expression analysis. They developed an approach where more than 16,000 synthetic oligonucleotides probes were in situ synthesized on a small array that can be scaled to contain tens of thousands of probes [50].

In 2002, Nuwasysir et al. [57] created an array from 3240 genes of mouse's liver samples where for each gene the array contained 20 different probes. Using the *quantitative PCR* (polymerase chain reaction) validation they were able to show that the used maskless photolithography technology produced high quality microarrays for gene expression analysis [57].

The data acquisition and preprocessing steps of oligonucleotide arrays and cDNA arrays have many differences which are not covered in this study. In the following chapters it is assumed that the data have been preprocessed such that unreliable measurements are marked as missing values.

3.3 Handling microarray data

3.3.1 Data preprocessing

Microarray data preprocessing is an important step in the microarray data analysis pipeline that should be done before any further analysis steps. In this step the statistical characteristics of the data are verified. Since many of the data analysis methods rely on the assumption of a certain statistical distribution of the data, the distribution of the data should be verified [84].

Another typical preprocessing step is the detection of the unreliable data. Unreliable data values can have high variation between technical replicates, or have lower intensity value than the background, or may not fit to the assumed statistical model of the data, i.e. are so-called outliers [84].

One typical preprocessing task is to remove the genes for which the expression remains constant over all of the conditions under analysis. This can be done e.g. by filtering out genes a_i in which the difference d_i between the maximum and the minimum expression values is smaller than a constant λ [77, 78, 80], more specifically in (3.3) the value of λ could be like 1.5 [P2].

$$d_i = \arg\max_j a_{ij} - \arg\min_j a_{ij} < \lambda \tag{3.3}$$

Finally, the preprocessing step typically includes the normalization and standardization of the data. The purpose of the normalization is to remove systematic variation of the data so that meaningful biological comparisons can be made [4, 64]. The standardization allows us to compare and combine the results of the different microarray experiments [84].

There are many ways to normalize the gene expression ratio data. A simple and intuitive way is the log-transformation [P5][77, 78]. It transforms expression ratios such that the expression ratios of underexpressed genes will be smaller than 0 and correspondingly expression ratios of overexpressed genes will be greater than 0 [84]. Another approach, in case of two-channel detection, is called *total intensity normalization* [64] where the normalization factor

$$N_{total} = \frac{\sum_{i=1}^{n} I_{Ri}}{\sum_{i=1}^{n} I_{Gi}} \tag{3.4}$$

is used to scale one or both intensities such that the normalized expression ratio becomes, in the case of scaling only the green intensities,

$$I_i = \frac{I_{Ri}}{I_{Gi}} = \frac{I_{Ri}}{N_{total} \cdot I_{Gi}}, \tag{3.5}$$

where n is the number of spots and I_{Ri} (I_{Gi}) is the intensity value of the red (green) channel in the ith spot. The result of this normalization is that the mean expression ratio is 1 [64].

The problem with log-transformation and total intensity normalization is that they cannot remove the expression ratio's possible systematic dependency on intensity which can appear e.g. as extra variation at low intensity spots and curvature in a *ratio-intensity* (R-I) plot [64, 84]. An R-I plot is produced by transforming the intensity values such that $\log_{10}(R_i \cdot G_i)$ is used as the x-coordinate, and $\log_2(R_i/G_i)$ as the y-coordinate of the new plot. *Lowess normalization* (locally weighted scatterplot smoothing) is a procedure which can be used to correct the systematic intensity dependent deviation. It corrects this deviation by performing a local weighted linear regression as a function of intensity (log_{10}) and subtracting the estimated ratio (log_2) from the observed ratio for each spot [64].

An example of typical standardization is to make the average expression value of each experimental condition to 0 and the standard deviation to 1. This can be achieved by subtracting the average expression value

$$\mu_j = \frac{1}{M}\sum_{i=1}^{M} a_{ij} \tag{3.6}$$

of each experiment condition from all the gene's expression values and dividing the gene's subtracted expression value by the experiment condition's standard deviation:

$$\sigma_j = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(a_{ij} - \mu_j)^2}. \tag{3.7}$$

The standardized value will then be

$$\acute{a}_{ij} = \frac{a_{ij} - \mu_j}{\sigma_j}. \tag{3.8}$$

3.3.2 Fundamentals of data analysis

We will briefly summarize here the basic concepts of microarray data analysis methods. In itself the raw data is useful only as a valuable resource [12]. The purpose of data analysis is to turn the raw data to information and finally to knowledge. There are a large number of different data analysis methods developed by the statisticians and computer scientists. The most common methods of microarray data analysis can be roughly classified into the following groups: classification, regression, clustering, and visualization [12]. The classification and regression are part of the *supervised learning* methods whereas clustering is one of the *unsupervised learning* methods.

The *classification* is a task in which the algorithm called as *classifier* is trained with data objects a_i with known classification y (*training data*) such that the algorithm can be later used to predict the probable class y of a new data object x_* [12]. The use of training data is a common approach of all supervised learning methods. The number of possible classifications of a data object is typically finite, such as $y \in$(yes, no) or $y \in$ (1, 2, 3, 4). An example of classification could be a classifier that predicts the severity class (lethal, non-lethal) of a tumor based on the microarray data experiment performed on the tumor sample [P5].

One of simplest classifiers is the *k nearest neighbor classifier* (*k-NN*) [21]. The classification of k-NN is based on the majority vote of k nearest (most similar) known classification examples a_i of x_* [33, 79].

The similarity d (or dissimilarity) between the x_* and a_i can be measured by different similarity (dissimilarity) measures. Dissimilarity measures are often called distance functions or metrics. Perhaps the best known distance metric is the Euclidean distance d_E. The Euclidean distance between the vectors a_i and x_* is defined as below where a_{ij} is the jth component of vector a_i and x_{*j} is the jth component of x_* [79].

$$d_E = \sqrt{\sum_{j=1}^{N} (a_{ij} - x_{*j})^2} \tag{3.9}$$

Figure 3.3 shows a simple example of 7 data objects with known classifications (4 for class "filled circle" and 3 for class "open circle"). These seven data objects and one new with "+" symbol are plotted onto xy-plot. Let $k = 3$ be the neighborhood size. Lines in the plot indicate the three closest data objects of the new object. Since the majority (2) of the three nearest neighbors belongs to class "open circle", also the new object is classified into that class.

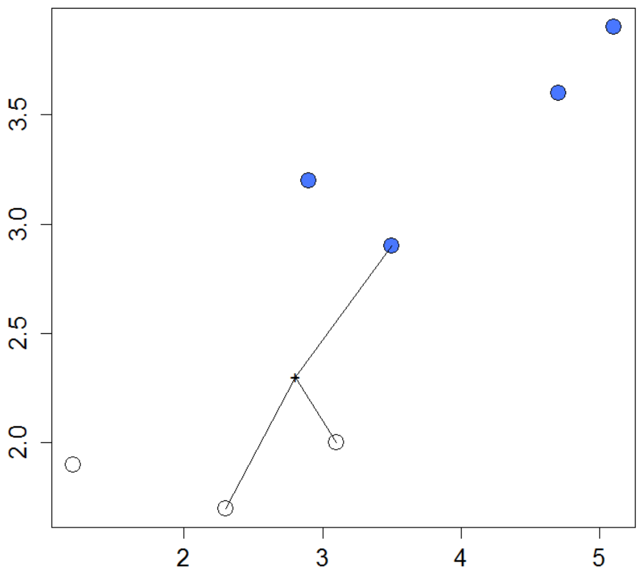

Figure 3.3: Classification of data object "+" with a 3-NN classifier.

More formally, let us assume that there are Y different classes.

k-NN algorithm

1. Find the k-neighborhood $N_k(x_*)$ of object x_* (see below).
2. For each class c_i, $(i = 1, .., Y)$, the number of occurrences $|k_i|$ in the $N_k(x)$ is calculated such that $\sum_i |k_i| = k$.
3. Object x_* is classified to class c_i with $\arg\max_i |k_i|$ in $N_k(x_*)$.

Neighborhood $N_k(x_*)$ of sample x_* is defined by the k most similar known classification examples a_i which are found by sorting a_i's by their distance from x_* and choosing only the first k classification examples with smallest distance from x_* [33].

Regression [31] acts as classification but instead of a finite number classes it will produce continuous value classification with the infinite number of possible results. Thus, regression is used when the classification result is in the real value space $\mathbf{R}$. An example of regression task could be predicting the probability $\{p \in \mathbf{R} \mid 0 \leq p \leq 1\}$ of patient survival from the cancer with the microarray analysis of patient's blood sample [56]. The regression can also be used as a classification method by discretization of the continuous result into a finite number of classes.

Adapting Hastie et al. [33], a *linear regression* model is formulated in the following way. Given an input a_i, the output $\hat{y}$ can be predicted via model

$$\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{N} a_{ij}\hat{\beta}_j, \tag{3.10}$$

where $\hat{\beta}_0$ is called the intercept. It is possible to simplify the model by including constant 1 in the a_i and including $\hat{\beta}_0$ in the vector of coefficients $\hat{\beta}$. The model can then be written in the vector form as an inner product

$$\hat{y} = a_i^\intercal \hat{\beta}. \tag{3.11}$$

This linear model could be fitted to a set of training data e.g. by the *least squares* method [46] where the coefficient of β is picked to minimize the residual sum of squares between the known y_is and predicted $\hat{y}$s

$$RSS(\beta) = \sum_{i=1}^{M} (y_i - a_i^\intercal \beta)^2, \tag{3.12}$$

which is in matrix notation

$$RSS(\beta) = (\mathbf{y} - \mathbf{A}\beta)^\intercal (\mathbf{y} - \mathbf{A}\beta), \tag{3.13}$$

where $\mathbf{A}$ is the data matrix and $\mathbf{y}$ is a vector of outputs in the training data. When differentiating with respect to β, the normal equation is got

$$\mathbf{A}^\intercal (\mathbf{y} - \mathbf{A}\beta) = 0. \tag{3.14}$$

The unique solution is the following, if $\mathbf{A}^\intercal \mathbf{A}$ is nonsingular

$$\hat{\beta} = (\mathbf{A}^\intercal \mathbf{A})^{-1} \mathbf{A}^\intercal \mathbf{y}. \tag{3.15}$$

Then the predicted value can be calculated for a new input vector x as $\hat{y} = x^{\intercal}\hat{\beta}$.

Data *clustering* (cluster analysis [82]) belongs to so called exploratory data analysis methods. It is typically applied to a data set to reveal possible intrinsic patterns and structure of the data set [79]. Clustering is called an unsupervised learning method since, unlike classification, it does not use learning data [12]. Clustering tries to group the data set into a finite number of subgroups (*clusters*) such that data objects inside a cluster are on an average more similar to each other than to any data object outside the cluster. A typical example of this method is the hierarchical clustering of gene expression profiles of microarray data with purpose of finding interesting subgroups of genes from the data set [74].

For an example of the clustering methods let us see how the k-means clustering [53] (also called as *generalized Lloyd algorithm* [47] and *ISODATA algorithm*[8]) works. The idea of k-means clustering is to divide (*partition*) the data objects a_i of $\mathbf{A}_{M\times N}$ into k clusters represented by *centroids* c_j (center vector) of the clusters. The algorithm consists of two steps: (A) partition and (B) centroid calculation, which are iterated until the algorithm converges. In step (A), each data object a_i is assigned to the cluster j for which the centroid c_j is closest to a_i. Then in step (B), the algorithm recalculates the centroid of each cluster as an average of the data vectors that are assigned to it in step (A) [12].

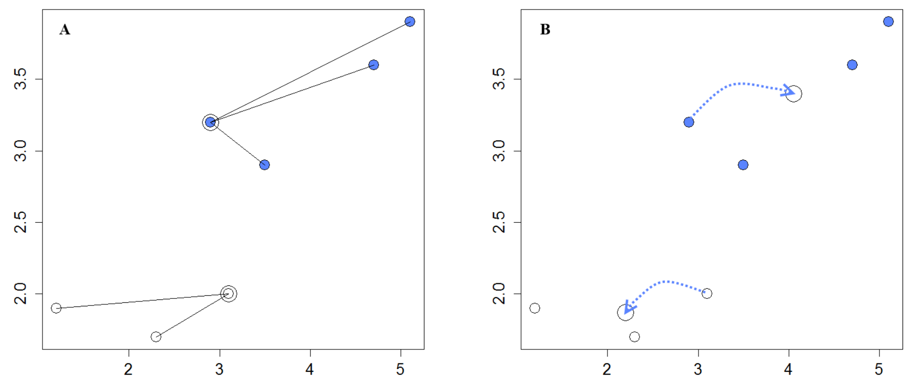

Figure 3.4: Example of k-means clustering; partition step (A), and centroid calculation (B). Bigger open circles denote cluster centroids. Straight line segments in A mark the data objects that are partitioned into the corresponding centroid. Arrows in B show how the new cluster centroids are calculated as averages of data objects that were assigned to the corresponding clusters.

A simplified example of k-means clustering of 7 data objects is presented in Fig. 3.4. We use $k = 2$, and two data objects are randomly selected for initial centroids (bigger open circles). Then, all the objects are assigned into the closest centroid (step A). Finally, in step (B) each centroids is recalculated from its data objects (step B). In this example k-means clustering stops with one iteration since centroids are not moved anymore in the following iteration.

The *visualization techniques* are used in many phases of the data analysis pipeline. First, fundamental visualizations such as *histograms*, *heat maps*, and *scatter plots* are applied to verify the data quality and to see the basic properties (like distribution) of the data [12]. Then, one could apply some dimension reduction algorithm such as *principal component analysis* (PCA) [38] to see how the data is projected into 2 dimensional space. Further on, other data analysis tools such as classification or clustering need visualizations to help interpretation of the results. Later on in this thesis we will study the visualization of biological interaction networks too.

Chapter 4

Dealing with incomplete biological data

The biological and technical backgrounds of the research of the present work were discussed in the previous chapters. In Chapter 3, a research instrument called gene microchip was shortly reviewed. Our interest in the gene microchips is in their capability of providing a high throughput method for acquiring massive amounts of data at once for biological or medical research studies. From the point of view of a computer scientist, gene microchips provide many fascinating research questions. In this study, the following three challenges are considered: how to cope with the incomplete data of a microarray experiment, how to visualize biological results of the experiments, and how to handle the high amount of data with the slow algorithms.

In this chapter, we shall discuss what the missing values are, why they cause problems, and how to cope with them. Fundamentally, the missing values are data elements for which there is no observed signal or the signal is suspected to be unreliable. The treatment of missing values is a well known statistical problem since 1987 when Little and Rubin wrote their textbook of Statistical Analysis with Missing Data [49].

In the context of statistics, there are basically three types of missing values proposed by Little and Rubin [49] (missing data mechanisms): the values that are *missing completely at random* (MCAR), the values that are *missing at random* (MAR), and values that are *not missing at random* (NMAR) [34, 49].

To understand these different missing value mechanisms better, a missing data indicator matrix [49] $\mathbf{M} = (m_{ij})_{M \times N}$ is used to indicate values of $\mathbf{A}$ that are missing. Value $m_{ij} = 1$ if a_{ij} is missing, otherwise $m_{ij} = 0$. Suppose that it is possible to divide the original data into the missing part $\mathbf{A}_{miss}$ and the completely observed part $\mathbf{A}_{obs}$. The missing data indicator $\mathbf{M}$ is treated as a random variable which has a probability distribution that is controlled by the unknown parameter ϕ [49]. Then, according to Little and Rubin [49], the data is of MCAR-type if $\mathbf{M}$ does not depend on $\mathbf{A}$, i.e. $f(\mathbf{M}|\mathbf{A}, \phi) = f(\mathbf{M}|\phi)$. The data is of MAR-type if missingness depends only on the observed part $\mathbf{A}_{obs}$ of $\mathbf{A}$, i.e. $f(\mathbf{M}|\mathbf{A}, \phi) = f(\mathbf{M}|\mathbf{A}_{obs}, \phi)$. If the missingness depends on the missing part $\mathbf{A}_{miss}$ of the data, the data is of the NMAR-type.

An example of a MCAR missing value in the microarray data could be for example a dust particle that prevents receiving of a reliable signal value for a particular data point. MAR missing values allow some non-randomness which does not depend on the underlying missing data itself. For instance, the cause of a MAR missing value could be that some of the probe cDNA clones are not available for the microarray experiment because those were run out in the previous experiment. But if the existence of a missing value depends on the underlying value itself, the data is of the NMAR-type. An example reason for NMAR could be a bug in the scanning software which will report spots with very high intensity as missing or unreliable.

4.1 Causes of incomplete biological data

In the context of gene microarrays, the missing values can occur in various phases of the microarray study pipeline. Since each experiment contains lots of manual laboratory work before the actual microarray experiment, there is a possibility for human mistakes even before the actual microarray is prepared. The array itself might contain dust, scratches or some factory defects. Considering that microarrays themselves are small (typically glass slide) instruments containing a lot of very small spots from which the signal values are measured, it is very probable that some amount of noise exists in the observed data. An example of noise is the substantial cell-to-cell variation of gene expression even between two genetically identical cells [63].

Speaking of noise itself, it is a well-known problem for every digital photographer. In digital photography the noise problem is addressed basically in three ways: (1) with noise reduction algorithms, (2) by sampling multiple photos together, and (3) with bigger, more sensitive and thus more expensive imaging sensors. In biological context, there are almost exactly the same options available.

As we approach the missing value problem using tools of algorithmic computer science, we will focus the interest on the approach (1) presented above. The algorithms designed for compensating, augmenting or replacing the missing values in data are called as *missing value imputation algorithms.* In microarrays, the approach (2) could be utilized for instance by making the arrays to contain multiple redundant experiments (like in Figure 3.1) or making multiple identical microarray studies. The approach (3) is typically

	0	10	20	30	40
YHR007C	0.89	0.84	0.76	1.14	1.19
YBR218C	0.85	0.81	1.30		1.34
YAL051W	1.03	0.76	0.75	0.85	0.89
YAL053W	0.87	0.75	0.85	1.00	
YAL054C	1.59	1.14	1.00		1.14
YAL055W	1.47	1.00		1.09	1.04

Table 4.1: Small subset of data from the study of DeRisi et al. [23]. The four empty slots represent missing values.

addressed using more precise laboratory methods, such as quantitative PCR, for verifying the most important results of the faster and cheaper microarray studies.

4.2 Missing value imputation

The missing value imputation of gene expression data is a widely studied research problem since about the year 2000. It is a process where missing data values are replaced with the estimates determined from their supposed values. One of first surveys of different imputation methods on gene expression data is by Troyanskaya et al. [81]. They proposed several imputation methods for missing gene expression data including the k-NN imputation presented in Section 4.2.2. Before that, the missing values were typically either excluded, zero imputed or average imputed (Section 4.2.1) [81]. In 2004 De Brevern et al. studied many public microarray data sets and observed the wideness and severity of missing values to the methods used for further analysis of data [22].

To demonstrate how the missing value imputation methods work, a very small part of data obtained from a real microarray study is used (see Table 4.1). The data originates from a study of the metabolic and genetic control of gene expression of budding yeast [23].

The table shows the gene expression profiles of 6 genes over 5 different time points during the metabolic shift from yeast fermentation to respiration. This process is called the *diauxic shift*. Each value represents the expression ratio between the reference genes (from the yeast sample that was not in the diauxic shift phase) labeled with Cy3, and the genes from the yeast sample at different time points of the diauxic cycle labeled with Cy5. The values close to 1 are supposed to represent a situation where the gene from the reference sample is expressed similar with the gene of the studied sample. If the value is greater than 1, the gene is supposed to be overexpressed in the studied sample. Similarly the values below 1 are from genes underexpressed in the studied sample. The empty four slots in the Table 4.1 represent missing values.

One typical task to do after obtaining the data values such as those in Table 4.1, is to find the set of similarly expressed genes. In the simplest form this could be solved by calculating the correlation between the rows of the table. But how to calculate the correlation coefficient when there is missing values present. Basically, there are three choices, (1) rule out all the rows or columns containing a missing value, (2) adjust the formula of calculating the correlation coefficient, or (3) impute the missing values of the data table.

Method (1) causes clearly a massive loss of data since in the worst case there would be one missing value in each row and column [22]. Method (2) is a much better choice, but what about if we would like to use another formula for calculation of the expression profile similarities? It is rather obvious that the method (3) is the most straightforward option here; it will produce a complete data matrix that is ready for any known statistical analyses.

In the following Sections 4.2.1 to 4.2.3 a number of different approaches to applying the imputation on data sets containing missing values will be studied. The discussion will start with the simplest approaches and then proceed into more specific ones.

4.2.1 Average imputation

In the imputation methods based on the averaging, a statistical average value is used as a substitute for the missing values. In the simplest form, the value zero (0) is used as an imputation value. This is of course appropriate only if the expected expression value of the genes is close to zero.

More typically either the average of the whole data set is used or the average of the row or column of the missing value is used. In Table 4.2, the missing values are imputed as column averages.

The mathematical formula of average imputation could be defined as in (4.1) (row average) and (4.2) (column average) where a_{op} is missing and other values are present.

$$\hat{a}_{op} = \frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq p}}^{N} a_{oj}, \tag{4.1}$$

or

$$\hat{a}_{op} = \frac{1}{M-1} \sum_{\substack{i=1 \\ i \neq o}}^{M} a_{ip}. \tag{4.2}$$

	0	10	20	30	40
YHR007C	0.89	0.84	0.76	1.14	1.19
YBR218C	0.85	0.81	1.30	**1.02**	1.34
YAL051W	1.03	0.76	0.75	0.85	0.89
YAL053W	0.87	0.75	0.85	1.00	**1.12**
YAL054C	1.59	1.14	1.00	**1.02**	1.14
YAL055W	1.47	1.00	**0.93**	1.09	1.04

Table 4.2: Column-average imputed example data. The imputed values are presented with bold.

The first formula is used if the missing value of row o is imputed with the average of the existing values of the row. The second formula uses an average of the existing values of column p to impute its missing one. It is possible to generalize these functions for cases where there are multiple missing values in row o or in column p.

At the time point 30 there were two missing values (for genes YBR218C and YAL054C) and since the average of the non-missing values of the column is used, both imputed values are the same (1.02). The imputation of genes YBR218C and YAL054C will be studied in the following chapter.

4.2.2 Local average imputation

Let us look closer the gene expression profiles of the genes YBR218C and YAL054C in Table 4.1. Both of these contain a missing value at the same time point (30). In Section 4.2.1 (Table 4.2), these missing values were imputed as the same average value. Figure 4.1 shows plots of the expression profiles of all the genes in Table 4.1.

The gene expression data are presented as a graph of gene expression profiles a_i. The genes that contained a missing value at time point 30 are plotted with dashed lines to distinguish them from the other profiles. To calculate the similarities such as correlation (or dissimilarities such as the Euclidean distance) between the profiles, one could of course use the previously imputed full profiles of Table 4.2. But instead of it, correlation will be derived in the way mentioned earlier; namely by applying the correlation calculation that takes the missing values into account.

The reason why we are not using the average imputed values is that we would like to have the correlation calculations to be as unbiased as possible. In the following, the correlations are calculated by first constructing a complete data matrix by excluding the missing values from the matrix. It is done here in a simple way by removing the column 30 and removing the row

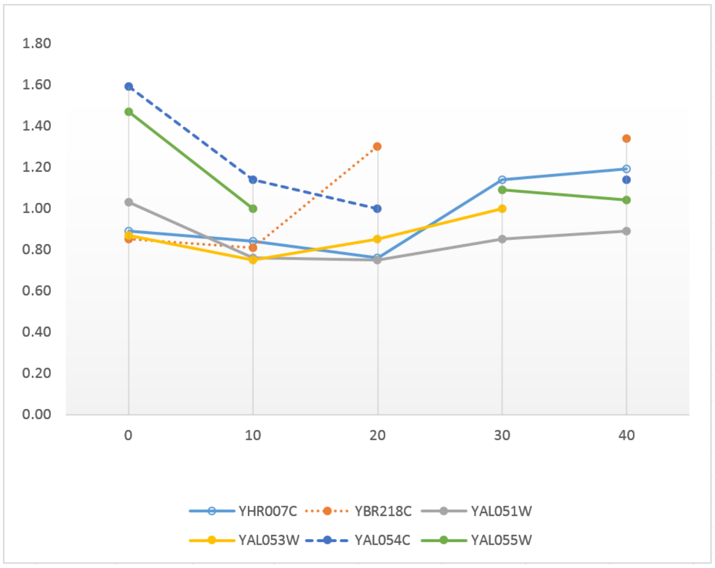

Figure 4.1: Gene expression profiles of the genes of the Table 4.1. The genes that contained a missing value at time point 30 are plotted with dashed lines.

	0	10	20	40
YHR007C	0.89	0.84	0.76	1.19
YBR218C	0.85	0.81	1.30	1.34
YAL051W	1.03	0.76	0.75	0.89
YAL054C	1.59	1.14	1.00	1.14

	YHR007C	YBR218C	YAL051W
YHR007C	1		
YBR218C	0.40	1	
YAL051W	0.39	-0.25	1
YAL054C	0.06	-0.61	0.91

Table 4.3: Complete sub-matrix of Table 1 (left) and correlation coefficients between its rows (right).

YAL055W. The resulting complete submatrix with correlation coefficients is presented in Table 4.3.

It is observed that the correlation between the genes YBR218C and YAL054C is low (-0.61). Thus, it might be a better idea to use a more sophisticated method than simple averaging to impute missing values of those genes. One way to incorporate the correlation in the imputation is to use the local average imputation technique. The local average imputation uses the correlation or some other similarity (or dissimilarity) measure to find the set of most similar vectors (i.e. the neighborhood) from the data from which the average imputation is calculated.

One common choice for dissimilarity measure is the Euclidean distance d_E (see Eq. 3.9). It can be generalized to work also with vectors that contain missing values: the squared difference is calculated only for the values a_{oj} and a_{ij}, where both values are present.

Perhaps the best known local average imputation method is the *k-Nearest Neighbor imputation* (k-NN). This method uses the k-NN classifier (see Section 3.2.2) for selection of k nearest neighbor genes for the gene a_o with missing values.

Let us now see how the k-NN imputation works. To impute a missing value for a_{op}, the algorithm first finds the neighborhood $N_k(a_o)$ of a_o. That is done by calculating the distances d_{Ei} between a_o and a_i where $i = 1, \ldots, o-1, o+1, \ldots, M$ and then sorting the distances d_{Ei}. After that the k first vectors with smallest values of d_{Ei} are taken and the average of their pth components is used as an estimate for missing value a_{op}.

Now, as we have seen how the similarity (or dissimilarity) between the vectors could be used as to guide the imputation, let us see how the missing values of Table 4.1 are imputed using the k-NN imputation. Instead of correlation the Euclidean distance is used here as the distance metric for simplicity.

Let us impute the missing value at column 30 of row YBR218C using the 2-NN imputation. When calculating the Euclidean distances between the row YBR218C and other rows, the two closest rows to YBR218C are the rows YAL053W and YHR007C and thus the missing value is estimated as (1.14 + 1.00) / 2 = 1.07.

Results of the 2-NN imputation are presented in Table 4.4, where all the missing values have different imputation values if compared to the ones got with the column-average imputation technique (Table 4.2).

We have now studied a rather simple way to improve the original average imputation using the average calculation of nearest k rows of the matrix $\mathbf{A}$. One can easily see the k-NN imputation as a generalization of the average imputation technique since using the value $k = M - 1$, the imputation technique reduces to the average imputation.

One drawback of the k-NN algorithm is the dependency on the selection of the parameter k. Troyanskaya et al. (2001) studied the effect of the neighborhood size on the imputation of microarray data; they applied the imputation on multiple data sets and showed that values of k between 10 and 20 provided the best imputation accuracies [81]. Tuikkala et al. also studied the selection of the value k on multiple microarray datasets and proposed 20 as the recommended value of k [P1].

4.2.3 Weighted average imputation

In the previous chapter the local average imputation was considered as a generalization of the average imputation algorithm. An apparent further generalization is to apply weights in the average calculation. It was demonstrated in the previous chapter that using only the most similar rows to impute the missing value is a sensible generalization of the simple average

	0	10	20	30	40
YHR007C	0.89	0.84	0.76	1.14	1.19
YBR218C	0.85	0.81	1.30	**1.07**	1.34
YAL051W	1.03	0.76	0.75	0.85	0.89
YAL053W	0.87	0.75	0.85	1.00	**1.04**
YAL054C	1.59	1.14	1.00	**0.97**	1.14
YAL055W	1.47	1.00	**0.88**	1.09	1.04

Table 4.4: 2-NN imputed values of the Table 4.1.

imputation. Thus, weighting for example by distances can be seen as a further generalization of the average imputation.

Similarly as k-NN imputation algorithm has its classifier counterpart (the k-NN classifier), same holds for weighted average imputation. The classification methods that use a weighting function in the distance calculation are called *kernel methods* [12].

The *weighted average imputation* is based on the *weighted average* calculation. While in standard average each sample contributes equally to the result, in the weighted average each sample has a unique weight that specifies the amount of its contribution to the result. More formally the weighted average for column j is shown in formula (4.3) where w_i is the weight of gene a_i.

$$\overline{a_{\cdot j}} = \frac{\sum_{i=1}^{M} w_i a_{ij}}{\sum_{i=1}^{M} w_i} \tag{4.3}$$

Standard average is a special case of the weighted average, where each $w_i \equiv 1$. For the local average calculation it holds:

$$w_i \equiv 1 \text{ if } a_i \in N_k(a_o) \text{ and } w_i \equiv 0 \text{ otherwise} \tag{4.4}$$

The weights w_i could be defined as in Section 4.2.2 to 0s and 1s, or more generally to be anything more complicated as long as the formula (4.3) is used in the calculation. Typically only normalized weights $w_i \in [0, 1]$ are used.

One obvious way to define the weights is to use the similarity of rows as the weights. For instance, the distances d_i between a_o and a_i can first be normalized by dividing each distance by the biggest row distance $\max_i(d_i)$ of $\mathbf{A}$, thus all the distances would be between 0 and 1. After that the similarities are obtained from the distances by subtracting these from 1. Another possibility is to use some external information about the relatedness between the a_o and a_i. For example, in case of gene expression data, if we know from the literature that genes YHR007C and YBR218C belong to the same functional group of genes, we could increase the weight of row (gene) YHR007C that is used in the weighted imputation of the missing value in the gene YBR218C.

In addition to average calculation, the weighting functions are possible to incorporate also into the neighborhood calculation. In this scheme the final distance d_{Ei} (see Section 4.2.2) between the gene a_o with missing value and its neighbor gene a_i depends also on the weighting function.

4.2.4 GO-based imputation

The GO-based imputation is an example of the weighted average imputation. In the publication P1 of this thesis the k-NN imputation was reorganized to use the *semantic similarity* information when determining the k-neighborhood of the gene to be imputed [P1]. The semantic similarity tells how much *mutual information* two genes mapped into the Gene Ontology share [52]. We will study the Gene Ontology and the semantic similarity more in the Section 5.3.1; here it is enough to understand that semantic similarity reflects the functional closeness of two genes. The idea of GO-based imputation is that the semantic similarity will help us to gather more functionally similar genes into the neighborhood than the simple distance-based neighborhood selection could do [P1].

In the publication (P1), a combined dissimilarity measure is formed by combining the semantic dissimilarity d_{Si}(see Section 5.3.1) with the Euclidean distance using a weight $\alpha \in [0, 1]$. The weight controls how much the semantic dissimilarity contributes to the combined distance value:

$$d_{Ci}(a_o, a_i) = d_{Si}(a_o, a_i)^{\alpha} \cdot d_{Ei}(a_o, a_i) \tag{4.5}$$

The parameter α is calculated adaptively as a first step of the imputation process using a small non-missing subset of the original data as a learning dataset [P1]. The semantic similarity information has been utilized also for clustering of gene microarray experiments [60].

4.2.5 Local least squares imputation

In the publications (P1) and (P2) of this thesis the local least squares imputation (LLS) method has been used as a baseline to evaluate the GO-based k-NN imputation. LLS imputation has originally been proposed by Kim et al. (2005). It consists of two steps. First, a procedure similar to the k-NN imputation is used to select the k-neighborhood for the gene with missing value. Then, the missing value is predicted by the least squares regression using the genes in the k-neighborhood to predict the missing value. The advantage of LLS over k-NN imputation is that it utilizes the correlation structure of the data [43].

In addition to the methods presented above, there are several other imputation algorithms which have been used for imputing gene expression datasets. In the following some of those are shortly introduced. Zero imputation replaces missing values with 0s. Iterative LLS imputation is a modification of the LLS imputation which iteratively improves the imputation by using the results of the previous imputation iteration as an input of the current iteration [16]. BPCA (Bayesian Principal Component Analysis) imputation involves Bayesian estimation of missing values with the iterative expectation maximization algorithm [58]. SVR imputation uses so-called radial basis kernel functions for neighborhood selection after which the missing value estimates are calculated with quadratic optimization [88].

4.3 Measuring the imputation accuracy

Since there are so many different imputation methods, a way to measure the accuracy of the missing value estimation technique is needed. A common strategy is to take a complete data matrix $\mathbf{A}_{M\times N}$ and place randomly a certain amount (e.g. 1 %) of missing values to it (see e.g. [16, 43, 81] and [P1]). These randomly generated missing values in $\mathbf{A}'_{M\times N}$ are then imputed and the imputed values are compared to the original values.

Perhaps the most frequently used method for comparisons has been the normalized root mean squared error (NRMSE) [81][P1]. This measure can be calculated when we have the original complete data with no missing values. The NRMS error between the original data matrix $\mathbf{A}_{M\times N}$ and imputed data $\mathbf{B} = (b_{ij})_{M\times N}$ is calculated with formula below.

$$NRMSE = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(a_{ij} - b_{ij})^2}{\sum_{i=1}^{M}\sum_{j=1}^{N} m_{ij} \cdot a_{ij}^2}}, \tag{4.6}$$

where the $\mathbf{M} = (m_{ij})_{M\times N}$ is the missing value indicator matrix of $\mathbf{A}'$. For example the missing value indicator matrix for the data of Table 4.1 is the following

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Another way to measure the imputation accuracy is to see how the imputation affects the further data analysis tasks applied to the data. Jörnsten et al. (2005) studied how imputation affected the significance analysis of differential expression [42]. They also proposed a new LinCmb imputation technique that adaptively uses estimates of some existing imputation methods to calculate the final imputation. They showed that missing values affect the significance analysis by increasing the false positive and the false negative rate of the results when compared to the results of the complete dataset [42]. They also proposed that their new imputation method is superior to widely used methods in terms of the NRMS error. In terms of false positive and negative rates, the algorithm is competitive to more advanced methods such as BPCA [58].

Scheel et al. (2005) studied the detection of differentially expressed genes from imputed data [70]. They compared imputation methods by investigating the percentage of lost differentially expressed genes compared to the set of differentially expressed genes of the complete dataset. They showed that their new way to compare imputation algorithms provides useful information that the NRMS error cannot find [70].

There are also studies on classification accuracy of imputed data [73, 87] which show that cross-validated classification accuracy of imputed datasets provides another alternative to the NRMS error. The first study of how missing values affect the stability of clusters of hierarchical clustering was by Brevern et al. [22].

In publication (P2) the imputation accuracy was evaluated by analyzing its effect on the preservation of original k-means clustering structure and on the Gene Ontology enrichment analysis of the clusters. It was shown that imputation is always a better choice than leaving the missing values in the data. Another observation of the study was that the BPCA method is one of the best imputation methods.

Chapter 5

Visualizing biological data

5.1 Introduction to visualization

Visualizing plain figures and sets of numbers is essential for making data more sensible for an observer. Edward Tufte writes in his book Beautiful Evidence [83] how visualization is an *evidence*. An evidence has a producer and a consumer; analytical thinking is used to produce a visualization from a set of figures and to communicate the evidence to the observer [83]. Good visualization is not only an evidence as in the saying “seeing is believing” but also as “seeing is understanding” [15]. Visualization should be also reliable; it is common knowledge that it is possible to mislead human perception with simple optical illusion such as the Grid illusion presented in Fig. 5.1, where the observer tends to see small colored squares in the crossings of white lines. The grid illusion was first published by Ludimar Hermann in 1870 [35].

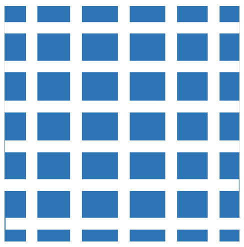

Figure 5.1: Grid Illusion by Ludimar Hermann (1870) [35].

The credits of the first statistical visualizations go to R. A. Fisher (1925) and his book of Statistical Methods for Research Workers [28]. The X-rays discovered in 1938 provided the first time in history a way to visualize non-visible things. In 1960 the *computer graphics* revolutionized the way how visualization can be created, presented and altered “in the fly” [15]. In 1977

John Tukey wrote his book Exploratory Data Analysis in which he provided a number of visualization tools for data analysts [85].

Visualization can be classified to different types of visual presentations. There are among others: charts, graphs, colored matrices, contours, vector fields, volume visualizations, multidimensional visualizations, geometric modelings, animations, sparklines and combinations of these [15]. In the present study the focus of interest is especially in graphs as they provide a natural way to visualize the relationships between genes and gene products [P4]. In this study, graph visualizations are used for biological data called *biological interaction networks*, in order to illustrate the interactions between biological components such as genes and gene products.

5.2 Visualizing biological data

In the following we define *biological data* as a set of signals measured from a biological sample and converted to digital values. Examples of biological data are the gene expression data discussed in Chapters 3 and 4, the DNA sequence information, and the information of relationships between genes and gene products. Different types of biological data have their own characteristics that might lead to different visualization needs. For instance, gene expression data has typically very high dimensionality which limits the usefulness of many visualization methods which have been designed to work in two or three dimensions. In this study we will present visualization methods for gene expression data and biological interaction networks.

5.2.1 Visualizing gene expression data

Chapters 3 and 4 included a discussion of some properties of the gene expression data analysis. In the following we limit our interest to the cDNA microarray gene expression data sets. As an example of the properties of typical cDNA microarray gene expression data set let us consider the data sets used in the publication (P2). Eight different microarray data sets were analyzed in that study. Dimensionality N of the data sets varied from 7 to 26 and the number M of genes was between 4771 and 7070. Each except one of the data sets contained also missing values: at the lowest 0.4% and at the highest 6.7% of all values were missing. The datasets were classified to three different types: time series, steady state, and mixed type (i.e. multiple time series) data sets. There were also notable differences in the correlation structures between the genes of each data set [P2].

The matrix or the data table representation that is seen for example in Table 4.1 might be the simplest form of gene expression data visualization. The problem of the data table is that it requires much space and is therefore practical only for small (sub) sets of the data. To ease the space requirements of the visualization, the size of the cells of the data table can be make smaller. That will result that the numbers will eventually become unreadable. One way to overcome the problem is to convert the numbers to colors.

Figure 5.2 visualizes about 4% of the whole diauxic shift data set with the open source TM4 Microarray Software Suite [68]. This type of visualization is called as a *color matrix* (it can be called also as *heat map*) visualization. The color matrix of Fig. 5.2 is transposed such that rows represent experimental conditions (time points) and columns represent genes. The gene names are left out from Fig. 5.2, but time points are visible. The color scale on the right hand side shows how the numerical expression ratios are converted to colors.

The colors of individual cells represent the expression ratio of genes such that bright red colors indicates overexpression of the gene in the studied sample when compared to the expression of the reference sample. Likewise the bright green colors indicates that the gene is underexpressed in the studied sample. The closer to black the color is the more similar is the gene expressed both in the studied and in the reference sample. Grey values are used for representing the missing values.

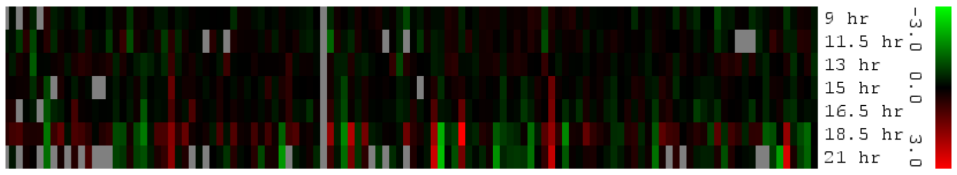

Figure 5.2: Color matrix visualization of part of the diauxic shift data set. Grey values are used for representing the missing values.

Figure 5.2 reveals at least two things. First, most of the genes are expressed almost similarly in the studied and the reference sample; this is seen from the abundance of black and dark colors. Secondly, it seems that in this small subset of data the most action is happening at the time point 18.5 h, since there seem to be more differently expressed genes than at any other time point.

In addition to these observations, Fig. 5.2 also shows several limitations of the color matrix visualization technique. First, the visualization still needs much space: even if the width of the cells is reduced to 1 pixel the whole data set would not fit into a normal computer screen. Secondly, the visualized data seems to be in disorder; the visualization might be more informative if the genes with similar expression profiles would be close to each other in the color matrix. Finally, there are a lot of missing values in the data which will distract the further data analysis.

To address these limitations we will first impute the missing values of the data with 20-NN imputation. Imputation provides us a complete data set that is much easier to analyze and visualize further on. Secondly, to increase the order of the data set and enhance its visualization we group the genes with similar expression profile near to each other. This can be done with the grouping algorithm presented below. This algorithm is developed from the hierarchical grouping algorithm presented by Joe H. Ward (1963) [89]. The result of that kind of grouping is presented in Fig. 5.3.

Hierarchical grouping algorithm

1. Suppose given data matrix $\mathbf{A}_{M \times N}$ and distance measure $d(a_j, a_k)$.

2. Let $\mathbf{B}_{M \times N}$ be a copy of $\mathbf{A}_{M \times N}$, and let $X = M$

3. Let $F = \{S_1, S_2, \ldots, S_M\}$ be a data structure of M sequences such that initially each $S_i = (i)$. Let S_{ij} denote jth element of S_i.

4. Find the two genes (rows) a_i and a_l from $\mathbf{B}_{X \times N}$ such that their expression profiles are the most similar to each other: $d(a_i, a_l) = min_{j,k=1}^{X} d(a_j, a_k)$, where $j < k$.

5. Combine sequence S_l with S_i and remove S_l from F such that indexes of the sequences following S_l are updated as following $F' = \{S_1, \ldots, S'_i, S_{i+1} \ldots, S_{l-1}, S'_l, \ldots, S'_{M-1}\}$, where $S'_i = S_i \cup S_l$ and $S'_l = S_{l+1}, \ldots, S'_{M-1} = S_M$.

6. Calculate an average pseudo gene a_k for the genes indexed by the combined sequence S'_i, if e.g. $S'_i = (i, l)$, then $a_k = ((a_{i1}+a_{l1})/2, \ldots, (a_{iN}+a_{lN})/2)$.

7. Replace genes a_i and a_l of $\mathbf{B}_{X \times N}$ with a_k thus producing $\mathbf{B}_{X-1 \times N}$, let $X = X - 1$

8. Continue from step 4 until there is only one pseudogene left in $\mathbf{B}_{X \times N}$, i.e. $X = 1$, and thus only one combined sequence S'_1 left in the F'.

9. Finally, rearrange rows of $\mathbf{A}_{M \times N}$ into $\mathbf{A}'_{M \times N}$ such that ith row of $\mathbf{A}'_{M \times N}$ is the S_{1i}th row of the $\mathbf{A}_{M \times N}$. In the end, the genes (rows) of $\mathbf{A}'{}_{M \times N}$ are grouped together such that genes with similar expression profile are located near to each other.

The algorithm above is very similar to *agglomerative hierarchical clustering*, which will be discussed in the following section. The algorithm can also by used as a data reduction tool when the genes which are close enough to each other are combined to a single average gene and thus the visualization can be packed to a more compact representation.

5.2.2 Visualization and clustering

Clustering (see Section 3.3.2) can also be used as a data visualization method. The goal of microarray data clustering is to divide rows (genes) of the data set into several clusters, such that each cluster contains only the genes that are similar to each other and also all *clusters* (as represented by a centroid gene) are dissimilar to each other.

Clustering is typically one of the first tools that is applied to preprocessed gene expression data [24]. It allows also a more advanced visualization of the data than a simple color matrix visualization.

Hierarchical clustering of diauxic shift data set is presented in Fig. 5.3. Hierarchical clustering can be done by the grouping algorithm presented in the previous section; its result is further visualized by drawing a *dendrogram* that reveals the sub-clustering structure of the data with a *binary tree*. The dendrogram is drawn such that whenever two genes (or pseudo genes) a_i and a_l are clustered together then these two genes are connected with an arc shaped edge to represent the pseudo-gene a_k as a parent node of the genes in the binary tree. The height of the arc represents the distance between the (pseudo) genes that are grouped together. In the clustering of Fig. 5.3 we have used the correlation as a distance measure between the genes. The distance scale is shown on the top right corner of Fig. 5.3. The distance scale shows the distance of two genes a_i and a_l at the moment when they were clustered together.

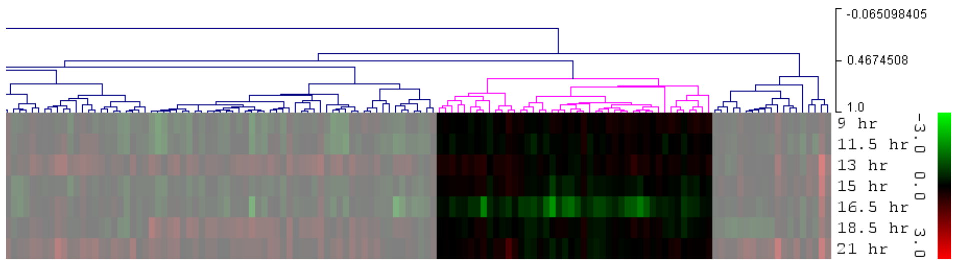

Figure 5.3: Hierarchical clustering of a part of the *diauxic* shift data with one highlighted sub-cluster.

Besides the hierarchical clustering there are many other clustering algorithms such as k-means and *SOM* clustering [79]. The k-means clustering (see Section 3.3.2) and hierarchical clustering are some of the most popular clustering algorithms applied to gene expression data [24]. For a more detailed description of different clustering algorithms see Andreopoulos et al. (2009) [5].

5.3 Biological interaction networks and visualization

Next step after visualization and exploratory data analysis such as clustering, is to deduct biologically relevant conclusions from the data. A typical goal is to derive biological interactions from the experiments. These interactions happen between the molecules like proteins and nucleic acids and can form wide networks of interactions.

These networks are commonly graphs containing a set of *nodes* and *edges*. The nodes are typically either genes or gene products such as proteins and the edges represent the relatedness between the nodes. It is typical that an interaction network is relevant only for a single organism like yeast, or human, but there are also examples of more general graph-like structures that can be applied to multiple different organisms.

Researchers are interested in many different kinds of biological interaction networks; the type of a network typically depends on the type of the molecules represented by the nodes.

In the protein-protein interaction (PPI) networks the nodes of the networks are proteins and the interactions between them are the (typically undirected) edges of the network. An edge between the pair of nodes indicates that the corresponding proteins physically bind [1]. In the gene-regulatory (GR) network edges are typically directed and thus have the *source* and *target* node. The source node depicts the transcription factor such as a protein that regulates the gene (the target node). Edges can involve weights which are used to model the degree of the regulation. In the genetic interaction (GI) network, nodes represent genes and the a edge between the nodes (genes) is an indication that the gene affects to the function of the other gene. The effect of a gene to the other gene's function can be varied: the first gene can e.g. reduce the expression of the other genes.

In this study we will treat the biological networks as the repository of biological knowledge on the relationships between the molecules. The biological interaction networks are constructed by gathering information of relationships between the selected nodes of the network. The relationship information can be e.g. from a particular microarray experiment, or it could have been derived from multiple experiments, or it could be even predicted from known biological models [3].

5.3.1 Gene Ontology

The Gene Ontology (GO) can be thought as an example of a network of biological knowledge shared by several different organisms. The definition for an ontology in general, is a set (vocabulary) of objects (beings, conceptual elements), their properties (nature), and their relations to other objects [17].

relation	an example
is a	mitochondrial DNA repair **is a** mitochondrial DNA metabolic process
occurs in	mitochondrial DNA metabolic process **occurs in** mitochondrion
part of	mitochondrion (is) **part of** cell
regulates	regulation of meiotic cell cycle **regulates** meiotic cell cycle

Table 5.1: Subset of the term relations in the Gene Ontology

The GO could be seen as a graph-like structured biological network but it is more than that. The GO is a dynamic and controlled vocabulary that can be applied to describe roles of biological processes, molecular functions and cellular components of several organisms [6]. The GO is maintained and actively developed by the Gene Ontology Consortium [6] which is a set of communities for model organisms, protein databases, and research.

From a practical point of view, the structure of the GO is a set of Gene Ontology *terms* which are divided into three separate domains: cellular components, molecular function, and biological process. A GO term can be in relation to a set of other GO terms. Each term has its id, name, namespace, definition, etc. Examples of types of relations used in the GO are presented in Table 5.1.

One can see from Table 5.1 that some of the relations are related to certain parts of the cell and others are linked to biological processes of the cell. The GO graph contains three subgraphs: one for biological processes (BP), one for molecular functions (MF), and one for cellular components (CC). On the top of the GO graph these three subgraphs are linked to one root term named gene ontology. A small part of the BP Gene Ontology is visualized in Fig. 5.4. This visualization is produced with QuickGO Gene Ontology browser [14]. The visualized graph shows the acyclic directed rooted graph structure of the Gene Ontology.

The GO Consortium also maintains databases of annotated genes and gene products for several organisms. Each annotated gene or gene product typically has mappings to several GO terms. For instance, yeast's (*Saccharomyces cerevisiae*) protein YKL192C (Mitochondrial matrix acyl carrier protein) is mapped i.a. to the GO term GO:0006633 (fatty acid biosynthetic process).

The GO is typically used as a tool for biological evaluation of the results of downstream data analysis methods such as clustering [P2], finding of differentially expressed genes [77, 78, 80], or network visualization [P4].

Semantic similarity in the GO

As stated before, the semantic similarity can be interpreted as a measure of how much knowledge content is shared by two entities [52]. In Section 5.3.1,

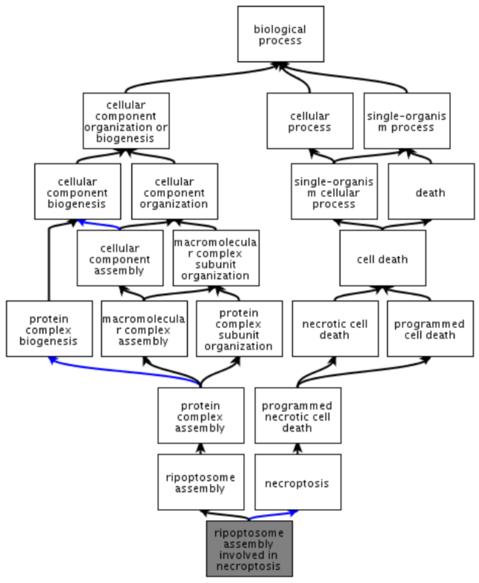
biological process
cellular component organization or biogenesis
cellular process
single-organism process
cellular component biogenesis
cellular component organization
single-organism cellular process
death
cellular component assembly
macromolecular complex subunit organization
cell death
protein complex biogenesis
macromolecular complex assembly
protein complex subunit organization
necrotic cell death
programmed cell death
protein complex assembly
programmed necrotic cell death
ripoptosome assembly
necroptosis
ripoptosome assembly involved in necroptosis

Figure 5.4: Visualization of a part of the biological process ontology starting from the GO term GO:1901026 "ripoptosome assembly involved in necroptosis".

we described the Gene Ontology as a tree-like graph structure of connected GO terms. The semantic similarity of two GO terms c_i and c_j can be defined as the *mutual information content* of these two terms. Based on these observations, Lord et al. [52] proposed a straightforward way to determine the semantic similarity of GO terms c_i and c_j: it is measured as the information content $p(c)$ of the *minimum subsumer* c of terms c_i and c_j. For instance, in Fig. 5.4 the semantic similarity of the terms "protein complex biogenesis" and "protein complex subunit organization" is the information content of the term "cellular component organization or biogenesis". The information content $p(c)$ of term c can be estimated as the probability of the occurrence of the term and its child terms [52]. More formally:

$$p(c) = \frac{|c|}{\sum_{i=1}^{N} |c_i|}, \tag{5.1}$$

where $|c|$ is the number of occurrences of c (i.e. 1 + the number of child terms of c), N is the total number of GO terms in the ontology, and $|c_i|$ is the number of occurrences of the term c_i. As a standard distance (dissimilarity) measure, the Euclidean distance d_E was used in the previous chapter. Another dissimilarity measure, called semantic dissimilarity, is needed here. Semantic dissimilarity measure $d_S(a_i, a_j)$ [52][P1] is calculated with the algorithm that first associates the genes a_i and a_j to a set of GO terms T_i and

T_j. GO term c is included in the set T_i (T_j) if there is a GO annotation that associates a_i (a_j) to c. Then, the information content $p_{ij}(c)$ is calculated between each $c_i \in T_i$ and $c_j \in T_j$ as above. Finally, the semantic dissimilarity is calculated as an average of the information content values $p_{ij}(c)$ [P1].

5.3.2 Network visualization

The network visualization is a process where nodes and edges of the network are placed in a two- or three-dimensional space. The visualization process is formalized as a *layout algorithm*. The (automatic) network visualization dates back to 1960s when D. Knuth published the article on drawing flowcharts [44]. After that, several algorithms have been proposed on drawing different kind of graphs and networks. When drawing the visualization of a network, several *aesthetic criteria* are taken into account to maintain the readability of the visual outcome [10]. These aesthetic criteria include minimization of the number of crossings between the edges, minimization of the area of the visualization, and maximization of the smallest angle between the edges from the same node [10]. These criteria form hard optimization problems and may conflict with each other such that optimizing one criterion weakens another. Beside the selected aesthetic criteria, the network visualization is associated with *drawing conventions* which tell how the nodes and edges should be drawn. In the following, the nodes are drawn as small circles and edges are straight lines between the nodes.

One of the simplest layout algorithms is the *grid layout*: it first draws the nodes of the network in a random order into a given rectangular area (like computer screen) starting from the top left corner and continuing to the right as long as there is space for the next node. When the first row of the area is filled with evenly spaced nodes the algorithm continues from the next row (i.e. below the node that was placed to the top left corner). The spacing between the nodes is calculated such that the given area is fully utilized. After all nodes have been placed, the edges are drawn between the nodes as straight lines.

An example of the grid layout visualization of protein-protein and protein-DNA interactions from a study of yeast galactose utilization pathway [41] is presented in Fig. 5.5. The layout has been produced with Cytoscape [72] network visualization software. The grid layout algorithm is a fast and simple way to produce a network visualization but it has many disadvantages. First, the edges between the nodes placed on the same row or column are not distinguishable from other edges of the same row or column. Then, the possible substructures (like clusters) of the graph are not easy to find from the visualization. Finally, the layout produced with the algorithm is often far from aesthetic due to the typically very high number of edge intersections.

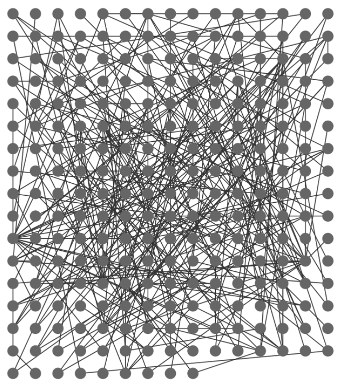

Figure 5.5: Grid layout visualization of a subset of protein-protein and protein-DNA interactions of yeast.

There are different approaches to producing readable network visualizations. One of the most used approaches for networks with straight line edges is the *force-directed approach* (or *spring embedding*) which simulates a system of spring forces between the nodes of the graph [10]. This idea was originally proposed by Eades [26]. In this approach, a natural spring length l_{uv} is assigned for each pair of nodes (u, v) of the graph. The value l_{uv} can be chosen as the number of edges on the shortest path between the nodes u and v [10].

The spring forces help to keep the lengths of the edges at optimal level. The system needs also so called global repulsive forces, which push the nodes further away from each other, to prevent the spring system from collapsing to a locally optimal solution [86]. Typically, these global repulsive forces are calculated within a predefined distance R from each node to its neighbor nodes to speed up the calculation [86]. An iterative optimization technique is used to find the configuration of nodes such that the energy of the spring forces is minimized. An example of the force-directed visualization of the network of Fig. 5.5 is presented in Fig. 5.6.

The multilevel layout (MLL) algorithm used in the publication (P4) is a modification of the multilevel force-directed placement algorithm proposed by Walshaw [86]. The force-directed node placement algorithm used in Walshaw's method is a modification of the Fruchterman & Reingold (FR) algorithm [30] that is based on the original idea of Eades.

The multilevel optimization of the Walshaw's algorithm is the first practical implementation of the idea where the graph is iteratively coarsened (or clustered) by combining the closest nodes into a cluster (*metanode*) [86]. This coarsening is continued until the graph is reduced to a smaller size than a given threshold. This set of gradually coarsened graphs are then laid out by a force-directed (or other) algorithm starting from the smallest graph.

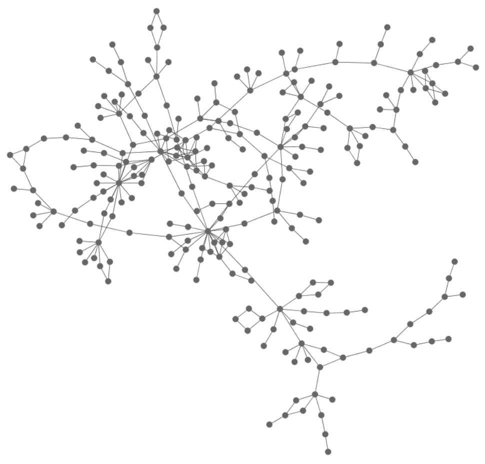
Figure 5.6: Force-directed layout visualization of the same data as in Fig. 5.5.

The layout calculated for the smallest graph is used as an initial layout of the parent graph. This process is iterated until the original (un-coarsened) graph is laid out.

The closest nodes in the coarsening phase are found by calculating a *matching* which means that a *maximal independent subset* of the graph edges is found. Each pair of two nodes connected by an edge in the subset is combined to a metanode and the edge and the two nodes are removed from the graph. In the original graph an unit weight is assigned for each node; the weight of a metanode is calculated as a sum of the weights of the combined nodes.

As the finding of the maximal independent set is a computationally costly operation, Walshaw used a non-optimal method where the randomly ordered list of nodes is searched for nodes that are not yet matched (i.e. the node nor its neighbor node are not in the subset) [86]. The order in which the nodes are selected into the subset is based on the increasing weight of the nodes [86].

For further information on graph visualization we suggest the reader to refer the textbook “Algorithms for the Visualization of Graphs” by G. Battista [10].

Chapter 6

Summary of publications

This thesis is built around publications (P1) to (P5) presented below. Since the research articles deal with multiple topics of interest; the main topics that are dealt with in them are summarized in Table 6.1.

	(P1)	(P2)	(P3)	(P4)	(P5)
Microarray data analysis	×	×			×
Missing value imputation	×	×			
Algorithm development	×		×	×	
Clustering or classification		×	×	×	×
Biological networks	×			×	
Visualization				×	

Table 6.1: Summary of topics studied with in the publications of the thesis. Columns (P1) to (P5) corresponds the publications and the rows to the most important topics of the publications

6.1 Improving Missing Value Estimation in Microarray Data with Gene Ontology (P1)

In publication (P1), we introduced an idea of improving the missing value imputation with biological a priori information of gene relationships. It was shown how the *semantic similarity* information of genes mapped into the Gene Ontology can be used to improve the accuracy of missing value imputation. A Gene Ontology based k-NN imputation technique was introduced. The power of the technique was evaluated using four publicly available yeast microarray datasets. The GO-based k-NN imputation was compared against the standard k-NN imputation and more advanced local least squares (LLS) imputation [43].

The experimental results with four datasets suggest that the semantic similarity information obtained from the GO enhances the imputation accuracy of the k-NN algorithm (compared to the original k-NN imputation) especially when there are lots of missing values and the number of experimental conditions (N) of the dataset is small. The GO-based local least squares (LLS) imputation did not provide such a clear improvement over the original LLS method.

In the tests, the datasets were first made complete by removing genes containing missing values from each of them. Then, a specific amount (1%, 5%, 10%, and 20 %) of missing values were randomly added to each dataset and the NRMS error was calculated between the original and imputed values. This testing procedure was repeated ten times for each dataset, for each missing value percentage and for each imputation method.

Figure 2 of (P1) shows a comparison of the NRMS errors of the imputation methods for different missing value percentages in the *diauxic* and *histone* datasets. Diauxic is time-series data that contains the expression values of 6068 genes in seven experimental conditions [23]. The histone data is from a study of the effect of nucleosomes and silencing factors on the yeast's global gene expression: it is time-series and contains 6181 genes in 7 experimental conditions [91].

Figure 2 of the publication shows how the increased number of missing values increases the benefit of the GO-based k-NN imputation over the normal k-NN. It was also shown that although the proposed GO-based imputation was able to improve the k-NN imputation, the LLS imputation produces still more accurate imputation results. In addition, the GO-based approach was not so successful when it was applied to the LLS imputation. This phenomenon is partly explained by the fact that the k-NN imputation typically uses a smaller neighborhood than the LLS imputation (20 genes vs 150 genes) and thus k-NN imputation benefits more from the wisely selected genes. Moreover, the GO-based imputation technique was developed to the neighborhood selection rather than for the estimate calculation, thus its benefit decreases as the size of the neighborhood increases. Finally, the results of LLS imputation were already so good that there was little room for fine tuning that would have been available from the GO-based approach.

The study left room for future research. First, only a few yeast microarray datasets were studied. It would be interesting to see how GO-based imputation would perform for other organisms such as human. Secondly, more study is needed to incorporate the semantic dissimilarity calculation not only into the neighborhood selection but also into the estimation step of the k-NN and into other imputation methods. Further, there are many other ways to calculate the semantic similarity in the GO than the method of Lord et al. used in (P1). Finally, the semantic similarity in the GO is not the only external information that could be applied to imputation; for example the similarity of gene's protein sequences could be used too [65].

6.2 Missing Value Imputation Improves Clustering and Interpretation of Gene Expression Microarray Data (P2)

The effect of missing values and imputation on the clustering of the gene expression data is discussed in publication (P2). As clustering is one of the first exploratory tools used on a new microarray dataset, it is important to know which imputation technology should be applied to the dataset such that the clustering results would be good.

In addition to the clustering, publication (P2) compares the imputation against the Gene Ontology enrichment analysis of the clusters and evaluates how different properties of the data, such as correlation structure and missing value distribution, affect on the imputation results. The publication consists of an analysis of eight public gene microarray datasets with seven imputation algorithms and one non-imputing approach. Used algorithm are zero imputation (ZERO), row average (RAVG), k-NN, Bayesian principal component analysis (BPCA), local least squares (LLS), iterated LLS (iLLS), and support vector regression (SVR) imputation.

The imputation accuracy was evaluated by analyzing its effect on the preservation of the original k-means clustering structure and on the Gene Ontology enrichment analysis of the clusters.

The study was initiated by preprocessing the gene expression datasets by filtering out the genes with low variability and standardizing the datasets. Then, genes with missing values were removed from each dataset to produce eight complete datasets. Missing values were generated into each test dataset using a similar probability distribution of the missing values as there was in the original dataset.

Then, the k-means clustering was applied with various amounts of clusters ($k = 2, 3, \ldots, 10$) to the complete datasets to produce a set of 8 reference clusterings for each dataset. Next, each missing value dataset produced earlier was imputed with all the imputation algorithms. The datasets were then clustered and the produced clusterings were compared to the corresponding reference clusterings.

Figure 6.1 illustrates the reference clustering of the complete data (Fig. 6.1 A) and a clustering (Fig. 6.1 B) obtained when the corresponding dataset with missing values was imputed and clustered. The moved three genes demonstrate how the imputation affects the resulting clusterings.

The comparison of the clusterings in publication (P2) was done by comparing the clustering of the complete data to the corresponding clustering of

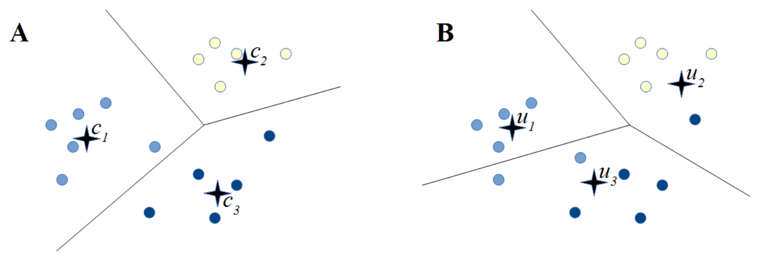

Figure 6.1: (A) An example of the reference clustering of a complete data, (B) corresponding clustering obtained after missing value imputation. Circles represent clustered genes and stars are the cluster centroids.

the imputed data using the average distance of partitioning (ADBP) as the measure of the agreement. The idea of the ADBP measure is to sum up the differences $D(c_j, u_j)$ of the corresponding cluster pairs of the two clusterings and normalize the sum of differences such that the ADBP error 0 indicates that the clusterings are identical, and 1 means that the clusterings are totally different. The difference of a cluster pair is calculated by calculating a set-theoretic difference of u_j of c_j and vice versa, the differences of all cluster pairs are summed up and divided with summed cluster sizes. For example, the ADBP error between the clusterings A and B in Fig. 6.1. is:

$$\begin{aligned} ADBP(A,B) &= \frac{1}{k}\sum_{j=1}^{k} D(c_j,u_j) \\ &= \frac{1}{3}(D(c_1,u_1)+D(c_2,u_2)+D(c_3,u_3)) \\ &= \frac{1}{3}(\frac{1}{6+4}(2+0)+\frac{1}{5+6}(0+1)+\frac{1}{5+6}(1+2)) \approx 0.19 \end{aligned}$$

By definition, ZERO imputation always has the NRMS error one. RAVG was the second worst as expected and it outperformed the ZERO imputation in all the studied cases. The next worst imputation method in terms of NRMSE was k-NN. It operated properly only when there was a strong correlation structure in the dataset. There was no clear winner among the more advanced imputation methods since the imputation accuracy of each method depended on the properties of the dataset being imputed. On the other hand, when counting the ranks of the methods, iLLS and BPCA were among the two best.

When comparing the ADBP error of the clusterings one can clearly see how seriously missing values can affect the ability of the clustering algorithm to reproduce the reference clustering. Even as small amount as 0.5% had a distinct impact on the clustering result. Another observation was that the imputation was always a better choice than leaving the missing values in the data (the Nimp approach). Perhaps the most striking observation was that there was no clear winner among the more advanced imputation methods. As expected the ZERO and RAVG methods were clearly the worst methods, but k-NN was almost as good as other more advanced methods in all but two datasets (Spellman98 and Hirao03) which had a relatively low correlation structure.

In general, the ADBP error of clusterings of GO terms confirmed the results obtained from gene clustering comparison. The biological interpretation of the clustering results could be preserved if missing value ratio was low enough and a sophisticated imputation method was used.

As a conclusion, imputation is a good choice when there is intention to make a cluster analysis of gene microarray data having missing values. According to the present results, the BPCA is one of the best imputation methods if a fast and reliable imputation method is needed. A benefit of the method is that its performance is independent on the properties of the dataset.

When the missing value rate goes above 5%, none of the methods could reasonably correct the influence of missing values on the consistency of clustering results. One possible approach could then be to filter out a certain amount of genes which contain the highest amount of the missing values.

6.3 A multilevel layout algorithm for visualizing physical and genetic interaction networks, with emphasis on their modular organization (P3)

The problem of visualization of very large biological interaction networks was studied in publication (P3). One of the most popular software suites especially developed for visualization of biological networks is Cytoscape [72]. A multilevel layout (MLL) plug-in for Cytoscape was implemented in the present work. The multilevel optimization method, originally proposed by Walshaw [86] was modified for capturing the modular organization of biological networks. The current implementation works with very large networks and provides reasonable visualizations of complicated networks. In addition, a network validation tool was developed. The tool uses external biological information to assess the quality of a given layout.

The MLL algorithm modifies the Walshaw algorithm by adding an option to use node degree weighting in the matching procedure. This option will help in finding a bigger subset of edges (and thus better coarsening) when there are 'star-like' structures present in the graph. Further, the M-tree structure [19] was used to quickly find the neighbor nodes of the current node in the calculation of the global repulsive forces. Finally, another option was added to the algorithm to help emphasizing dense cluster structures of biological networks [P4]. Figure 6.2 demonstrates the visualization produced with the MLL algorithm; the node degree weighted matching and the clustering options were used here.

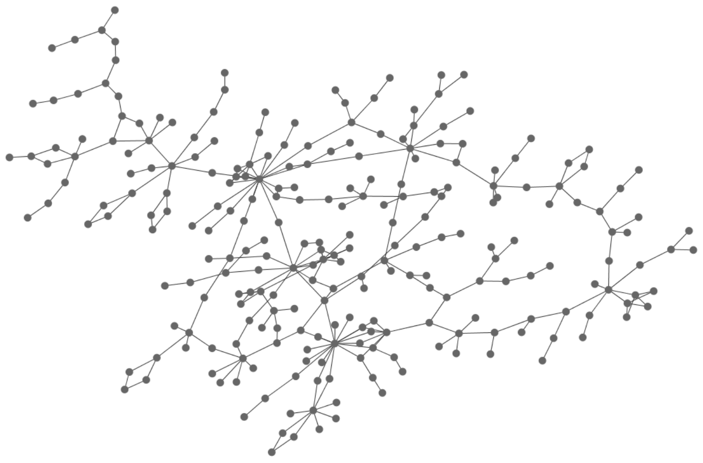

Figure 6.2: Multilevel layout (MLL) visualization of the same data as in Fig. 5.5.

Evaluation of the MLL algorithm showed that it produced biologically relevant and visually pleasant layouts in most biological network types. The computational complexity of the algorithm was reduced by using the M-tree data structure to index the node distance calculations in the step where the repulsive forces are calculated to the nodes that are close the node that is currently processed. As a conclusion the implementation of the MLL algorithm offers a competing option for other layout solutions available in the Cytoscape.

6.4 Accelerating GLA with an M-Tree (P4)

In publication (P4), algorithmic techniques to reduce the computational time of the generalized Lloyd algorithm (GLA, called also the k-means). The algorithm is commonly used as a codebook generation algorithm in vector quantization [45], in data mining [79], as well as in k-means clustering of gene expression profiles [24]. The idea was to utilize the M-tree data structure [19] to reduce the number of distance calculations needed in the GLA. The algorithm is called k-means clustering in the following, for sake of consistency.

Basically, the speed of the k-means clustering mostly depends on the number of distance calculations since it is the most costly operation of the algorithm. In the original version of the algorithm each partitioning step requires $\mathcal{O}(k \cdot M)$ distance calculation, where k is the number clusters and M is the number of data objects.

The idea in (P4) is to build an M-tree over the set of k centroids at the beginning of each partition step. The M-tree structure makes it possible to find the nearest centroid for a data object using a logarithmic number of distance calculations. Since an M-tree must be rebuilt at the beginning of the each partition step, it causes an overhead of approximately of $k \cdot log(k)$ distance calculations. On the other hand, the number of distance calculations reduces to $\mathcal{O}(log(k)) \cdot M$ for each partition step.

The computational results suggest that the M-tree optimized k-means clustering clearly outperforms the standard k-means in terms of the number of distance calculations needed. The new approach was also compared to the so called triangle inequality elimination (TIE) method in which the distance calculations between data object x and cluster centroid c_i are avoided if the distance between c_i and the nearest centroid c_a found thus far (in k-NN search) is greater than 4 times the distance between x and c_a [18][P4]:

$$d(c_i, c_a) > 4 \cdot d(x, c_a). \tag{6.1}$$

This means that a significant part of the distance calculations could be omitted by computationally low cost multiplication operation.

6.5 Altered expression of p120catenin predicts poor outcome in invasive breast cancer (P5)

This publication provides an example how to apply microarray data analysis techniques on clinical data. The contribution in this work was mainly the applying of statistical and algorithmic data analysis techniques on the given raw cDNA microarray data. Other related publications are summarized after Chapter 7.

Publication (P5) focuses on the role of regulator proteins on the prognosis of the invasive breast cancer. First, cDNA microarrays were used to find genes that were differentially expressed on the cancer in general and also between the metastatic and the non-metastatic (local) groups. The microarray data analysis was performed on tumor samples obtained from 10 patients. The same patients were used to obtain reference samples from the healthy breast tissue outside of the tumor. The study includes also an immunohistochemical analysis of 341 tissue arrays.

The RNA that was extracted from the tumor samples and a pooled reference sample was hybridized on the cDNA microarrays. The final gene expression ratios were $\log_2$-transformed and intensity-normalized with the LOWESS method. Data were analyzed in two phases. First, differentially expressed genes were found between the tumor and the health tissue samples. Secondly, we identified gene pairs that could be used to achieve as clear classification as possible between the metastatic and local breast cancer groups.

The differentially expressed genes were found similarly as in the related publications [77, 78]. Shortly, the p-value < 0.05 of t-test against zero and average absolute gene expression value above 0.5 were used as threshold values.

Separation in gene expression between local and metastasized cancer cases was performed with the 1-Nearest Neighbor method, searching for pairs of genes which were able to separate the groups of the local and the metastasized cases without any errors. The gene pairs were required to produce linear separation between the classed. The classification results were validated with the *leave-one-out* method and a classification score was calculated for the gene pairs using Fisher's discrimination ratio.

As a result, p120catenin was found to be down-regulated with genes producing E-cadherin and α-cadhering. Further immunohistochemistry results verified that p120catenin is an independent predictor of breast cancer survival.

Chapter 7

Conclusions

The massive amount of quantitative data (such as gene expression data) available has initiated a change in the field of biology. The biological research has become more and more data-centered. This change has affected other sciences such as computer science, statistics, and mathematics. The challenges in biological data acquisition, organization, analysis, and visualization have implied development of even more sophisticated computational methods. The development of algorithms for these needs has been the main topic of the thesis.

7.1 Achievements of the Thesis

In this thesis, we have addressed several practical problems in terms of developing and analyzing algorithmic techniques to be used in the microarray data processing tasks. The data analysis begins by preprocessing, such as missing value imputation. It then continues by downstream data analysis tasks such as clustering and classification and by visualization of the results.

It was shown how the missing value imputation of cDNA microarray data can be improved using the Gene Ontology as a source of a priori information on the semantic similarity of genes (P1). We then introduced a novel way to measure the imputation accuracy based on the clustering of imputed data (P2).

To help the visualization of very large biological interaction networks we developed a multilevel layout plug-in (P3) for Cytoscape software suite [72]. Alongside with that, we proposed an algorithm (and accompanying Cytoscape plug-in) to reliably assess the biological relevance of any biological interaction network layout.

The usability of many data analysis techniques depends on the computational complexity of the algorithms behind the methods. We gave a way to improve the computation efficiency of k-means clustering with an M-tree index [19] built on the centroids of the k-means clustering (P4). This indexing structure could be incorporated also in other algorithms such as k-NN imputation and in classification methods that rely on neighbor searches.

Finally, we showed how analysis of the gene expression data of breast cancer study can be analyzed with basic statistical and algorithmic techniques (P5).

7.2 Final remarks

Only a small portion of the whole field of biological data analysis is discussed in detail in the thesis. The field is an interesting research area since there is so much room for further improvements: the algorithms can be optimized for the computational, statistical and biological performance, and for more aesthetic visual outcomes. On the other hand, the nature of the biological data itself and the importance of the possible results brings an extra motivation to the development of even better methods for this domain.

The missing value imputation (topic of the older publications (P1) and (P2)) has been studied also after the publications of the thesis. For instance, Oh et al. studied the biological impact of missing values on three clustering, three differential expression detection (DE), and four classification methods [59]. They found out that the DE methods were the most sensitive to the missing values. Otherwise their results support the ones of our study (P2), where it was hard to find a single best imputation method for all cases [59], but instead more advanced methods such as BPCA and LLS seemed frequently produce good imputation results [59].

Also the topic of the oldest publication (P4) of the thesis, the optimization of k-means clustering, is still under ongoing research. For example Fausett et al. (2013), accelerated the nearest neighbor search method of the k-means algorithm by utilizing geometric relations among input vectors [27]. They proposed that the new method was superior to other tested methods such as TIE for the used test data.

As future work, missing data imputation methods could be applied also to other research topics of bioinformatics where missing, uncertain, or unreliable information is present. One example of this is the *genome-wide association study* (GWAS) [67], where imputation has been used for estimating uncertain genotypes [39, 54].

List of related co-authored publications not included in the thesis.

The author has taken part to bio-medical research projects. Below, a short description of their most important results alongside with explanation of the used statistical and algorithmic techniques is provided below. The most relevant contribution of the candidate to these publications was statistical and algorithmic data analysis.

1. *Maaria Tringham, Johanna Kurko, Laura Tanner, Johannes Tuikkala, Olli S. Nevalainen, Harri Niinikoski, Kirsti Näntö-Salonen, Marja Hietala, Olli Simell, Juha Mykkänen, Exploring the Transcriptomic Variation Caused by the Finnish Founder Mutation of Lysinuric Intolerance (LPI). Molecular genetics and metabolism 105, 408-415, 2012.*

   In this publication we studied genome-wide gene expression profiling of patients having autosomal recessive disorder called lysinuric protein intolerance (LPI). LPI patients have various symptoms such as failure to thrive, protein aversion, and anemia. The study subjects here were 13 Finnish LPI patients and for a control group we had 10 healthy volunteers. The sex and the age of each control volunteer was chosen to match the ones of the studied patient.

   The RNA of each subject was extracted from the blood sample and hybridized on an Illumina Expression BeadChip Array. Gene expression values were normalized with quantile normalization and signal log ratios (SLR) were computed between the expression values of the studied and control samples. Genes where SLR was below 0.75 or where the control expression value was below 300 (the average background value being 220) were filtered out. The resulting 9920 genes were further filtered such that the fold change had to be at least 0.8 and p-value of t-test at most 0.05. As a result of this analysis we discovered 487 (439) significantly up-regulated (down-regulated) genes [80].

The functional distribution of these up- and down-regulated genes was studied by plotting the gene frequencies against their Gene Ontology annotations (GOA). Moreover, "the ratio of the GOA frequency of the genes with altered expression to those of the genes with unaltered expression was plotted for each GOA category in order to pursue the change of gene functions observed in LPI". Based on these analysis steps we were able to divide the up- and down-regulated genes into different Gene Ontology categories that were distinct from the categories of the control samples. For instance, inflammatory response, immune system processes and apoptosis had signification accumulation of the up- and down-regulated genes.

2. *Kati Talvinen, Johannes Tuikkala, Olli Nevalainen, A Rantanen, Pirkko Hirsimäki, Jari Sundström, Pauliina Kronqvist, Proliferation Marker Securin Identifies Favourable Outcome in Invasive Ductal Breast Cancer. British Journal of Cancer 99, 335-340, 2008.*

   This publication studied the gene expression and the immunohistochemical analysis of the invasive ductal breast cancer. The cDNA microarray analysis was performed on the tumor samples from 10 patients. Five normal tissue samples outside the tumor were used as a pooled control samples to enable the two-channel cDNA microarray analysis. The RNA extracted from the studied and control samples was labeled and hybridized on the microarray slide containing "approximately 4000 probes of genes with proven or suspected roles in human cancer" [78]. The array contained three technical replicates (spots) of each probe.

   The spots of the scanned arrays were verified by visual inspection such that only high quality spots were taken into account. The expression value of the probe was calculated as a median of replicate spots when all spots were accepted, and as a mean value if one of the spots was rejected. The probes with less than two accepted spots were considered missing values. Then, the gene expression ratios were calculated and results were normalized by LOWESS method [93] and the data was $\log_2$ transformed.

   Genes with significantly different expression were found by determining the mean expression ratios and comparing them by the t-test. A gene was considered as up-regulated (down-regulated) if the p-value of the t-test against 0 was less than 0.05 and the average expression ratio was greater than 0.5 or less than -0.5 in case of down-regulated gene. The Gene Ontology information of the differentially expressed genes was studied with the GoMiner software [94].

As a result of the above analysis we were able to find 119 up-regulated and 224 down-regulated genes. The article states: "These (genes) represented several significantly deregulated gene groups encoding proteins participating in DNA replication, regulation of cell proliferation, protein biosynthesis, superoxide dismutase activity and cytokine production" [78]. Further immunohistochemistry analysis of proliferation-related genes suggests that there is a subgroup of the patients for which the ductal breast cancer has better prognosis when the expression of the securin is low.

3. *Kati Talvinen, Johannes Tuikkala, Juha Grönroos, Heikki Huhtinen, Pauliina Kronqvist, Tero Aittokallio, Olli S. Nevalainen, Heikki Hiekkanen, Timo Nevalainen, Jari Sundström, Biochemical and Clinical Approaches in Evaluating the Prognosis of Colonic Cancer. Anticancer Research 26(6), 2006.*

   The prognosis of the colon cancer. The colon cancer is classified to be in different stages depending on the prognosis of the patient. The most favorable class is the stage I, the stages II-III have also improved prognosis when adjuvant treatments are applied after surgery. Stage IV is the most fatal for the patient. We studied the gene expression of the tumor samples from 6 patients with different stage classification. Healthy colon tissue of the same patients was used as a reference sample.

   The analysis of the gene expression data was made similarly as in the related publication (2). As a result of mining the differentially expressed genes, 69 genes were found to be up-regulated and 89 down-regulated compared with the corresponding reference sample. The Gene Ontology analysis of these genes showed that processes such as cellular defense, cell structure, motility and cell division were found to be "notably represented among the most deregulated genes" [77]. In the further histochemistry analysis the histological grade was found to be associated with the cancer stage classification.

Bibliography

[1] T. Aittokallio and B. Schwikowski. Graph-based methods for analysing networks in cell biology. *Briefings in bioinformatics*, 7(3):243–255, 2006.

[2] B. Alberts. *Molecular biology of the cell.* Garland Science, New York, 2008.

[3] M. Albrecht, A. Kerren, K. Klein, O. Kohlbacher, P. Mutzel, W. Paul, F. Schreiber, and M. Wybrow. On open problems in biological network visualization. In *Graph Drawing*, pages 256–267. Springer, 2010.

[4] D. B. Allison, X. Cui, G. P. Page, and M. Sabripour. Microarray data analysis: from disarray to consolidation and consensus. *Nature Reviews Genetics*, 7(1):55–65, 2006.

[5] B. Andreopoulos, A. An, X. Wang, and M. Schroeder. A roadmap of clustering algorithms: finding a match for a biomedical application. *Briefings in Bioinformatics*, 10(3):297–314, 2009.

[6] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, et al. Gene Ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

[7] L. H. Augenlicht and D. Kobrin. Cloning and screening of sequences expressed in a mouse colon tumor. *Cancer research*, 42(3):1088–1093, 1982.

[8] G. H. Ball and D. J. Hall. ISODATA, a novel method of data analysis and pattern classification. Technical report, DTIC Document, 1965.

[9] M. Barnes, J. Freudenberg, S. Thompson, B. Aronow, and P. Pavlidis. Experimental comparison and cross-validation of the Affymetrix and Illumina gene expression analysis platforms. *Nucleic acids research*, 33(18):5914–5923, 2005.

[10] G. Battista. *Graph drawing: algorithms for the visualization of graphs.* Prentice Hall, Upper Saddle River, N.J, 1999.

[11] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and E. W. Sayers. GenBank. *Nucleic acids research*, 38 suppl. 1:D46–D51, 2010.

[12] M. Berthold, C. Borgelt, and F. Höppner. *Guide to intelligent data analysis*, volume 42. Springer, 2010.

[13] N. V. Bhagavan. *Medical biochemistry.* Academic press, 2002.

[14] D. Binns, E. Dimmer, R. Huntley, D. Barrell, C. O'Donovan, and R. Apweiler. QuickGO: a web-based tool for Gene Ontology searching. *Bioinformatics*, 25(22):3045–3046, 2009.

[15] J. R. Brown, R. Earnshaw, M. Jern, and J. Vince. *Visualization: using computer graphics to explore data and present information.* John Wiley & Sons, Inc., 1995.

[16] Z. Cai, M. Heydari, and G. Lin. Iterated local least squares microarray missing value imputation. *Journal of Bioinformatics and Computational Biology*, 4(05):935–957, 2006.

[17] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent systems*, 14(1):20–26, 1999.

[18] S.-H. Chen and W. Hsieh. Fast algorithm for VQ codebook design. In *Communications, Speech and Vision, IEE Proceedings I*, volume 138, pages 357–362. IET, 1991.

[19] P. Ciaccia, M. Patella, F. Rabitti, and P. Zezula. Indexing Metric Spaces with M-Tree. In *SEBD*, volume 97, pages 67–86, 1997.

[20] F. Collins, E. Lander, J. Rogers, R. Waterston, and I. Conso. Finishing the euchromatic sequence of the human genome. *Nature*, 431(7011):931–945, 2004.

[21] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[22] A. G. de Brevern, S. Hazout, and A. Malpertuy. Influence of microarrays experiments missing values on the stability of gene groups by hierarchical clustering. *BMC bioinformatics*, 5(1):114, 2004.

[23] J. L. DeRisi, V. R. Iyer, and P. O. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278(5338):680–686, 1997.

[24] P. D'haeseleer et al. How does gene expression clustering work? *Nature biotechnology*, 23(12):1499–1502, 2005.

[25] D. J. Duggan, M. Bittner, Y. Chen, P. Meltzer, and J. M. Trent. Expression profiling using cDNA microarrays. *Nature genetics*, 21:10–14, 1999.

[26] P. Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42, 1984.

[27] A. Fausett and M. E. Celebi. An accelerated nearest neighbor search method for the k-means clustering algorithm. In *FLAIRS Conference*, 2013.

[28] R. Fisher. *Statistical Methods For Research Workers.* Cosmo study guides. Cosmo Publications, 1925.

[29] R. D. Fleischmann, M. D. Adams, O. White, R. A. Clayton, E. F. Kirkness, A. R. Kerlavage, C. J. Bult, J.-F. Tomb, B. A. Dougherty, J. M. Merrick, et al. Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. *Science*, 269(5223):496–512, 1995.

[30] T. M. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991.

[31] F. Galton. Kinship and correlation. *The North American Review*, pages 419–431, 1890.

[32] K. L. Gunderson, F. J. Steemers, G. Lee, L. G. Mendoza, and M. S. Chee. A genome-wide scalable SNP genotyping assay using microarray technology. *Nature genetics*, 37(5):549–554, 2005.

[33] T. Hastie. *The elements of statistical learning : data mining, inference, and prediction.* Springer, New York, 2009.

[34] D. F. Heitjan and S. Basu. Distinguishing missing at random and missing completely at random. *The American Statistician*, 50(3):207–213, 1996.

[35] L. Hermann. Eine erscheinung simultanen contrastes. *Pflügers Archiv European Journal of Physiology*, 3(1):13–15, 1870.

[36] B. Hesper and P. Hogeweg. Bioinformatica: een werkconcept. *Kameleon*, 1(6):28–29, 1970.

[37] P. Hogeweg. The roots of bioinformatics in theoretical biology. *PLoS computational biology*, 7(3):e1002021, 2011.

[38] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[39] B. Howie, C. Fuchsberger, M. Stephens, J. Marchini, and G. R. Abecasis. Fast and accurate genotype imputation in genome-wide association studies through pre-phasing. *Nature genetics*, 44(8):955–959, 2012.

[40] T. R. Hughes, M. Mao, A. R. Jones, J. Burchard, M. J. Marton, K. W. Shannon, S. M. Lefkowitz, M. Ziman, J. M. Schelter, M. R. Meyer, et al. Expression profiling using microarrays fabricated by an ink-jet oligonucleotide synthesizer. *Nature biotechnology*, 19(4):342–347, 2001.

[41] T. Ideker, V. Thorsson, J. A. Ranish, R. Christmas, J. Buhler, J. K. Eng, R. Bumgarner, D. R. Goodlett, R. Aebersold, and L. Hood. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 292(5518):929–934, 2001.

[42] R. Jörnsten, H.-Y. Wang, W. J. Welsh, and M. Ouyang. DNA microarray data imputation and significance analysis of differential expression. *Bioinformatics*, 21(22):4155–4161, 2005.

[43] H. Kim, G. H. Golub, and H. Park. Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics*, 21(2):187–198, 2005.

[44] D. E. Knuth. Computer-drawn flowcharts. *Communications of the ACM*, 6(9):555–563, 1963.

[45] T. Kohonen. *Self-organizing maps.* Springer, Berlin New York, 2001.

[46] A. M. Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes.* F. Didot, 1805.

[47] Y. Linde, A. Buzo, and R. Gray. An Algorithm for Vector Quantizer Design. *Communications, IEEE Transactions on*, 28(1):84–95, Jan 1980.

[48] R. Lipshutz, D. Morris, M. Chee, E. Hubbell, M. Kozal, N. Shah, N. Shen, R. Yang, and S. Fodor. Using oligonucleotide probe arrays to access genetic diversity. *Biotechniques*, 19(3):442–447, 1995.

[49] R. Little. *Statistical analysis with missing data.* Wiley, Hoboken, N.J, 2002.

[50] D. J. Lockhart, H. Dong, M. C. Byrne, M. T. Follettie, M. V. Gallo, M. S. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Norton, et al. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature biotechnology*, 14(13):1675–1680, 1996.

[51] D. J. Lockhart and E. A. Winzeler. Genomics, gene expression and DNA arrays. *nature*, 405(6788):827–836, 2000.

[52] P. W. Lord, R. D. Stevens, A. Brass, and C. A. Goble. Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics*, 19(10):1275–1283, 2003.

[53] J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. California, USA, 1967.

[54] J. Marchini and B. Howie. Genotype imputation for genome-wide association studies. *Nature Reviews Genetics*, 11(7):499–511, 2010.

[55] S. A. McCarroll, F. G. Kuruvilla, J. M. Korn, S. Cawley, J. Nemesh, A. Wysoker, M. H. Shapero, P. I. de Bakker, J. B. Maller, A. Kirby, et al. Integrated detection and population-genetic analysis of SNPs and copy number variation. *Nature genetics*, 40(10):1166–1174, 2008.

[56] D. V. Nguyen and D. M. Rocke. Partial least squares proportional hazard regression for application to DNA microarray survival data. *Bioinformatics*, 18(12):1625–1632, 2002.

[57] E. F. Nuwaysir, W. Huang, T. J. Albert, J. Singh, K. Nuwaysir, A. Pitas, T. Richmond, T. Gorski, J. P. Berg, J. Ballin, et al. Gene expression analysis using oligonucleotide arrays produced by maskless photolithography. *Genome research*, 12(11):1749–1755, 2002.

[58] S. Oba, M.-a. Sato, I. Takemasa, M. Monden, K.-i. Matsubara, and S. Ishii. A bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, 19(16):2088–2096, 2003.

[59] S. Oh, D. D. Kang, G. N. Brock, and G. C. Tseng. Biological impact of missing-value imputation on downstream analyses of gene expression profiles. *Bioinformatics*, 27(1):78–86, 2011.

[60] K. Ovaska, M. Laakso, and S. Hautaniemi. Fast Gene Ontology based clustering for microarray experiments. *BioData mining*, 1(1):11, 2008.

[61] T. A. Patterson, E. K. Lobenhofer, S. B. Fulmer-Smentek, P. J. Collins, T.-M. Chu, W. Bao, H. Fang, E. S. Kawasaki, J. Hager, I. R. Tikhonova, et al. Performance comparison of one-color and two-color platforms within the MicroArray Quality Control (MAQC) project. *Nature biotechnology*, 24(9):1140–1150, 2006.

[62] A. C. Pease, D. Solas, E. J. Sullivan, M. T. Cronin, C. P. Holmes, and S. Fodor. Light-generated oligonucleotide arrays for rapid DNA

sequence analysis. *Proceedings of the National Academy of Sciences of the United States of America*, 91(11):5022, 1994.

[63] Y. Pilpel. Noise in biological systems: pros, cons, and mechanisms of control. In *Yeast Systems Biology*, pages 407–425. Springer, 2011.

[64] J. Quackenbush. Microarray data normalization and transformation. *Nature genetics*, 32:496–501, 2002.

[65] G. P. Raghava and J. H. Han. Correlation and prediction of gene expression level from amino acid and dipeptide composition of its protein. *BMC bioinformatics*, 6(1):59, 2005.

[66] B. Ren, F. Robert, J. J. Wyrick, O. Aparicio, E. G. Jennings, I. Simon, J. Zeitlinger, J. Schreiber, N. Hannett, E. Kanin, et al. Genome-wide location and function of DNA binding proteins. *Science*, 290(5500):2306–2309, 2000.

[67] N. Risch, K. Merikangas, et al. The future of genetic studies of complex human diseases. *Science*, 273(5281):1516–1517, 1996.

[68] A. I. Saeed, N. K. Bhagabati, J. C. Braisted, W. Liang, V. Sharov, E. A. Howe, J. Li, M. Thiagarajan, J. A. White, and J. Quackenbush. TM4 Microarray Software Suite. *Methods in enzymology*, 411:134–193, 2006.

[69] F. Sanger, S. Nicklen, and A. R. Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences*, 74(12):5463–5467, 1977.

[70] I. Scheel, M. Aldrin, I. K. Glad, R. Sørum, H. Lyng, and A. Frigessi. The influence of missing value imputation on detection of differentially expressed genes from microarray data. *Bioinformatics*, 21(23):4272–4279, 2005.

[71] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270(5235):467–470, 1995.

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

[73] Y. Shi, Z. Cai, and G. Lin. Classification accuracy based microarray missing value imputation. *Bioinformatics Algorithms: Techniques and Applications*, pages 303–328, 2008.

[74] T. Sørlie, C. M. Perou, R. Tibshirani, T. Aas, S. Geisler, H. Johnsen, T. Hastie, M. B. Eisen, M. van de Rijn, S. S. Jeffrey, et al. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences*, 98(19):10869–10874, 2001.

[75] E. M. Southern. Detection of specific sequences among DNA fragments separated by gel electrophoresis. *Journal of molecular biology*, 98(3):503–517, 1975.

[76] E. Southern, U. Maskos, and J. Elder. Analyzing and comparing nucleic acid sequences by hybridization to arrays of oligonucleotides: evaluation using experimental models. *Genomics*, 13(4):1008–1017, 1992.

[77] K. Talvinen, J. Tuikkala, J. Grönroos, H. Huhtinen, P. Kronqvist, T. Aittokallio, O. Nevalainen, H. Hiekkanen, T. Nevalainen, and J. Sundström. Biochemical and clinical approaches in evaluating the prognosis of colon cancer. *Anticancer research*, 26(6C):4745–4751, 2006.

[78] K. Talvinen, J. Tuikkala, O. Nevalainen, A. Rantala, P. Hirsimäki, J. Sundström, and P. Kronqvist. Proliferation marker securin identifies favourable outcome in invasive ductal breast cancer. *British journal of cancer*, 99(2):335–340, 2008.

[79] S. Theodoridis. *Pattern recognition.* Academic Press, Burlington, MA London, 2009.

[80] M. Tringham, J. Kurko, L. Tanner, J. Tuikkala, O. S. Nevalainen, H. Niinikoski, K. Näntö-Salonen, M. Hietala, O. Simell, and J. Mykkänen. Exploring the transcriptomic variation caused by the Finnish founder mutation of lysinuric protein intolerance (LPI). *Molecular Genetics and Metabolism*, 105(3):408–415, 2012.

[81] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[82] R. C. Tryon. *Cluster analysis: correlation profile and orthometric (factor) analysis for the isolation of unities in mind and personality.* Edwards brother, Incorporated, lithoprinters and publishers, 1939.

[83] E. R. Tufte. *Beautiful evidence*, volume 23. Graphics Press Cheshire, CT, 2006.

[84] J. Tuimala and M. M. Laine editors. *DNA microarray data analysis.* CSC-Scientific Computing, 2003.

[85] J. W. Tukey. Exploratory data analysis. *Reading, Ma*, 231(11):149–160, 1977.

[86] C. Walshaw. A multilevel algorithm for force-directed graph-drawing. *J. Graph Algorithms Appl.*, 7(3):253–285, 2003.

[87] D. Wang, Y. Lv, Z. Guo, X. Li, Y. Li, J. Zhu, D. Yang, J. Xu, C. Wang, S. Rao, et al. Effects of replacing the unreliable cDNA microarray measurements on the disease classification based on gene expression profiles and functional modules. *Bioinformatics*, 22(23):2883–2889, 2006.

[88] X. Wang, A. Li, Z. Jiang, and H. Feng. Missing value estimation for DNA microarray gene expression data by Support Vector Regression imputation and orthogonal coding scheme. *BMC bioinformatics*, 7(1):32, 2006.

[89] J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

[90] J. D. Watson, T. A. Baker, S. P. Bell, A. Gann, M. Levine, R. Losick, and I. CSHLP. *Molecular biology of the gene.* Pearson/Benjamin Cummings; Cold Spring Harbor Laboratory Press, San Francisco; Cold Spring Harbor, N.Y., 6th edition, December 2007.

[91] J. J. Wyrick, F. C. Holstege, E. G. Jennings, H. C. Causton, D. Shore, M. Grunstein, E. S. Lander, and R. A. Young. Chromosomal landscape of nucleosome-dependent gene expression and silencing in yeast. *Nature*, 402(6760):418–421, 1999.

[92] Y. H. Yang, M. J. Buckley, and T. P. Speed. Analysis of cDNA microarray images. *Briefings in bioinformatics*, 2(4):341–349, 2001.

[93] Y. H. Yang, S. Dudoit, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T. P. Speed. Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic acids research*, 30(4):e15–e15, 2002.

[94] B. R. Zeeberg, W. Feng, G. Wang, M. D. Wang, A. T. Fojo, M. Sunshine, S. Narasimhan, D. W. Kane, W. C. Reinhold, S. Lababidi, et al. GoMiner: a resource for biological interpretation of genomic and proteomic data. *Genome Biol*, 4(4):R28, 2003.

Publication 1

Johannes Tuikkala, Laura Elo, Olli S. Nevalainen, Tero Aittokallio, Improving Missing Value Estimation in Microarray Data with Gene Ontology. *Bioinformatics* 22(5):566-572, 2006.

Publication 2

Johannes Tuikkala, Laura L. Elo, Olli S. Nevalainen, Tero Aittokallio, Missing Value Imputation Improves Clustering and Interpretation of Gene Expression Microarray Data. *BMC Bioinformatics* 9(1):202, 2008.

Publication 3

Johannes Tuikkala, Heidi Vähämaa, Pekka Salmela, Olli S. Nevalainen and Tero Aittokallio, A multilevel layout algorithm for visualizing physical and genetic interaction networks, with emphasis on their modular organization. *BioData Mining*, 5:2, 2012.

Publication 4

Olli Luoma, Johannes Tuikkala, Olli S. Nevalainen, Accelerating GLA with an M-Tree. In *Proceedings of the Second World Enformatika Congress (WEC'05)* (2):196-199, 2005.

Publication 5

Kati Talvinen, Johannes Tuikkala, Marjukka Nykänen, Anssi Nieminen, Jorma Anttinen, Olli S. Nevalainen, Saija Hurme, Teijo Kuopio, Pauliina Kronqvist, Altered expression of p120 catenin predicts poor outcome in invasive breast cancer. *Journal of Cancer Research and Clinical Oncology*, 136(9):1377-1387, 2010.

Turku Centre for Computer Science TUCS Dissertations

ISBN 978-952-12-3126-1
ISSN 1239-1883

Johannes Tuikkala

Algorithmic Techniques in Gene Expression Processing

Johannes Tuikkala

Algorithmic Techniques in Gene Expression Processing: From Imputation to Visualization