Universität Bielefeld

Technische Fakultät
Center for Biotechnology (CeBiTec)

EMMA2

A MAGE-Compliant System for the Analysis of Microarray Data in Integrated Functional Genomics

Zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften an der Technischen Fakultät der Universität Bielefeld vorgelegte Dissertation

von

Michael Dondrup

29. Januar 2007

Michael Dondrup
Rubensweg 2
33613 Bielefeld
`mdondrup@cebitec.uni-bielefeld.de`

Supervisors:	Prof. Dr. Robert Giegerich
	Prof. Dr. Alfred Pühler

Gedruckt auf säurefreiem, alterungsbeständigem Papier. Printed on acid-free paper.

Summary

Since the acquisition of the first complete genomic sequences, many advances have been made in the field of functional genomics. High-throughput methods have been developed to study gene-expression and metabolic pathways.

Microarrays have become a highly popular method to measure the transcriptional regulation in functional genomics. Microarrays allow to measure the expression levels of thousands of genes in parallel, but the measured datasets contain a certain level of technical and biological variation.

Many methods for the analysis of large datasets from error-prone mircorarray experiments have been developed, including normalization, statistical inference, and machine learning. Attempts to standardize the annotation of microarray data, such as Minimum Information About a Microarray Experiment (MIAME), the MAGE-ML format for data interchange, and ontologies, have been made.

The existing software systems for microarray data analysis have only rudimentary implementations of the mentioned standards and are hard to extend.

The EMMA2 software has been designed to resolve these shortcomings. Its specification includes full support of MIAME and MAGE-ML as well as the support of ontologies. Integration of genomic annotation data and other internal and external data-sources has been an important requirement.

The specification, design, and implementation of EMMA2 follows an object-oriented development paradigm. This is reflected in the use of object-oriented modeling tools such as the Unified Modeling Language (UML).

During the design phase, the MAGE object-model was taken as the core of the application to model microarray data and their annotations. Additional models were needed to complement MAGE by classes for access control and data analysis.

The software has been implemented using a code-generation approach. The back-end code and database definitions have been derived from the joint object model defined in UML. EMMA2 can be used via a web-interface and contains a Laboratory Information Management System (LIMS) component.

A flexible PlugIn system for data analysis, which includes methods for pre-processing, normalization, statistical tests, cluster analysis, and visualization, has been added. Integration of other functional genomics data sources has been implemented by using the integration layer BRIDGE and also by the use of web-services. Data integration allows for several new visualization components using metabolic pathway data and functional categories.

The system is successfully applied in eight national and international projects. More that 2700 microarrays have been processed using EMMA2. Furthermore, an evaluation study has been carried out to compare the performance of inference tests for microarrays. As a result of this study, two methods (SAM an CyberT) can be recommended for experiments with very few replicates, while for larger numbers of replicates the t-test performs comparable.

Contents

List of Figures

List of Tables

. . . Men at first made use of the instruments supplied by nature to accomplish very easy pieces of workmanship, laboriously and imperfectly, and then, when these were finished, wrought other things more difficult with less labour and greater perfection; and so gradually mounted from the simplest operations to the making of tools. . . , to the making of more complex tools. . . , till they arrived at making with small expenditure of labour, the vast number of complicated mechanisms which they now possess.

So, in like manner, the intellect, by its native strength, makes for itself intellectual instruments, whereby it acquires strength for performing other intellectual operations, and from these operations gets again fresh instruments, or the power of pushing its investigations further, and thus gradually proceeds till it reaches the summit of wisdom.

Benedict de Spinoza (1632–1677), *De intellectus emendatione.* (On the Improvement of the Understanding, cited from the Unabridged Elwes Translation, Dover Publications, New York)

CHAPTER 1

Motivation and Overview

The introduction of microarray technology has marked a paradigm shift in genomics. Thus far, research was mainly focused on single or small sets of genes of interest. These genes were analyzed closely to reveal their function. For the first time, the introduction of microarrays provides a holistic view on the expression of the whole genome of an organism. While whole genome expression analyses offers fascinating new insights into the molecular machinery of the cell, there are also new challenges for data handling and analysis: The massive amount of data whole genome studies create.

In spite of the fact that there were already lots of well established methods for multivariate analysis, management and data-mining of large data sets, it took some time to adapt them to this new area of research. The adoption of known methods and the invention of new specialized methods proceeds concurrently with the introduction of microarray technology. The research community has experienced a vast trend of growth in the application of high-throughput methods, and in particular microarrays, over the last decade. Microarray studies have given rise to interesting publications and successive research projects. Despite their apparent success, microarrays suffer from influences of variation. This is of course true for any measurement technology, especially in biology, where the variation of the subject of measurement can be high. Nevertheless, the technology needs to be constantly improved.

The number of methods for microarray analysis is increasing, as well. Some of the authors of new methods claim to have them designed with a focus on noisy microarray data. There is still no clear guidance on the merits of all the new methods. Thus, comparative studies of methods are required but there are only very few independent evaluations.

Microarray analyses certainly offer a high potential of achieving reliable results if they are well planned and conducted. Cost efficiency is a major goal in many studies, and this means to study a maximum of interesting variables with a minimum of experiments and repetitions. These goals seem contradictory, but with good preparation, planning, and thorough knowledge of applicable methods, they are achievable. As the resulting data are costly, we need good tools to record the complete experimental process and also the analysis methods applied. This is so important because the results cannot be evaluated without knowing the experimental conditions under which they were obtained. There are now some evolving standards about which information to record and which formats to use for data-interchange.

Regarding the recent advances in the technology, its opportunities, the possible pit-falls, and the number of methods for data analysis, a system is needed to help the users to reliably store and process their data.

Standardization is required for storage and the most important process of data analysis. We need a certain level of guidance on the applicable methods, in particular for users without deep knowledge in statistics. This includes to evaluate the flood of new methods developed and to give recommendations on which method to use. It is truly a broad vision, hardly feasible to accomplish in an isolated environment, without close cooperation between the computer-science side and the users, the experimenters.

The Center for Biotechnology (CeBiTec) was established at Bielefeld University in 1998 as an interdisciplinary institution. Intended as a joint effort to couple research groups, it serves as a platform for cooperation and knowledge transfer between biology, computer-science, chemistry and physics. The relevance of computational methods and infrastructure was recognized early on within the CeBiTec. It was therefore equiped with a large service unit of computer-scientists providing central computer hardware, software, and support. This unit was termed Bioinformatics Resource Facility (BRF).

Within this environment, a computational infrastructure could be developed to utilize microarray technology. A novel standards-compliant system, baptized EMMA2, was developed. For the development process of EMMA2, close cooperation between users and developers was found to be very beneficial. With having all the data from a large microarray core-facility at hands, and especially with the input of a large community of users from all over the world, EMMA2 has undergone an evolutionary development process.

All steps in the workflow of an experiment are now standardized, ranging from acquisition of experimental protocols and annotations to handling of raw and transformed data. Further progress towards standardization of the whole analysis steps and evaluation of methods, for example normalization and statistical tests, have been made. The availability of the EMMA2 system, has contributed to many fields of reasearch.

Anyway, the system would not be complete, without a close integration of other data-sources such as genome annotations, metabolic pathways and proteome measurements. To provide a system to the international research community, which would allow for terse integration of these sources was another aim of this project. For integration, naive hyperlinking every gene to a data-base was simply not enough. Instead, a complete bi-directional implementation to include all sorts of external annotations into computational methods was imagined.

In the following Chapters, design, development, implementation, and application of EMMA2 is described. Chapter 2 begins with a brief overview on the biological foundation of gene-expression and regulation. Furthermore, the vast new field of functional genomics is introduced.

Chapter 3 introduces the reader to the fundamentals of microarray technology. Also, there is a focus on possible problems and pit-falls one might encounter while working with microarrays; some methods to overcome these influences, including normalization and statistical inference will be presented that allow to make sense out of noisy data.

Chapter 4 is dedicated to all of the most progressive methods of standardization and software systems in the field of microarrays. Standardization is introduced as a complex process that involves defining the content of communication, machine-readable formats and vocabularies, and finally, pieces of software implementing them.

Modern software development can be described as an evolutionary cycle; as the first step it is necessary to obtain a sound impression of what the software should do and how it should look like. There are many aspects to requirement analysis. They come from very different motivations, not to forget the challenge of the author to deliver a refined piece of software with previously unseen features. Requirements may pile up and so they have to be collected, ranked and condensed into a half-formal specification described in Chapter 5.

Chapter 6 deals with the more formal aspects of designing a system. It is almost infeasible to build a complex application as a monolithic piece of source-code. One should rather use a modular approach and decompose the whole system into smaller reusable and manageable portions communicating with each other via well defined interfaces.

Now, it is time to fill the components with source code and finally bring to life the working application. Chapter 7 describes in detail the implementation of EMMA2, which other software and programming languages were applied, and how their employment results in an extensible database system with a highly versatile interface.

After having built a system, it is most important to test the applicability of EMMA2 in real-world projects. There are currently many projects which apply EMMA2. Some are internal evaluation projects, dealing with assessment of new methods of statistics and visualization. By far the largest range of projects is dedicated to microarray research in a diverse range of organisms and environments. How much EMMA2 can contribute to academic research can be figured out by reading Chapter 8 on “Results and Applications”. Finally, a discussion of the system, its implementation, and new insights gained is given in Chapter 9. An outlook on the perspectives of EMMA2 in microarray research and its integration with other systems at the CeBiTec concludes the work.

CHAPTER 2

Introduction

At which point in the development of a method, technology or tool, is it justified to call it a well-established one? Is it the point in time where it is so popular that everyone in the field knows the method exists and attributes high expectation to it? In biotechnology, this could be the case when major (non-scientific) media report on the method as a candidate to help to find the ultimate cure for cancer. Or is it at a later stage, when the initial sensational promotion is replaced by a more down-to-earth view on the capabilities and deficiencies of the tool? Hopefully and finally, a tool or method can reach a state where it is so ubiquitous that its presence is hardly taken notice of – except in the case of failure or malfunction.

For microarrays, we have, almost certainly, overcome the state of euphoria, reaching a turning point at which we gain a better perspective on the benefits this tool has to offer as well as the problems it poses for the researcher. Since their introduction, in a first study in 1995, they have played a key role in the functional analysis of genomes. Although, there have been other techniques to measure gene expression in cells, such as Northern blots and targeted macroarrays, microarrays allowed the measurement of RNA abundance for thousands of genes in parallel for the first time.

Many expectations have been raised by this new ability, especially for biomedical research. Although, some of the most high-flying hopes have not been fulfilled up to now, microarrays provide an excellent measurement technology for gene-expression analysis. This is especially the case when used in combination with other laboratory techniques.

Cost efficiency is one of the major arguments for the introduction of microarrays. Its relevance can also be deduced from the large and growing amount of microarray experiments submitted to public repositories. As of July 2006 the public microarray repository ArrayExpress located at the European Bioinformatics Institute contained more than 1500 experiments consisting of a total of over 45000 microarrays. As of December 2006 these figures have increased to more than 1700 experiments and 52000 arrays.

The biological object of study – from the genes of an organism to the development of cell and their observable phenotype – and the primary process which microarrays target is a complex one. 'Regulation' is used as a short term to summarize the constant adaptation of an organism to its environment. Many entry points for regulation have evolved during evolution of life. Improved regulation allows an organism to better adapt to changes in its environment providing an advantage over other competitors. Regulation of gene expression is an important mechanism among other mechanisms of adaptation such as metabolic regulation, motility or intercellular signalling, and only the first step of gene regulation are covered by microarray analyses.

2.1 Information Flow and Regulation in the Cell

The central dogma of molecular biology describes protein expression as a directed flow of information: from DNA over intermediate messenger RNA (mRNA) towards the end product, a protein. The nucleotide sequence of DNA encodes the amino acids sequence of final proteins. DNA carries the inheritable traits of an organism. The resulting proteins perform primary roles in the metabolism and serve as structural elements of all organisms. Figure 2.1 on the facing page summarizes the central dogma and the molecular machinery involved (Lewin, 2004).

Not all genes are equally expressed under any given condition. The biological process of gene expression is a multi-step process including several stages of regulation. The molecular mechanisms of gene expression and its regulation vary in essential details between the domains of life: eukarya, bacteria, and archea (and also for plastid genomes of eukaryotes and for viruses). These differences affect the molecular equipment of the cell and further processing steps at the mRNA level as eukaryote cells have a far more complex structure than a bacterial or archeal cell and the mRNA has to be transported between cell compartments. On the opposite, in bacteria, there is fewer room for modifications of the mRNA, as the translation into protein is performed directly on the transcribed mRNA strand in proximity to the replicon. Despite this, the central dogma of molecular biology is common to all domains of life.

At the first stage, regulation can occur on DNA-level, for example by methylation of nucleotides of the DNA strand. The most well-studied regulatory mechanisms exist on the transcription level, the synthesis of a complementary strand of mRNA from the chromosomal DNA. This process is carried out by the enzyme RNA polymerase (RNApol).

The process of transcription is divided into three phases: During the *initiation phase*, RNA polymerase attaches to the DNA at a specific binding site of a gene,

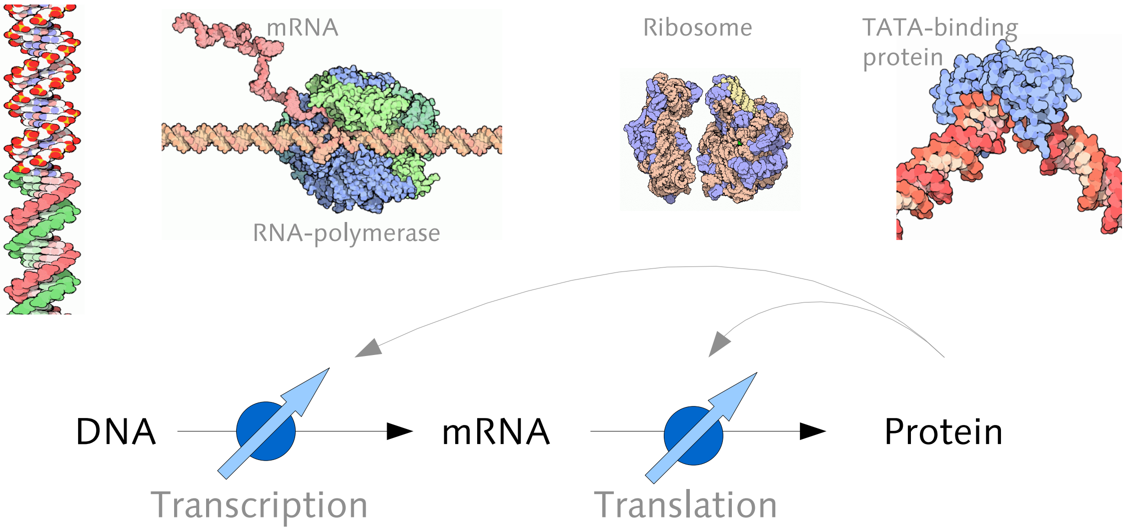


Figure 2.1: A computer scientist's view on the regulation of protein biosynthesis. In the top row, the key molecules involved in gene expression are depicted (molecule images are taken from the Protein Data Bank (`http://www.pdb.org`), entries 1bna: Drew *et al.* (1981), 1i6h: Gnatt *et al.* (2001), 1ffk: Ban *et al.* (2000), 1ytb: Kim *et al.* (1993)).

the promoter, and forms the closed complex. The double-stranded DNA is opened. After the open complex is formed, the RNA polymerase starts transcription of the template strand at the transcription start site. Transcription occurs in a small open region (of ≈ 12 basepairs), the transcription bubble. In the consecutive *elongation phase*, the RNA polymerase moves forward on the DNA strand and opens it while elongating the RNA strand. The produced RNA sequence is complementary to the template strand. After passing of the RNA polymerase, the transcription bubble is closed again and the DNA forms a duble-strand. The *termination phase* is entered when the RNA polymerase encounters a termination signal. It detaches from the DNA and the RNA transcript is released.

Global transcriptional programs can be activated by alterations of RNA polymerase, which is a complex molecule consiting of multiple subunits. In bacteria, the core enzyme consists of five sub-units and has unspecific affinity to DNA. Another co-factor is required for the RNA polymerase to specifically detect promoter sequences, the σ-subunit (also called σ factor). The core enzyme and σ form the holo-enzyme, which has a specific affinity to the bacterial promoter sequences. There are multiple σ-factors in the genome of bacteria that detect different promoter sequences. Global transcriptional programs can be observed by exchanging σ-factors in reaction of bacteria to global changes in their environment, for example an increase of temperature.

In eukaryotes and archea RNApol consists of ≥ 10 subunits and cannot initiate transcription directly. Transcription factors are required to place the RNA polymerase in the exact position and assist the formation of the open complex. Transcription factors are by definition those molecules that are not part of RNApol but are required for the initiation of transcription. In eukaryotes and archea, the transcription factors bind specific AT-rich consensus sequences and often form larger complexes of transcription factors and co-factors. An essential molecule is the TATA binding protein (TBP, a drawing is shown in Figure 2.1 on the previous page). It is a sub-unit of the transcription factor TFIID which recruits other factor necessary to initiate transcription. In addition, eukaryotes (but neither bacteria or archea) possess three specialized RNA-polymerases: (RNApol I for ribosomal RNA, RNApol II for protein coding mRNA, and RNApol III for other RNA genes).

Binding efficiency and transcription is further modulated by regulatory proteins and their binding to the DNA-strand. They are also called *trans*-acting regulatory elements, because the genomic sequence coding for them is often far away from the target of regulation. The binding site of regulatory genes is called *cis*-acting site and is often in the vicinity of the promoter. In bacteria, there exist regulatory sites of positive and negative control.

Regulatory proteins of positive control enable the RNA-polymerase to bind the DNA-strand more efficiently, while regulatory proteins of negative control inhibit transcription. Activators are examples *trans*-acting elements of positive control in bacteria. Repressors have the opposite regulatory effect. The actual regulatory influence depends on the combination of *trans* and *cis* elements. There can be combinations of proteins and binding sites that reverse the regulatory effect.

The activity of regulatory proteins is often further modulated by other molecules, so called effectors. In one of the most well studied examples for regulation of bacterial gene expression, the Lac-operon[1] in *Escherichia coli*, the presence of the molecule allolactose, an isoform of the sugar lactose, modifies the repressor of this operon. While the repressor binds to the operator sequence in the absence of allolactose (and thereby lactose) inhibiting the expression of genes of the lactose metabolism, in presence of allolactose the repressor changes its conformation and detaches from the operator, hence allowing the transcription of the genes in the operon.

In eukarya and archea, there exist enhancers as additional *cis*-regulatory elements. They can sometimes be relatively distant from the promoter sequence, but still influence the rate of transcription by enabling trans-acting factors to bind to them.

In prokaryotes, the nascent mRNA is translated by ribosomes which directly attach to it before transcription is complete. In eukaryotes, the pre-mRNA is further processed. A capping is added to the 5'-end of the pre-mRNA; after the transcript is released, a poly-A tail is added to the 3'-end. Furthermore, non-coding sections (introns) are removed from the transcript in a process of splicing. The mature mRNA is afterwards transported to the cytoplasm through a pore of the nucleus. As a result of the processing, eukaryotic mRNA has a much longer

[1] An operon is a transcription unit in prokaryotes which consists of possibly many genes which are transcribed into a single *polycistronic* mRNA.

half-life than prokaryotic mRNA.

Variable decay-rates of different mRNA transcripts can also be encountered and seem to play a role in regulation. Other regulatory elements have been discovered, some of which are themselves RNA namely micro RNA (miRNA). These molecules function in combination with protein complexes, modify the mRNA, or interfere with the initiation or termination of the translation process. This can lead to complete inhibition of the translation of transcripts. This mechanism is also known as RNA interference (RNAi).

Control of the translation process can be seen as a further means for the cell to regulate its gene expression profile. In addition, post-translational modification of proteins includes changes in the conformation of the 3-dimensional structure of proteins. The process of protein-folding, namely the process of the formation of a 3-dimensional structure, is even more complex and only partially understood.

Despite these problems, knowledge about protein structure is important for the study of transcriptional regulation to automatically detect trans-acting elements. Regulatory proteins often exhibit specific amino-acid sequences, so called DNA binding domains. These domains are of particular interest to predict candidates for regulatory proteins and can be predicted using computer programs to search against databases of protein domains (for example PFAM (Sonnhammer *et al.*, 1998) using the hmmer tools). The TRANSFAC database contains a large number of transcription factors, regulated genes and corresponding binding sites for eukaryotes (Matys *et al.*, 2006).

The problem of predicting binding sites from the databases and the matching of DNA sequence and its protein domain counterpart, would be an interesting next step. It remains unsolved by direct inference from the sequence or with search patterns or weight matrices for binding sites (Rahmann *et al.*, 2003).

Signals, which influence transcription and translation, are emitted by an only fragmentally understood cellular signalling network. Resulting from these insights, information flow is not simply linear from DNA to protein, but is seen as a complex network of regulatory processes between DNA, RNA, proteins, metabolites, and environmental conditions. It appears that RNA, and thus its measurement, plays a key role in this regulatory network, apart from the RNA being reduced to an intermediate product of protein-expression. In fact, RNA is the only bio-molecule, that has the ability to encompass many roles. RNA can serve as a means of inheritance, a regulatory molecule, and also have an enzymatic activity.

Many bioinformatics methods have been developed to predict the secondary structure of RNA, that is formed by self-hybridization of a single strand. It is believed that these secondary structures play an important role in regulation of expression as well.

In summary, from the role that is attributed to RNA in modern functional genomics, it appears that we are experiencing the formation of an 'RNA world'[2],

[2] This term is otherwise used for the hypothesis stating that RNA was the first stable bio-polymer in the origin of evolution of life.

where the measurement of the transcriptional state of an organism is one of the most prominent techniques in biological research.

2.2 The New 'Omics Approaches

It is often said that biological science has reached the 'post-genomics era'. The acquisition of the complete genomic sequence of an organism has become an almost routine task by the development of ever faster DNA-sequencing techniques. In fact, DNA-sequencing landmarks the introduction of the first so called high-throughput method into modern genomics.

Other ground-breaking technological advances are the introduction of *de-novo* protein-sequencing, metabolic analysis, sequencing of ESTs and last but not least several tools, made to study transcription such as Serial Analysis of Gene Expression (SAGE Velculescu *et al.*, 1995) or microarrays. Other high-throughput methods were developed which allow the creation of genome-wide deletion mutants for microorganisms.

The availability of the first complete genomic sequences has created the need to determine which part of the sequence represents genes in the sense of classical genetics and thereafter to determine the function of these genes.

All newly developed technologies yield high-volume data, which has led to an increased necessity for application of computers and algorithms formerly unseen in biology. The emerging field of bioinformatics arose simultaneously with the new functional genomics methods and soon gained an irrevocable stance within genomics.

Algorithms and computational infrastructure have been developed to analyze and store large genomic datasets. A good example is the BLAST heuristic for sequence alignment, presumably up to now the most widely used bioinformatics method. The development of BLAST was fostered by the need to analyze the large amount of newly obtained genomic sequences more efficiently.

The term 'functional genomics', despite its vague use in literature, is often used to describe the effort of going beyond the mere genetics and DNA-sequence acquisition, to make use of all kinds of different sources of data, and to use them in combination with classical and newly developed laboratory techniques. Combining many applicable laboratory techniques and data processing methods, it seems feasible to describe the functional elements of the cell on all stages from geneexpression, over metabolic pathways to cellular signalling.

Another example are microarrays which allow the parallel measurement of tenthousand of transcripts of an organism. Microarray analysis results in large numerical datasets, which are completely different from the string-type data DNA-sequencing produces. Bioinformatics algorithms are required to handle such large datasets, while dealing with the high level of uncertainty exhibited by the new type of measurement .

The unique aspects of functional genomics have already created a significant number of novel insights into the molecular machinery of life. As a consequence, there is also increasing commercial potential for the new techniques. For medical reasearch, many of the functional genomics methods are regarded to hold a high potential. The new sector of life-science companies has fostered the development of out-of-the-box methods for large scale screening and data analysis.

Functional genomics methods could be able to shed new light on questions of genetics, for example the essential definition of what is a gene. It is amazing that, though the term is of such primary importance, no uniformly accepted definition could be found. In fact, the question of "what is a gene" is very closely related to the origin of the intermediate product, mRNA, which can measured by microarrays.

2.3 Analysis of Transcriptional Regulation – Goals and Visions

The upcoming high-throughput methods to study the transcriptome (a term which, parallel to the term genome, denotes the entirety of transcripts of an organism) have raised high expectations, similar to the expectations that were raised by the human genome project. Most hopes are directed towards medical applications of microarrays, for example studying cancer and other diseases to improve therapeutic treatment.

One of the simplest questions which can be answered by any method for studying transcription is, whether a set of genes is expressed under the conditions studied. Some methods such as quantitative RT-PCR (Wong and Medrano, 2005) allow to assess the actual number of transcripts on a quantitative level, while others, for example EST-sequencing, allow to determine the exact sequence of the transcripts *de-novo*.

Another approach, which is followed by a vast number of researchers, is to determine gene-function by patterns of common regulation between genes. Often, genes which function in a similar metabolic pathway or share another common function, show similar patterns of gene-expression in transcription profiling experiments. This approach has been termed 'guilt-by-association' in an editorial by Quackenbush (2003).

Inference of function based on 'guilt-by-association' has often been criticized for its vagueness of definition and lack of stringency. If 'guilt-by-association' is over-interpreted, one could get the impression that the function of the genes is encoded in their expression profile, which is dubious at best. Despite possible misconceptions, the approach provides a useful source of new hypotheses about co-regulated genes. Especially, when the expression of conserved genes can be compared over multiple species, for instance in the study of Stuart *et al.* (2003).

The molecular machinery of the regulation of gene expression is controlled by a multitude of cellular players, for example effectors and transcription factors. It is often described in terms of a complex network structure. Measurement of transcription under different conditions using mutants created by functional genomics laboratory methods can help to elucidate the structure of these regulatory networks and to identify the role of the involved molecules.

By elucidating the underlying mechanism of transcriptional networks, researchers hope to gain more effective therapies for diseases, to optimize food production, or to better understand pathogenic and non-pathogenic microorganisms. Far reaching applications of the array technology are prospected for the future. Schena (2003) prospects a wide range of clinical applications mainly related to civilisatory related health care issues, from the measurement of physical fitness, over predisposition for alcohol or drug abuse, to diagnostic tools for depression and schizophrenia. These are quite reasonable fields of application, but it does not seem too daring to presume that there is still a long way to go for the technology to become a standard tool of clinical diagnosis.

2.4 The Role of Free Open Source Software in 'Omics Research

The introduction of high-throughput techniques into biology has marked the beginning of the new disciplines of structural and functional genomics. Genome-wide analyses produce such large amounts of data that the use of computers and specialized software for storage and further analysis of the 'data flood' is mandatory. The more data sets exist, the larger the need for specialized applications for searching and comparing them. The task to find regularities and structure is known as data-mining.

Due to the sheer volume of the resulting data the experimenting biologists require strong support with respect to bioinformatics and also knowledge about the applicable algorithms and statistical methods of inference. This need has recently been pointed out by Miron and Nadon (2006); they have used the term 'inferential literacy' for this ability.

The algorithms and database systems needed to store, retrieve, and comprehensively analyze genome-wide datasets have to be provided in the form of software. This can be either proprietary commercial software or free open-source software. It is often hard to discriminate these forms of software exactly. In general, proprietary software applications require payment to acquire a, sometimes temporary, license to use the product, while free open-source software (FOSS) is provided, often by a large community of contributors, with an open license and with full access to the source code.

As many of the proprietary applications are tailored for the field of life-science, where possibly high revenues are to be expected from pharmaceutical or agricultural research, these software products are often priced as other business related software systems. The pricing and licensing policy often renders these applications unaffordable for academic institutions. On the other hand, concerns about a lack of quality, reliability, and economic advantages have often been raised as a counterargument against FOSS, in particular by vendors of proprietary systems. But Wong and Sayo (2004) and a large EU study on open source software, published recently,[3] demonstrate that FOSS is generally a powerful concept with a relevant economic impact.

An advantage of free open-source software is in particular the availability of the source code, which enables the user to inspect the source code and to become aware of the exact implementation of the algorithms used in a software package. Knowledge of the exact algorithm is a key feature for academic research and the publication of results based thereon.

The availability of the source code makes it possible to fix software errors or add new functionality. While this might not be applicable to the individual lab-researcher, the availability of the source code can create a large community of contributors, which share new or optimized source code. That way, software projects of no direct commercial interest or merchantability can be realized; as for example, public data repositories for gene-expression data are implemented as free open source solutions. Often, such free open-source projects are also supported by large companies.

[3] `http://ec.europa.eu/enterprise/ict/policy/doc/2006-11-20-flossimpact.pdf`

CHAPTER 3

Analysis of Gene-Expression – A Brief Introduction

Microarray technology has undergone a fast evolution during the late 1990's. Microarrays allow the parallel measurement of transcriptional regulation of thousands of genes (Schena *et al.*, 1995; Lipshutz *et al.*, 1995). Therefore, microarrays are often called a high-throughput technique (Lipshutz *et al.*, 1999; MacBeath and Schreiber, 2000; Miron and Nadon, 2006; Küster *et al.*, 2006). Microarrays have marked a turning point in functional genomics due to their wide range of applications and relative cost-efficiency. Since then, a variety of protocols for producing and applying microarrays in the life-sciences have been developed.

3.1 Microarray Technology

Despite all technological differences, the common principle of all microarray platforms is rather straight-forward. DNA molecules having defined nucleotide sequences are attached to the surface of a solid support, mostly coated glass. They are named reporter molecules or simply reporters (originally they were termed *probes* in the pioneering publication of Schena *et al.*) and share a small area of the surface. These regions are called features or spots and are arranged in a regular pattern. Current technology reaches a density of up to 10,000 features per cm^2.

To measure messenger RNA, it is extracted and optionally converted into copy DNA (cDNA) by reverse transcription using reverse transcriptase, an enzyme discovered in RNA-viruses. The RNA or cDNA is labeled with a marker substance allowing for approximate quantification of the number of DNA copies. The solution

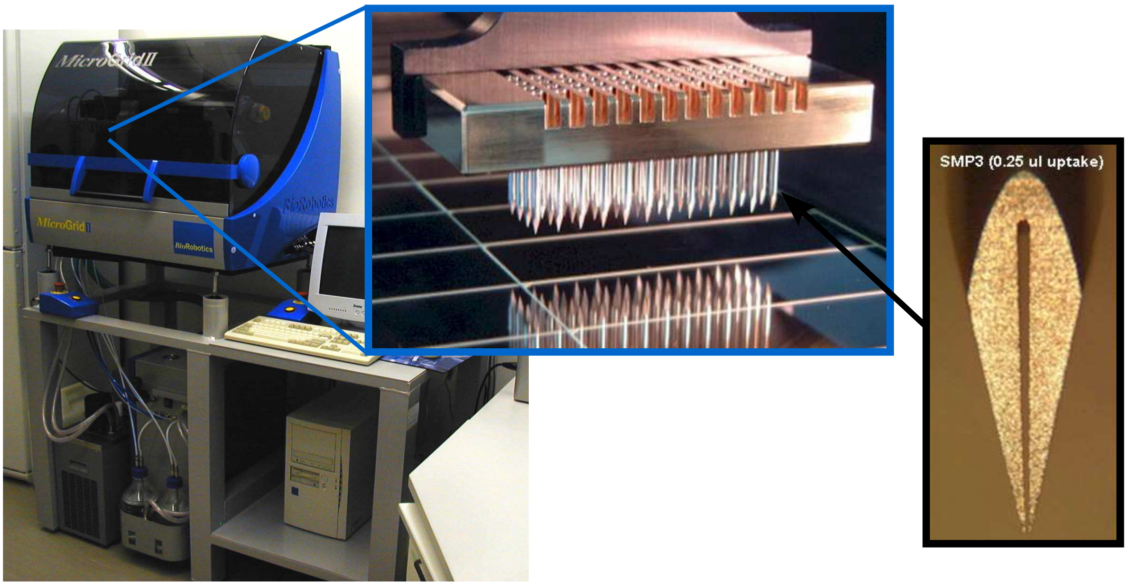


Figure 3.1: Robotic microarray spotter (Microgrid II) used at the CeBiTec (left), an enlarged view of the print head with 48 print-tips (center) and a print-tip (right).

of labeled molecules (also called *targets*) is then put on the surface of the microarray containing the reporters. In a process of parallel hybridization, the labeled single-stranded RNA or cDNA molecules (secondary structures of RNA and cDNA are resolved) hybridize with their single-stranded counterparts, representing the nucleotide sequence of the complementary strand on the surface of the microarray. The approximate number of bound target molecules in a given feature is measured by a detection device, often a scanner producing images.

Despite the large variety of techniques for microarray production, two main technological platforms can be differentiated: spotted two-color microarrays and *in-situ* synthesized microarrays (often referred to as ‘DNA-chips’).

3.1.1 Spotted Two-Color Microarrays

On spotted microarrays the reporter molecules are attached to a coated glass or plastic substrate (Schena *et al.*, 1995). A robotic spotting device (also often called spotter or arrayer, see Figure 3.1) is used to deposit a small volume (typical: $v <$ 1nl) of a solution of defined DNA molecules onto the surface of the substrate. The approximate DNA concentration of the spotted solution is typically $\leq 20\mu$M, whereas a typical feature has a diameter of $100\mu\text{m} \geq d \geq 50\mu\text{m}$.

The molecules are attached to the surface by covalent binding to molecules of the surface coating. The reporter molecules can be either generated by polymerase chain reactions (PCR), a standard technique to produce many copies of DNA fragments using the enzyme DNA polymerase, or they can be synthesized oligonucleotides. Typical lengths of PCR fragments for spotted microarrays range from approximately 150 to 300 base pairs. Oligonucleotides for spotted microarrays are normally produced with fixed lengths between 10 and 80 base pairs, depending on the field of application. Advantages of oligonucleotides over PCR fragments are cost efficiency and well-defined sequence features, for example the melting-point and the ability to control design features to reduce cross-hybridization with other genes of the organism. In contrast to oligonucleotides, which can be designed to bind to short and highly unique parts of a transcript with respect to genome, PCR fragments have an increased cross-hybridization probability due to their greater length. Also, PCR reactions may fail and yield no or reporter molecules with unexpected sequences for a small number of genes.

Novel reporters consisting of modified nucleotide molecules. Locked Nucleic Acids (LNA) or a mixture of DNA and synthetic molecules have been introduced by commercial vendors. These molecules are designed to hybridize cDNA like DNA reporters, but are build up from chemical compounds providing an increased stability and binding affinity (Braasch and Corey, 2001; Liu *et al.*, 2006). Synthetic oligomers have to be synthesized as they cannot be produced in PCR reactions with conventional DNA polymerase. As a consequence, synthetic modified oligomers can at present be produced only by commercial vendors. Despite their technical advantages, a wide-range application of synthetic oligomers in academic institutions is impeded by the resulting high costs.

Each spotted microarray can be used to compare gene expression between two different labeled extracts. The labelled extracts can stem from different environmental conditions or be derived from a comparison of different time points in the developmental process of the cell. Spellman *et al.* (1998), for example, have compared the different time-points in the cell cycle of the budding yeast *Saccharomyces cerevisiae* using spotted microarrays, Laub *et al.* (2000) compared several stages during the cell-cycle of the bacterium *Caulobacter crescentus*. Other applications include the comparison of different cell lines, mutants, or various tissue types such as cancer cell lines (Ho and Lau, 2002; Ramaswamy and Golub, 2002). mRNA is extracted from the cell cultures to be compared, reverse transcribed into cDNA and labeled with two different marker substances (see Figure 3.2 on the next page). The marker substances are attached to modified nucleotides used to synthesize the cDNA. These marker substances are mostly fluorophores such as Cyanine fluors (e.g. Cy3 and Cy5) and Alexa fluors (e.g. Alexa647 and Alexa555). Fluorophores absorb and emit light in a small range of the electromagnetic spectrum. Novel marker substances include nano particles and so called quantum dots (Huber *et al.*, 2004; Pedroso and Guillen, 2006).

The amount of marker substances bound within the feature area is evaluated using image scanners which produce a digital image. For fluorescent markers, a laser is used causing a fluorescent emission of light from the fluorophores. For two-color microarrays, these scanners produce two images, one for each emission wavelength. Currently, there are ongoing attempts to use more than two different dyes to increase the number of concurrent conditions that can be tested on one microarray. Many microarray scanners can be equipped with up to four detection channels.

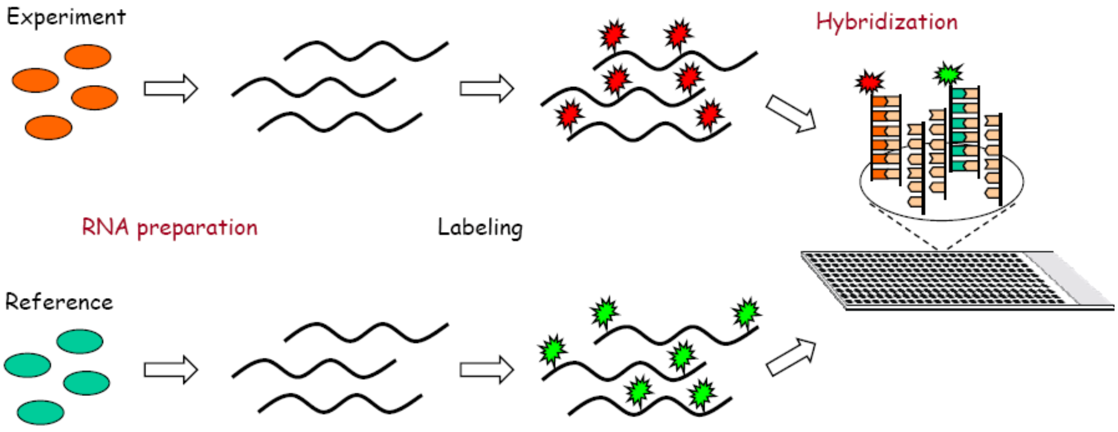


Figure 3.2: Simplified overview of a two color microarray experiment.

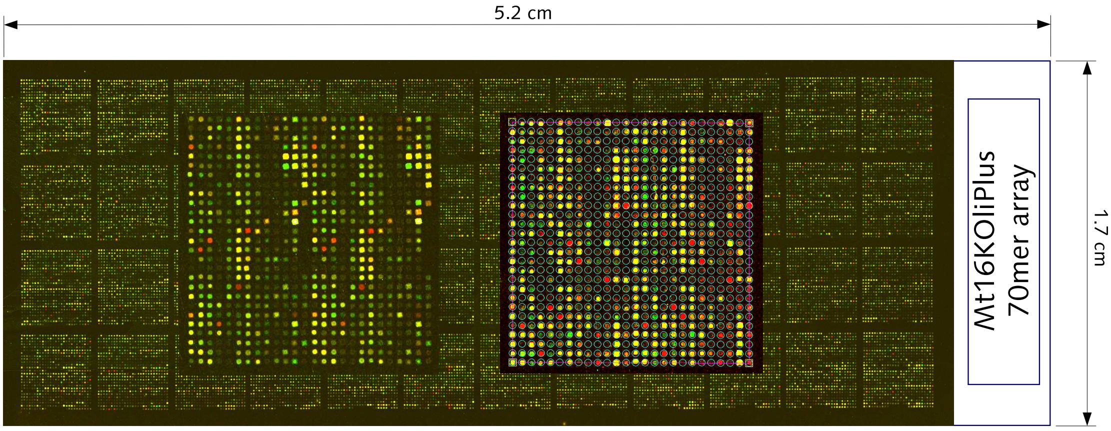


Figure 3.3: Enlarged false-color image of a spotted oligonucleotide microarray. In the center of the image, two enlarged views of a single grid are depicted. The right one depicts circles representing the results of an image segmentation (image provided by Helge Küster).

3.1.2 In-Situ Synthesized Microarrays

A photolithographic process has been developed by Fodor *et al.* (1993) which allows to synthesize oligonucleotides directly on a solid substrate. A first application of a microarray produced with *in-situ* synthesized oligonucleotides was presented by Lipshutz *et al.* (1995). The process has been made commercially available by Affymetrix Inc. under the trade mark 'GeneChip' for a variety of model organisms including *Homo sapiens*, *Rattus norvegicus*, *Mus musculus*, *Caenorhabditis elegans*, *Saccharomyces cerevisiae*, *Escherichia coli*, *Arabidopsis thaliana* and *Oryza sativa*.

The production of GeneChips is very different from spotted microarrays. While both share the use of glass as the solid substrate, the synthesis of oligonucleotides with typical lengths of 25 base pairs is performed in a light-directed process consisting of multiple repetitive steps. The substrate material of the chip is coated with covalent linker molecules, which a covered by photo-labile molecules. UV light is directed to specified locations on the array by a specific mask and the photo-labile molecules are removed. The linker molecules exposed to light emission are thereby activated. Nucleotides of the same type are then added and bind to the activated linker molecules. This step is repeated with another nucleotide and a further mask until all oligonucleotides have been synthesized with the desired sequences.

The production process requires the design of appropriate short oligonucleotide sequences to represent a gene or other target sequences and the respective masks to direct the synthesis.

NimbleGen Systems Inc. have developed the proprietary 'Maskless Array Synthesis' technology, as a light-directed synthesis that replaces the photolithographic masks by miniaturized mirrors which they call Digital Micromirror Device (DMD). This allows to produce customized microarrays which are also commercially sold by Affymetrix

Single short oligonucleotides might not always be very specific for a single gene and thus they are susceptible to cross-hybridization; a target sequence is represented by a set of different short oligonucleotides on an Affymetrix array. In addition to the oligonucleotides perfectly matching the target sequence (called PM probes), oligonucleotides with a missmatch base at the central position (called MM probes) are added. They allow to compare signal intensity derived from unspecific hybridization events, which are assumed to occur randomly, with the specific hybridization signal of a perfectly paired sequence.

The labeling and hybridization process is similar to the processes for spotted microarrays, except that GeneChips are used as a single-channel platform. Hybridizations and the image read out can be performed by using proprietary equipment only.

3.1.3 Applications of Microarrays for Genomic DNA Hybridizations

While microarrays have originally been developed to measure the abundance of mRNA, the microarrays contain mostly DNA nucleotides. It is therefore possible to hybridize DNA microarrays with genomic DNA instead of cDNA.

Applications of DNA hybridizations to microarrays include the detection of mutants or strains of an organism as well as the differentiation of species. The differentiation of microbial species is of special clinical interest. A recent example is given by Masson *et al.* (2006), who have employed spotted arrays printed on a plastic substrate to differentiate several species of *Helicobacter* and *Campylobacter*. Another interesting aspect of this approach is the use of a non-fluorescent detection system.

Detection methods for the DNA of pathogenic viruses in host cells is another application of the DNA hybridization method with clinical relevance. Long *et al.* (2004) have developed a microarray design for genotyping several point-mutations in the SARS coronavirus. The authors note that this array can also be used for detection and genotyping of the virus. Other applications of DNA hybridizations include the exploration of genetic diversity in a population (see for example Kidgell and Winzeler, 2005).

Moreover, DNA microarrays have a particular potential in the evolving field of metagenomics (Sebat *et al.*, 2003; Handelsman, 2004), aiming at the direct application of genomic analysis to environmental samples and environmental species communities (Gentry *et al.*, 2006). Applications include whole-genome microarrays developed for detection and identification of bacterial species from environmental samples. Such microarrays contain the whole genomic DNA of one species per spot.

3.1.4 Tiling Arrays

Conventional expression analysis microarrays contain sequences representing mostly predicted protein coding sequences. These predictions are based on algorithms that use intrinsic features of the DNA sequence, such as nucleotide frequency, and extrinsic features, for example sequence similarity with known coding sequences of other organisms.

Predictions of coding sequences are not completely reliable, in particular for eukaryotic genomes. Locating the exact position of the transcription start site and intron-exon boundaries is a not fully solved task. Hence, it can be concluded that conventional gene expression microarray designs are biased towards the results of gene prediction methods, since other transcribed regions which have been regarded as intergenic might exist.

To be able to detect transcripts from the whole genomic sequence of an organism, tiling microarrays have been developed (Bertone *et al.*, 2004). They cover a whole genome or parts of a genome using oligonucleotides of a fixed length having constant length gaps between them. Previously unknown transcripts, transcription start sites and intron-exon boundaries can thereby be detected with a precision up to the length of a single oligonucleotide. Tiling microarrays are produced using in-situ synthesized oligos due to the higher density this technique allows.

Oligonucleotides for tiling arrays need to cover the complete genomic sequence; therefore they cannot be optimized with respect to low cross-hybridization and equal melting temperature. As a result a higher variation of signal intensity between oligonucleotides is likely to occur. This problem has to be dealt with in subsequent data analysis steps.

3.1.5 Protein Arrays

Microarray production technology, originally developed to deposit DNA reporters, has been further adopted to small peptides, proteins, and antibodies. Protein microarrays can be produced by either spotting purified solutions of molecules or alternatively by light-directed synthesis of short peptides on the surface.

Spotted antibody arrays are mainly used for the detection and quantification of proteins and protein abundance in a complex mixture. New approaches allow the deposition of full-length functional proteins. This type of arrays is used to study protein–protein, protein–DNA and protein–small molecule interactions (Bertone and M, 2005; Doi *et al.*, 2002; Zhu *et al.*, 2001). Protein microarray experiments currently use single dye techniques.

3.2 A Sample Analysis Pipeline

Although there can be many different ways in which to perform an actual microarray experiment, on a more abstract level, several steps of analysis are common to every such experiment. A fixed series of operations or experimental steps has to be followed, which is imposed by the technological requirements. Some steps are not specific to microarrays but to biological experiments in general.

A rough categorization is sometimes based on laboratory versus computer based analysis task; but it does not seem too helpful because it is an artificial separations by the location where the analysis steps are performed. It is better to devide experiments into three logical stages:

A planing and design phase during which the experimenter defines initial hypothesis, consults the literature, defines which effects to study, which variables or quantities to measure, and which instruments to use.

The experiment conduction wherein the object of study is eventually exposed to experimental conditions and quantities of interest are measured.

The analysis phase in the course of which measurement results are evaluated and interpreted to achieve results.

These abstract definitions can be further subdivided, as there can be various levels of complexity depending on the type of experiment; and for microarrays many of these steps involve complex biochemical reactions. The following description of a microarray experiment pipeline is in accordance with the steps described in several textbooks on microarray analysis, although none of them has explicitly defined a workflow diagram (Baldi and Hatfield, 2002; Kohane *et al.*, 2003; Parmigiani *et al.*, 2003; Schena, 2003).

The planning and design phase is most important for a successful experiment. Planning might eventually give rise to the question whether to carry out this experiment at all, as possibly similar measurements have already been made. Moreover, it is necessary to identify free and dependent variables in the experiment, and, most importantly, the initial experimental hypothesis or experimental question should be stated[1]. It will be further assumed that the choice of measurement techniques has been made and includes microarrays.

The experimenters need to select the appropriate microarray technology and maybe even produce their own arrays. Also, the necessary number of replications of measurements need to be assessed to be able to measure expression at the desired level of confidence. Power analyses methods serve to calculate the approximate number of replications necessary. They can help in experiment design by providing an estimate of the number of replicates, given the desired power (the ability to detect a large proportion of the differentially expressed genes), the confidence level, and the variability of the data (see for example Pan *et al.* (2002), Black and Doerge (2002), Li *et al.* (2005), Page *et al.* (2006), and in particular Fu and Jansen (2006) for a software implementation of power analysis based on publicly available data).

Efficient experiment design needs to consder which conditions to compare directly if using a multi-channel platform. A good assignment of experimental factors should result in high experimental power, while minimizing the number of required microarrays and thus the costs. Many methods to find good or even optimal designs have been proposed in the works of Kerr and Churchill (2001), Kerr (2003), Churchill (2002), Yang and Speed (2002), Fu and Jansen (2006) and many more.

The phase of experiment conduction can be further divided into the generation of the actual biological sample and the measurement process. Sample generations is the process of studying an organism, a cell or parts thereof or even a community of organisms, while controlling the free experimental variables. Biological molecules of interest (RNA, DNA, protein) are consecutively extracted. Sample generation is the most variable part of the experiment. Sample measurement is the next step, that involves labeling and hybridization as complex biochemical reactions.

Further data acquisition and processing involves computer hardware and software and is still part of the measurement process. Images of the arrays are acquired. In a first step of processing, image analysis algorithms are applied to reduce pixel related image data to intensity statistics related to each feature on the array.

[1] Often this is as simple as trying find all the genes differentially expressed under condition X vs. condition Y.

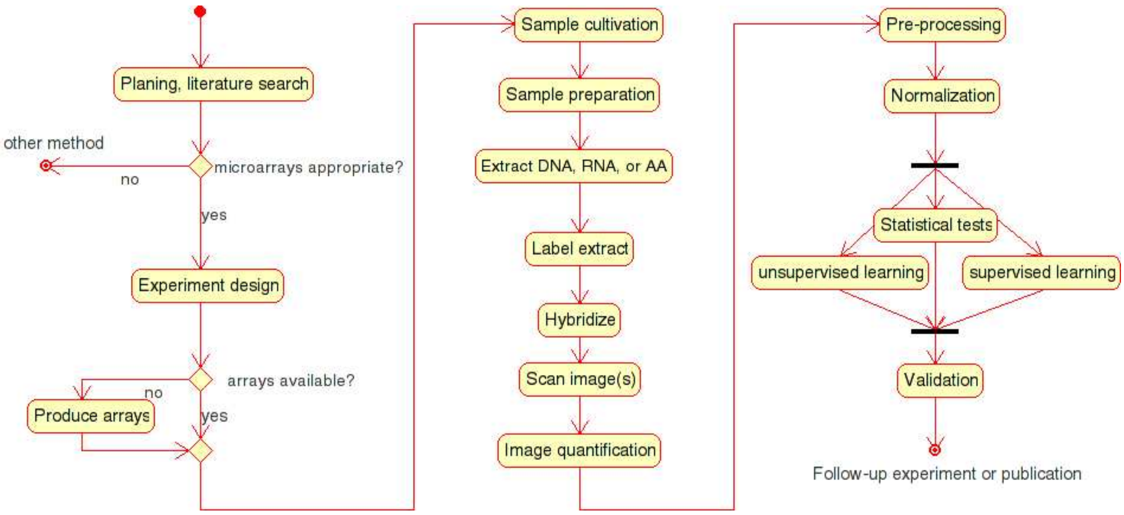


Figure 3.4: Idealized consecutive analysis steps in a microarray experiment as an activity diagram. The left part of the workflow represents the planning and preparation of the experiment, the central part depicts the actual conduction of the experiment, whereas the right part constitutes the analysis an interpretation of the measured data.

Intensity data can be filtered to remove measurement artifacts and need to be normalized to make replicate data comparable. Pre-processing can be seen as an intermediate step between biological measurement and interpretation.

The data analysis and interpretation step is again variable, but it almost always involves statistical inference to identify regulated genes. While some experiments might end here, others may involve machine learning techniques.

Supervised learning or classification methods allow to assign genes or cases into classes of objects, for example cancer types. Unsupervised methods can be used to seek for structure in the data or to identify groups of genes, tissues and patients. It is nor possible to predefine the exact workflow of the data analysis section as this greatly depends on the experimental question, prior assumptions or knowledge about the data.

The last step is usually the validation of the acquired results with the help of other measurement techniques and the generation of new hypotheses, which eventually give rise to new experiments. The whole work flow is summarized in Figure 3.4.

3.3 Properties Of Microarray Data

3.3.1 Variation and Replication

Microarrays have been proposed as a technique to measure transcript abundances. As a large-scale measurement technique, microarrays will almost certainly not be able to deliver exact quantifications of transcript abundances, but will be prone to some measurement error. Since its introduction, microarray technology has undergone many advances. These advances have been made in the different technological platforms, as well as in the available statistical data analysis methods, where some have been designed specifically for microarrays. “However”, as Shields (2006) puts it in his Editorial text – titled “MIAME[2], we have a problem” – in *Trends in Genetics*, “no amount of statistical or algorithmic knowledge can compensate for deficiencies in the technology itself”.

For this reason and for a well founded research, it is necessary to perform studies about the reliability and reproducibility of any method. Microarrays, in particular due to the complex laboratory work-flows involved, suffer from a variety of influences, that may result in a multitude of deviations. This is particularly important, as many researchers prospect the use of microarrays for drug discovery or even as a diagnostic tool in clinical disease classification (see for example Wang *et al.*, 2005; van de Vijver, 2005; van de Vijver *et al.*, 2002).

Unfortunately, the independent validation of microarray data is hampered by the lack of standards to compare its results with. For studies concerned with clinical prediction of malignancies, independent validation of their results should be possible in principle, because samples can be labeled by human experts and results from studies on a similar topic can be compared. The importance of the validation of studies has often been stressed, but “Validation is still an analysis and can be manipulated as can any analysis” (Ioannidis, 2005).

Ntzani and Ioannidis (2003) have carried out an evaluation study on previously published prediction studies of clinical diagnosis based on microarrays. Only a minority of these studies were found to comprise sufficient validation of the findings. Michiels and coworkers performed a re-analysis of seven large studies predicting prognosis of cancer patients based on microarray data (Michiels *et al.*, 2005). It was found, that the list of predictor genes was highly inconsistent between studies and that five out of seven studies did not perform better than random predictions.

In fact, the above mentioned methods make far-reaching assumptions about the data, in particular, that the expression profiles of the tissues contain information about cancer types or prognosis and that these changes in expression profiles are predominant over changes from other sources. This is not necessarily the case and is also hard to validate with the rather small number of cases selected for the classification studies. Hence, the variability of the resulting classifications might be caused by the statistically unsound foundation of the experiments and not by the technology.

In contrast to these rather frustrating findings, there are also more positive results when the measured quantitative value of mRNA is directly controlled by comparison with other methods, not indirectly by making strong (possibly unjustified) assumptions. The Microarray Data Quality Control (MAQC) project is concerned with reproducibility and quality assessment of microarray data. Within this project, measurements from several microarray platforms have been compared with each other and with other measurement techniques such a quantitative RT-PCR. MAQC

[2] Minimum Information About a Microarray Experiment (MIAME), see Section 4.1.1

members have compared the quantitative measurements of RNA levels and have found that the correlation between different measurement techniques was generally high (Patterson *et al.*, 2006; Consortium *et al.*, 2006; Canales *et al.*, 2006).

Despite these findings, microarray studies have been criticized for exhibiting variation and beeing hard to reproduce. This immediately gives rise to the question, which reasons for variability can be identified. It can be partially answered by the fact that all measurement techniques, not only microarrays, are subject to a certain level of measurement error. Variation can be classified by its origin, and it is common to differentiate technical and biological variation.

This differentiation might seems a bit arbitrary when looking at data from a real experiment, because variation observed in an experiment cannot be directly assigned to its source. The usual strategy to assess the level of technical and biological variation is to perform the measurements repeatedly, yielding replicated measurements, so called replicates. It can be deduced from statistical inference theory that the higher the variability of the data, the more replicates are required to achieve a significant result. On the other hand this implies experimenters can respond to higher variability by simply adding more replications. In any case, to have a closer look on the sources of variation can be helpful to reduce them in further analysis steps.

Technical Variation Deviations in the technical process can have a large influence on the results of microarray experiments. The extend of this influences depend on the applied technology and microarray platform and can be observed in replicated experiments in the same laboratory and also between laboratories. Primary causes are variations in the applications of protocols. Further influences stem from the microarray production process such as variation of feature sizes and concentrations (Bammler *et al.*, 2005).

Some studies have also shown the large impact of differential reporter sequences as a source of cross-platform variation. Other technical problems include scanner settings, as well as image segmentation and quantification (Repsilber and Ziegler, 2005; Yauk *et al.*, 2004, 2005).

Failed PCR reactions, contamination of spotting solution with other DNA, or even false reporters from wrongly assembled microtiter plates can be sources of errors, which often cause constant deviations and are sometimes hard to detect. In microarray experiments, technical variation can be assessed by using technical replicates. To achieve this, experimenters can use the same sample and perform the whole process of a microarray measurement. Material stemming from the same sample can then be hybridized to multiple microarrays of the same or of different platforms. Also, microarrays can carry replicate spots of the same reporter or different reporter sequences but for the same genes. Replicate spots can also be interpreted as technical replicates, but intra-array variation could underestimate global technical variability. A large number of technical replicates has been recommended for quality control studies or evaluation studies of technologies (Allison *et al.*, 2006).

In principle, technical variation can be reduced by conducting the experiment carefully, following standardized protocols rigidly and by improvements in the measurement techniques. Quantitative RT-PCR and other methods show less technical variation while lacking the high-throughput abilities of microarrays. These methods are often used to verify microarray data from few genes.

Biological Variation Assume that all kind of technical variation during measurement of a quantity could be reduced to zero, which is of course infeasible, there can still be variation in the measured quantities. This is in fact true for the process of gene transcription. Many authors have described the process of gene expression as a stochastic one; meaning that identical cells exposed to identical conditions will exhibit large fluctuations in gene expression. The effects of natural fluctuation in regulation is not interpreted as a general disadvantage, but seem to be beneficial or even necessary for the cell (see for example Rao *et al.* (2002); Blake *et al.* (2003); Kaern *et al.* (2005)).

As a consequence, there can exist no 'true' measurement of transcript abundance, even under the most rigid experimental control. Further influences can be associated with measurements of multiple cells. The amount of RNA extracted from a single cell is insufficient, so that cell cultures or tissue sample must be used. Not all cells in the sample are necessarily genetically identical, they can even belong to different cell lines or strains. Synchronization of cells with respect to their growth state is also an issue; cells of identical type can be in very different developmental stages. This is especially an issue when experiments are repeated with different organisms or patients (Baldi and Hatfield, 2002).

The level of biological variation is assessed by performing the experiment multiple times under the same conditions and by harvesting samples repeatedly. In general, the biological variation seems to be a bigger issue than the technical variation for assessment of significant results. Some publications recommend to prefer biological replicates over technical replicates in many cases (Allison *et al.*, 2006). Biological replicates will contain influence from both technical and biological variation, thereby serving to assess the overall variability of the experiment. If the number of replications exceeds the number of available microarrays, pooling is often used to generate a mixture of samples.

It is important to note that every sufficient measurement pipeline, not only those using microarrays, will exhibit biological variation. Any reasonable analysis should therefore contain biological replicates. To stress the effects of variation in further studies seems to be an important issue.

3.3.2 Preprocessing and Normalization

During the image analysis process a number of files containing measured data (in addition to some header information), which typically consist of large matrices having a row for each feature on the array and many columns with measurements and statistics for each spot. The ImaGene 6.0 software produces one quantification file for each channel of the microarray containing a configurable number of statistical measurements for each feature. GenePix on the other hand produce a single combined intensity file containing statistics for each channel, a do most other programs. The Affymetrix software produces so called 'CEL' files which contain measurements for all probes on the array.

To answer an experimental question, these values need to be condensed to a single value or at least a lower number of values describing the intensity or the differential intensity of a spot. It is not always obvious which one out of the many different statistics derived by quantification software is a representative value for the real spot intensity.

All softwares share the concept of estimating a foreground intensity and background intensity. Additional information is provided about the empirical standard deviation, spot shape and position. Most software also shares the concept of flags to provide additional information about the quality of a spot. Flags can be used to exclude the corresponding measurements from further analysis. There is no consensus on an optimal flagging strategy and how to treat estimated low-quality measurements.

Another open question is the merit of background correction. Background intensities are defined as unspecific measurement intensity, that can be observed, for example on a spotted microarray, in the proximity of a feature, where no DNA was found. Background correction is based on the assumption that the measured signal consists of the sum of the foreground signal and an unspecific signal of the microarray surface. Background correction is often carried out by subtracting local background estimates from the intensity observed in the feature area. Some authors note that it might be a good strategy to remove non-systematic error coming from background fluorescence (Chen *et al.*, 1997; Quackenbush, 2002), others have observed an increase in variance by background subtraction or have proposed different models for background estimation and correction (Yang *et al.*, 2001; Attoor *et al.*, 2004; Yin *et al.*, 2005).

Ratio computation is the next step in the data reduction process of microarray analysis. Suppose we have a microarray with a number of destinct features. We compare a measurement R of a treatment condition against a measurement G of a control condition. A measure of differential expression for the i-th gene could be given by the ratio T_i which can be written as

$$T_i = \frac{R_i}{G_i}$$

For spotted two-color microarrays it is necessary to compute ratios between two conditions. Microarray features may suffer from a variety of technical influences. The volume spotted on the surface as well as the concentration of the DNA within the solution is highly variable between individual features. As a result the single channel signals are highly variable, but for a comparison of two conditions both competitive hybridization signals should be affected by spot intensity variation by

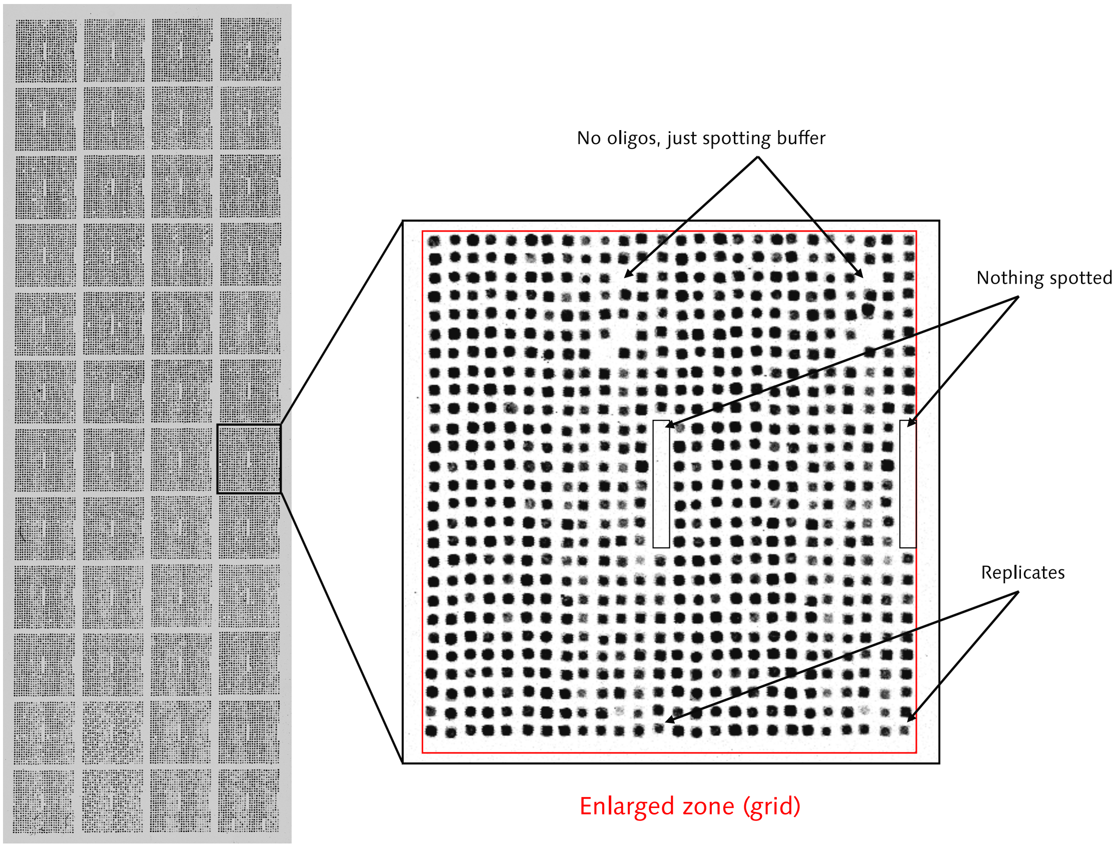


Figure 3.5: A test microarray hybridized with a solution of short labeled random sequences, which can be expected to almost equally bind to each reporter molecule on the surface, yielding a constant signal for each feature. Here, the variation in spot size and concentration is clearly visible (image provided by Helge Küster).

the same amount of variation. Figure 3.5 gives a good impression of single channel signal variation.

Single channel platforms such as Affymetrix arrays also allow for ratio computation between individual arrays. Ratio computation is necessary for single channels platforms, too, to be able to get a measure of differential expression between arrays. In contrast to two-channel methods, ration computation can be done between all arrays in the experiment. In general, single channel measurements require to have much less variation between features, because these cannot be compensated by within-array ratios.

It is quite common to apply a logarithmic transformation to the data. Li and Wong (2001) and Sásik *et al.* (2002) propose multiplicative noise models for microarray raw-data, which make the log-transformation favorable, as this leads to an additive error model for the transformed data. Multiplicative noise means the impact of noise on the signal increases with the intensity of the signal; additive noise models result in an constant influence of error on the signal and can in theory result in normally distributed data.

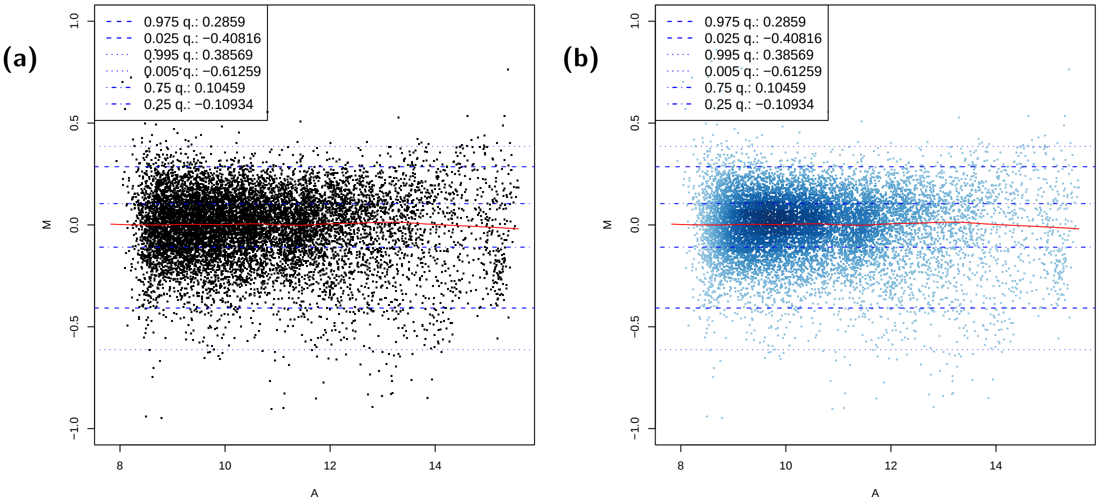


Figure 3.6: Two types of MA scatterplot depicting the global lowess normalized log-ratios of a single two-color microarray. This is a self-self hybridization experiment using the same RNA for both channels (data provided by Andrea Hüser), which should not exhibit differential expression. All variation in the images is a result of technical variation. The red line at the center is a lowess scatterplot smoothing line, the dashed line pairs denote the regions that contain 50% (inner pair), 95% (middle), and 99% (outer pair) of all data. Technical variation can be minimal in a well conducted setting, as 99% of the data is contained within $M \in [-0.61, 0.39]$. Plot **(a)** is a standard plot showing a black dot for each individual measurement, type **(b)** is a density scatterplot, depicting each region by a color proportional to the density of spots in the region. Regions of higher density are denoted by darker shades of blue.

Another, convenient advantage of a logarithmic ratio is that it makes up- and down-regulation comparable. Concerning ratios, a two-fold up-regulation corresponds to $T_i = 2$, a two-fold downregulation corresponds to $T_i = 0.5$, for a $\log_2$ transformation we get values of 1 and -1 respectively. As can be observed here, the log-transformation yields a symmetric range of values (Chen *et al.*, 1997; Quackenbush, 2002).

The intent of applying normalization to microarray data is to make the data from different microarrays within an experiment comparable. Therefore, it is necessary to remove systematic bias from the datasets (Quackenbush, 2002). A systematic bias in the data might originate from differences in RNA-concentrations between samples, differences in scanner settings, and differences in the labeling, bleaching, and detection behavior of the fluorophores. From inspection of technical replicate arrays hybridized with the same labeled extract, it can be concluded that scanner settings contribute a large portion to the between-arrays bias.

Other authors term these influences non-biological variation (Lu *et al.*, 2005). This term seems a bit misleading; it could also encompass stochastic variation (see Section 3.3.1), which cannot be approximated by a function and thus is not resolved by normalization. The causes of systematic variation are not evident from the data, and some authors claim that the systematic effects might in fact have a biological meaning (Finkelstein *et al.*, 2002).

Smyth and Speed (2003) propose a logarithmic transformation of the raw microarray intensities that is suitable for plotting differential expression on a log-scale and for the purpose of normalization of cDNA microarrays. This transformation maps the optionally background subtracted intensity values to a log-ratio (M) and log-intensity measure (A):

$$M_i = \log_2\left(\frac{R_i}{G_i}\right) = \log_2(R_i) - \log_2(G_i) \tag{3.1}$$

and

$$A_i = \log_2\sqrt{R_i G_i} = 1/2(\log_2(R_i) + \log_2(G_i)), \tag{3.2}$$

where R_i and G_i denote the intensity measure from the image analysis software with or without background correction for the ith feature. In a so-called MA-plot, the data can be depicted and systematic variation is made visible. By mapping the local density of the distribution on the colors of the plot, more information on the number of measurements in a region can be made visible. This avoids the effects creating black clouds by overplotting (see Figure 3.6 on the preceding page for a standard MA-plot and a density MA-plot). Systematic bias in a microarray experiment performed at the CeBiTec is depicted in Figure 3.7 on the next page.

A large number of methods for normalization of spotted microarray data has been developed since their introduction. Simple scaling methods adjust the data to have unit mean or median (see for example Chen *et al.*, 1997). The empirical distribution of the data is also scaled to have unit variance or Median Absolute Deviation (MAD) (Dudoit *et al.*, 2002). The original Affymetrix normalization method is a simple scaling method using a reference or 'baseline' array. In order to have the same mean intensity as the baseline array, all other array intensities are scaled with a constant factor,.

Dudoit *et al.* (2002) and Yang *et al.* (2002) have observed non-linear biases in MA-plots and describe a normalization method for correcting them. This method is based on the LOWESS algorithm, which is a robust scatterplot smoother using weighted local linear regression (Cleveland and Devlin, 1988). This approach assumes that the log-ratio M is a function of the logarithmic spot intensity A:

$$M_i = \log_2(R_i) - \log_2(G_i) - \mathrm{c}_D(A_i), \tag{3.3}$$

where c_D denotes a function on the log-intensity, estimated by regression from the empirical data distribution D.

Lowess normalization has been adopted for single-channel microarrays. As single-channel measurements do not allow for direct ratio computation, Bolstad *et al.* (2003) propose to apply a pairwise comparison of the intensities from all arrays to gain ratio-estimates and then correct the pairwise M-values by a lowess transformation. This method is termed 'cyclic-loess'. A disadvantage of this setting is that all possible pairwise comparisons have to be carried out, leading to quadratic complexity in the number of arrays.

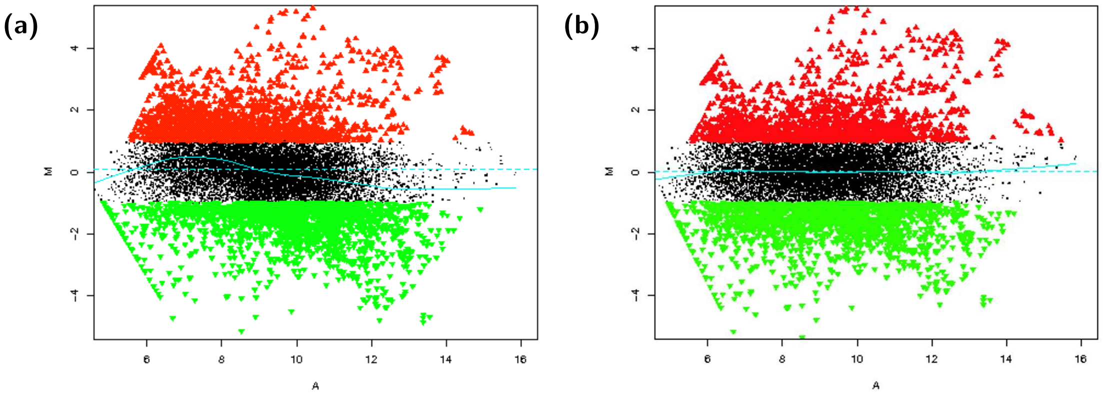


Figure 3.7: Systematic but non-linear bias **(a)** in an MA-scatterplot and its correction by using lowess-normalization **(b)**. The bias in **(a)** is made visible by the blue central lowess curve.

In the same publication, Bolstad *et al.* propose quantile normalization as another technique that is mainly intended for single-channel microarrays. The principle of the quantile normalization approach is to transform the empirical distributions of all microarrays in an experiment to a common empirical distribution. After the transformation, each distribution becomes a permutation sampled from the common distribution. This technique is suitable only for experiments comprising many arrays. For dual-channel microarrays several possibilities exist in how to use quantile normalization. It can be applied to data from each channel separately, to all channels together, or to the logarithmic transformation of the ratios. The latter might be preferable given further computation is based on log-ratios, because single channel intensities from a common distribution do in general not result in a common ratio distribution due to permutations of the single-channel values.

Other authors also vote for a functional dependence of the measurement variances of measurement intensity. Huber and colleagues have proposed a model for equalization of the variances of the data (Huber *et al.*, 2002). They claim that the logarithmic transformation inflates variance for low intensities. After applying their method, termed 'variance stabilizing transformation' (vsn), intensity dependent bias of variance should be removed. Variance stabilizing transformations also imply the calculation of a new difference statistic Δh. In this approach the log transformation from equation 3.1 is replaced by the inverse sinh transformation. The actual functional form is:

$$\hat{h}_i(y_{ik}) = \operatorname{arcsinh}(a_i + b_i y_{ik}) \tag{3.4}$$

The difference statistics is calculated as for logarithmic transformations:

$$\Delta h_{k;ij} = \hat{h}_i(y_{ik}) - \hat{h}_j(y_{jk}) \tag{3.5}$$

Here, i and j denote different arrays and k denotes the k-th feature. y_{ik} denotes the (possibly background corrected) intensity measurement for array i and feature k. a_i and b_i are array specific arbitrary real-valued parameters.

An advantage of the vsn-transformation is its ability to cope with intensity values ≤ 0 which can result from background correction, while the logarithm of these values is simply undefined. The authors state that the vsn transformation converges to the logarithm for large intensities, and there is only a slight difference for low intensities. If this is method is advantageous over the logarithm is questionable. Even more, the stabilizing transformation can also be interpreted as an artificial reduction of the variability of low-intensity spots, which also could have negative effects on the false discovery rate of downstream analyses.

In addition, in common experiments there might be almost no impact of this method, because it is common practice to remove low intensity measurements from further analyses due to their inherent variability.

The vsn method introduces parameters into normalization, which have to be set manually or estimated from each dataset. This might make datasets less comparable, as all datasets will be treated differently. In addition, significant advantages of this method over logarithmic tranformation for consecutive analysis steps could not be identified from evaluation studies.

A crucial step for normalization is to select a subset of representative features on which to carry out an estimation of a possible bias. Most normalization methods rely on the assumption that a vast proportion of the represented sequences is not differentially expressed. While this assumption may be valid for whole-genome microarrays, it has been stated that this might not be the case for very extreme conditions and especially for thematic arrays containing a sub-set of genes of interest (Park *et al.*, 2003). Choosing a subset of genes based on subjective or objective criteria already implies the violation of the stationarity assumption. In fact, the opposite may be true; it can be concluded that most theme-specific arrays might have a bias towards differential expression.

The normalization procedures should therefore be carried out on a non-differentially expressed set of features of the microarray. If the number of features represents a large portion of the genome or is known to be relatively invariant, the whole dataset can be used to compute the normalization function. However, if this is not the case, representative elements with an expected unchanged expression have to be selected. So-called housekeeping genes are often expected to be invariantly expressed. But this seems to be a misconception, because there is strong evidence that an always unregulated gene which, could be used as an ubiquitous internal control, does not exist (Eickhoff *et al.*, 1999; Lee *et al.*, 2002).

Other techniques include the deposition of external control elements such as positive controls, which bind every spotted sequence and can thereby be assumed to reveal an approximately constant signal under each condition (Park *et al.*, 2004). But hybridization characteristics of positive controls could be completely divergent from the experimental features, and hence are less representative in their behavior than housekeeping genes.

Spike-in controls may be added to the labeled extracts in known ratios of concentrations. For ratios of one they should roughly yield equal signals for each condition. This method has the same limitation of being potentially non-representative as for the positive controls. In addition, the spike-in controls have been produced in a different process than the experimental sequences. Therefore, spike-in controls cannot compensate for differences in total RNA concentration between probes or the effects of different laboratory protocols.

Control-based normalization methods usually have to deal with the small range of total signal intensity covered. This hinders the application of intensity-dependent lowess-normalization on control-sets. Instead, global location statistics namely mean and median should be applied (see Kroll and Wölfl (2002) for an overview on global and control-based methods).

Several evaluation studies have been carried out, which try to asses the relative merits of the different normalization approaches. Park *et al.* (2003) have carried out a comparison of methods on spotted cDNA microarray-data where they find intensity dependent methods to be superior over global scaling, and that linear and non-linear methods perform equally well. Later, Qin *et al.* (2006) found that methods removing intensity and spatial biases had avantages over global and single-bias removal methods.

Wu *et al.* (2005) evaluated normalization methods on Affymetrix data. They measured the effects of different methods on the outcome of a classification problem. That way, they could not determine a single-best method, while they found indication that quantile normalization has slight advantages. This is in accordance with the findings of Bolstad *et al.* (2003) and Ballman *et al.* (2004), who favor non-linear methods such as cyclic-loess and quantile normalization over baseline scaling methods for Affymetrix arrays.

As a conclusion, no consensus could be reached so far on which normalization is preferable. This is reflected in the recommendations of the Microarray Gene Expression Data Society (MGED)[3] on the web-pages of the MGED transformation and normalization working group: "What method should I be using? This section will hopefully be filled in soon....". [4]

A more practical advise can be derived from experiences gained while working with normalization at the CeBiTec. Often, the use of global intensity dependent lowess normalization is a safe choice. It will not manipulate the data too much; if the data contain no non-linear shift, the lowess method will behave almost like a simple median normalization while providing good correction if a curvature is observed in the data.

The optimal choice of a method seems to depend on the nature of the data and especially on the microarray platform and design. Therefore, it is recommended to use normalization interactively with MA-plots and other visualization methods, and to compare the effects of different methods on the data. With improvements in measurement technology, the relevance of sophisticated normalization methods might become less important in the future.

[3] This organization is aimed at providing standards for methods and formats for microarray data. See Section 4.1.

[4] `http://genome-www5.stanford.edu/mged/normalization.html#which`

3.4 Inferring Significant Changes

The most basic question when carrying out a microarray experiment is which genes are significantly (differentially) expressed in a sample or a comparison of two sample. The inference step is of primary importance, as for many experiments it is the only relevant analysis step (previous data acquisition and processing steps can be seen as preparations to the inference step). Also for machine learning steps, inference statistics play an important role for data reduction; a filter step can remove gene-expression profiles that do not change significantly.

In the earliest microarray studies fixed cut-offs for ratios or log-ratios were used. The choice of fixed cut-offs is, however, arbitrary and was soon regarded as bad practice (Quackenbush, 2002). Such a so called fold-change approach fails to provide an estimate of measurement error. Without an estimate of random effects, it is impossible to assess the probability of observing an event (in this case a specific M-value) within a sample just by chance.

Statistical tests are therefore the primary statistical tools to deal with all sources of variation in a replicated microarray experiment (see Section 3.3.1). Due to the relevance of inference methods for subsequent analyses and the lack of comprehensive evaluation studies for statistical tests on microarray data, this field is covered in more detail within this work.

3.4.1 Statistical tests

Statistical tests stem from the field of inference statistics established by, among others, W.S. Gosset and Fisher in the early twentieth century. Statistical hypothesis tests share the same principle: A null hypothesis (H_0) is assumed to be true, until there is enough evidence to reject it. The opposite hypothesis is termed alternative hypothesis (H_1). A value s, called test statistic, is computed from sampled data to describe a feature of the empirical distribution of the data. A threshold (s_α) of the test statistic is computed for the rejection of the null hypothesis. s_α is set such that a sufficiently low probability of observing s_α or a more extreme value is achieved, given H_0 is true. To compute the probability of an observation, the distribution of the test statistic under the null hypothesis needs to be known. The acceptable probability to observe a (false) rejection of H_0 is designated α-value. Its choice is completely arbitrary, however traditionally, α-values of 0.05 or 0.01 are used. To denote the lowest possible α-value at which the null hypothesis can still be rejected, p-values have been introduced. A p-value can be interpreted as the probability of observing an at least equally extreme test-statistic.

The empirical distribution of the test statistics depends on the empirical distribution of the data. Therefore, most of the following examples rely on assumptions about the data, from which a theoretical distribution of the test static is derived. In contrast, non-parametric tests are not based on a certain type of distribution of the data.

3.4.2 Student's T-Test

The classical t-test is a commonly used method for testing if microarray-data show significant expression. The null hypothesis of the t-test, that the mean values of two samples $(\bar{x}_1, \bar{x}_2)$ are equal (or differ by a certain constant μ; for normalized and log-transformed microarray-data, it can be assumed that $\mu = 0$). For a single sample, the null hypothesis is $\bar{x}$ from μ (often, $\mu = 0$).

The alternative hypothesis can be either: the difference in means is greater than μ or less than μ, resulting in a one-sided test; or: the difference in means is not equal to μ, yielding a two-sided test.

The test choice of the statistic depends on whether there are one or two samples. For the one-sample case it is

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{n}}} \ , \tag{3.6}$$

where s^2 denotes an empirical variance estimate such as

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \ ,$$

and n is the sample size. For two samples it is either

$$t = \frac{\bar{x}_1 - \bar{x}_2 - \mu}{\sqrt{\frac{s^2}{n}}} \ , \tag{3.7}$$

when common variance between samples is assumed or

$$t = \frac{\bar{x}_1 - \bar{x}_2 - \mu}{\sqrt{\frac{s_1^2}{n_1}} + \sqrt{\frac{s_2^2}{n_2}}} \tag{3.8}$$

for unequal variances (s_1^2, s_2^2).

Given that the samples are normally distributed, the distribution of t follows a Student-t distribution, having $\nu = n - 1$ degrees of freedom in the one-sample case, with density function:

$$f_\nu(t) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{(\sqrt{\nu\pi}\ \Gamma(\frac{\nu}{2}))(1 + \frac{x^2}{\nu})^{\frac{\nu+1}{2}}} \ .$$

Two main problems when applying the t-test can be identified: The first one is the often small sample size in microarray experiments resulting in a probably unstable estimate of the sample variance. The second problem is to determine the distribution of the test-statistic under H_0. The underlying assumption of normality of the data may not be true in real data. Any deviation from the idealized distribution may lead to deviations in the calculation of p-values. Several approaches have been developed, some of them specifically for microarrays, that aim at solving at least one of these problems.

3.4.3 Wilcoxon's Rank-Sum Test

Wilcoxon's rank-sum test is a non-parametric test (Siegel, 1956); as such it does not rely on assumptions about the distribution of the data, which is a useful property for possibly non-normally distributed microarrays data.

The null hypothesis of Wilcoxon's method is that the median of the sample is equal to 0. This hypothesis is assessed by assuming that under the null hypothesis, the sample should be symmetrically balanced around 0, so that the number of observations with $x_i < 0$ is equal to the number of observations with $x_i > 0$.

To compute the test statistic, the absolute values of the observations are ranked. The sum of the ranks of the positive observations is computed:

$$W^+ = \sum_{i=1}^{n} \mathrm{rank}(x_i : x_i > 0) \ ,$$

where n is the size of the sample. A p-value for the probability of observing a rank-sum W^+ or a more extreme value, can be calculated by forming all permutations out of $1, \ldots, n$ and counting those permutations, yielding a value $S \geq W^+$. It can be inferred that a minimum of 6 observations is necessary to achieve a confidence level of 0.05; this property renders Wilcoxon's test inappropriate for small sample microarray experiments.

3.4.4 Bayesian Testing – CyberT

Baldi and Long (2001) address the problem of small sample sizes and thereby the poor estimate of sample variances . They introduce a Bayesian framework for estimating the variance, which is then introduced into the standard t-test, turning it into a regularized t-test.

Instead of calculating variance estimates directly from the data (sometimes termed a 'frequentist approach'), in a Bayesian approach, a model M is introduced that generates the observed data D. Following Bayes' law, it is possible to optimize the model parameters with respect to the maximal posterior probability $\mathrm{P}(M|D)$ of the model, given the data.

$$\mathrm{P}(M|D) = \mathrm{P}(D|M)\mathrm{P}(M)/\mathrm{P}(D)$$

or $\mathrm{P}(D)$ may be disregarded as it is independent of M,

$$\mathrm{P}(M|D) \propto \mathrm{P}(D|M)\mathrm{P}(M)$$

This leads to the task of selecting an appropriate *prior probability* $\mathrm{P}(M)$. Therefore, the model parameters are themselves modeled as being drawn from distributions with parameters, named *hyperparameters.* Baldi and Long assume a normal distribution (with parameters μ and σ^2) as a model for microarray data:

$$\mathrm{P}(D|M) = \mathrm{P}(D|\mu, \sigma^2) \approx \prod_{x \in D} \mathcal{N}(x; \mu, \sigma^2).$$

By choosing a normal distribution as the prior $\mathrm{P}(M) = \mathrm{P}(\mu, \sigma^2)$ (a so-called conjugate prior, which has the same functional form as the posterior model) and by setting $\mu = \bar{x}$, the authors achieve a regularized single-point estimate for the variance:

$$\sigma^2 = \frac{\nu_0 \sigma_0^2 + (n-1)s^2}{\nu_0 + n - 2} \ , \tag{3.9}$$

with σ_0^2 defined as a background variance and ν_0 as a weight parameter for σ_0^2 and s^2 as the empirical sample variance. σ^2 can then be used in the standard t-test formula as in Equations 3.6, 3.7 or 3.8, resulting in a regularized t-test.

The regularized test requires two parameters to be set: σ_0^2 and ν_0. In the CyberT implementation, ν_0 is set such that $n + \nu_0 = K$ where K is a constant that defaults to 10. The background variance is computed from a fixed number of other feature measurements from all microarrays. The default implementation uses a symmetric window of size w around the measured values with $w = 101$. The parameter setting is rather ad-hoc and is called 'rule of thumb' by the authors.

The CyberT method was evaluated on synthetic data using sample sizes between 2 and 5, with data drawn from two normal distributions. On this dataset, CyberT exhibits a decreased Type I error rate and increased power compared to the standard t-test. It is not clear whether such a positive result can be achieved on real data. While this test might have increased power, this is opposed by the necessity of setting arbitrary parameter values.

3.4.5 Linear Models and Empirical Bayes Methods – LIMMA

In 2004 Gordon Smyth published an interesting approach that combines an empirical Bayesian method with a moderated *t*-test and general linear models (Smyth, 2004, 2005). The method is implemented as an add-on library, for the statistical environment R.

General linear models, as they are used in LIMMA, have the advantage of allowing to reflect the experimental design including dye-swaps. The model is not restricted to a simple replicate design or two sample comparisons. A general linear model has the form

$$\mathbf{y_g} = \mathbf{b_g} + X\mathbf{a}_g$$

and is fitted to each gene. $\mathbf{y_g} = (y_{g1}, \ldots, y_{gn})^T$ denotes an n-dimensional response vector of log-ratios or intensity measurements from single channel microarrays; X is the experiment design matrix; $\mathbf{a}_g = (a_{g1}, \ldots, a_{gn})$ denotes the vector of regression coefficients, and $\mathbf{b_g}$ the intercept vector. Residual variance estimates of the model are used in the subsequent analysis.

3.4.6 Permutation Testing – SAM

Significance Analysis for Microarrays (SAM) is currently the most often cited method to determine significant regulation[5] (Tusher *et al.*, 2001). SAM has been developed to tackle the problem of the unknown distribution of the input data and also the problem of small sample sizes. One has to recall that the p-values arising from a t-test are valid only if the input data is normally distributed. The normality assumption could be violated in a microarray experiment, as the real data distribution is unknown.

Resampling is a technique to estimate an empirical distribution from the data. The idea behind resampling is to artificially extend the input data with a dataset showing no significant difference. To obtain such a dataset, SAM draws random samples from each group for each gene and re-assigns replicates randomly to group 1 and group 2 (for one-sample experiments, a random sample from the group is multiplied by -1). Under the assumption, that not many genes are differentially expressed, a dataset resembling an artificial background distribution of the test statistic can be achieved.

SAM uses a modified t-statistic of the form

$$b = \frac{\bar{x}_1 - \bar{x}_2}{s + s_0} \ ,$$

where $\bar{x}_1, \bar{x}_2$ denote the group means, s denotes the joint sample standard deviation of both samples, and s_0 is a small constant for stabilizing the standard deviation.

p-values are computed by calculating the b-statistics for the original data. After that, the resampling step is carried out and b-statistics for the resampled datasets are calculated. This is repeated and the background probabilities of observing an actual value of the b-statistic are estimated from the observed b-values. As the resampling process relies on random samples the p-values generated might not be fully self-consistent. This affects only the absolute p-values, but not their rank order, because the test statistic is independent of the permutation. This means that repeated applications of the algorithm to the same data might yield slightly different results.

3.4.7 Rank Products

Breitling *et al.* (2004) have proposed rank products to assess differential expression. The approach is based on the assumption that under the null hypothesis of no differential expression, the probability to observe a gene g at the highest ranking position with its expression measurement out of $n \in \mathbb{N}$ genes, for replicate i is $p_{1,g}^{up} = 1/n$. Given there are k such replicates and the events are independent, the probability is $1/n^k$. This can be generalized to all ranks r by defining the rank

[5] according to Google Scholar; this proposition needs to be interpreted with care, because the t-test is usually not cited

product:

$$\mathrm{RP}_{\mathrm{g}}^{up} = \prod_{i=1}^{k} r_{i,g}^{up}/n_i \ ,$$

where $r_{i,g}^{up}$ denotes the rank of gene g for its ith replicate in the list of all expression values in decreasing order. $\mathrm{RP}_{\mathrm{g}}^{down}$ is calculated from an ascending list of expression values. p-values are computed by resampling the association of genes and expression values and observing the probability of a certain RP-value.

The method is rather simple and straight-forward. It resembles closely the ranking of M-values. It does not take a measurement error into account, thus assuming equal variance for all genes. Also it assumes independence of replicate measurements. This is probably not true in a well conducted experiment, where replicated measurements should correlate well. Unlike the other methods, p-values also depend directly on the number of genes on microarray.

3.4.8 The VarMixt Approach

Delmar and colleagues have developed a novel approach for getting an improved estimate of the variance by the use of mixtures of distributions (Delmar *et al.*, 2005).

Given the set of genes $G = G_1, \ldots, G_n$ the gene-wise difference statistic Δ_g is assumed to be derived from a mixture of normal distributions:

$$\Delta_g \sim \mathcal{N}(\mu, \sigma)$$

They describe the real variance for each gene as the weighted sum of $k \geq 1$ variance classes $\mathbf{C} = \{C_1, \ldots C_k\}$ with variances $\sigma = \{\sigma_{C_1}, \ldots, \sigma_{C_k}\}$

$$Var(\delta_g|s) = \sum_{i=1}^{k} \sigma_{C_i}^2 \tau_{gi} \ ,$$

with $s^2 = s_g^2, \forall g \in G$ as the set of all empirical estimates of the gene-wise variances, and

$$\tau_{gi} = P(\sigma_g = \sigma_{C_i}|s)$$

as the conditional probability of observing the variance in variance class C_i of the variance model, given the observed empirical variances.

The variances are modeled as a weighted mixture of gamma distributions; the parameters of the mixture model are estimated from the observed variances using an expectations maximization (EM) approach. Instead of using the model estimate of variance in a regularized t-test, the mixture model is used to calculate $\Delta_g = \sum_{i=1}^{k} \mathcal{N}(\mu_g, \sigma_{C_i})\tau_{gi}$ as the test-statistic. The distribution of the statistic can therefore be directly inferred from this equation to calculate p-values.

The VarMixt approach requires to define a priori the number of variance classes for the given experiment. The authors state that this is not a trivial task, as there exists no exact estimate of the number of variance components in the model. On the other hand, Delmar and colleagues show that the algorithm is somewhat robust against the exact value of this number. They use the Bayesian Information Criterion (BIC) to select an appropriate number. In contrast to all other methods the variance estimates are not deterministic, because they are derived from a mixture model fitted by a non-deterministic EM-algorithm. This means multiple test runs on the same data could yield different results of the test statistic and the rank order of genes unlike any other test.

3.4.9 Tests for More than Two Groups

The methods described so far, assess whether one or two groups have location parameters significantly different from zero or from each other. If there are more than two groups of microarrays, for instance in a stress response analysis involving multiple stress conditions, one could use the two sample t-test to compare all groups. However, with many groups and different experimental factors, this is not efficient and also leads to cumulation of type I errors because of multiple testing. The Analysis of Variance (ANOVA) method provides a parametric extension of test procedures for this case. The null hypothesis is extended to $H_0 : \mu_1 = \mu_2 = \ldots = \mu_n$ for all group means μ_i for all groups. The rejection of H_0 is interpreted such that at least one group mean deviates significantly from the others; it does not tell which one or how many.

For multifactorial designs, an experimental setup having more than one free variable, ANOVA is superior to a naive t-test approach, because it allows the definition of an effects model. The effects model can resemble the experiment design and thereby capture the different sources of variation that can affect the data (Kerr and Churchill, 2001; Churchill, 2004).

3.4.10 Multiple Testing

There are ongoing discussions about whether or not microarray analysis is a multiple testing setup. Analyzing a microarray experiment involves the application of multiple tests, one for each gene. Doing a repeated experiment with repeated test procedures increases the probability of falsely rejecting the null hypothesis even if it is true in each case. If the significance level $\alpha = 0.05$, a commonly used value for many types of studies, is used, the test may yield 500 rejections from 10000 tests of the null hypothesis, even if we know the null-hypothesis is always true (Dudoit *et al.*, 2002).

Given a list of n p-values from multiple tests, Bonferroni's method is the simplest method to correct this: $p_{\text{bonferroni}} = p*n$. With Bonferroni's method the probability of a single false discovery in the multiple test experiment, the Family Wise Error Rate (FWER), is less than 5%. This is a very conservative method, as it is often tolerable, especially in microarray experiments, to allow a certain proportion of false discoveries to obtain a sufficient number of true discoveries. To address the problem of the reduced power of the Bonferroni correction, refined methods of controlling the FWER have been proposed (Holm, 1979; Hochberg, 1988; Hommel, 1988).

Other approaches, possibly better suited to the requirements of microarray experimenters, try to address the False Discovery Rate (FDR) that is defined as the proportion of false rejections within the number of all rejections (Benjamini and Hochberg, 1995; Reiner *et al.*, 2003; Liao *et al.*, 2004). A FDR value of 0.05 implies that a maximum of 5% false positives is observed among those genes called significant at this significance level.

3.5 Machine Learning Approaches

3.5.1 Cluster Analysis

Cluster analysis is a highly popular method to detect hidden structure in multivariate data, to generate hypothesis of co-regulation. Cluster analysis (or simply 'clustering') has become a very popular method also for the analysis of microarray data. Many authors have used a variety of different methods that were mostly developed for general multidimensional analyses.

The popularity of cluster-analysis is understandable, as it requires no or few prior hypotheses about the data. The application of cluster analysis is motivated by the 'guilt-by-association' assumption (see Section 2.3). If genes share a common mechanism of regulation (for example the same transcription factors) they could also be functionally related; accordingly, we might want to find groups of genes with similar expression profiles. This is, for example, true for operon structures in bacteria. All genes in an operon are transcribed in a polycistronic mRNA and should therefore have equal expression profiles.

The following principle is common to all clustering methods: Objects, numeric data vectors in the case of microarrays, are assigned predictor labels from a set of classes. The class assignment may be a hard assignment to a single class or a weighted gradual assignment to multiple classes. Therefore, the objects are compared by their pairwise similarity or dissimilarity and the algorithm is designed to optimize a certain criterion with respect to class assignment.

Given a defined optimization criterion (e.g. minimizing the quotient of distances within clusters by distances between clusters) and a distinct number of clusters, there is in principle a global optimal solution, but to find it requires to test all possible assignments. This is a combinatorial problem and hence computationally intractable except for very small problems. Given an optimization criterion, all algorithms for cluster analysis try to find a good, but possibly sub-optimal grouping of the data.

The definition of similarity or dissimilarity of the objects compared is essential for almost all clustering algorithms. The distance measures most commonly applied for interval scaled microarray data are based on general Minkowsky metrics. They compare two expression vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, and have the form:

$$d_p(\mathbf{x}, \mathbf{y}) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}, \ p \in \mathbb{N} \ ,$$

where $\mathbf{x}$ and $\mathbf{y}$ represent real-valued expression vectors of two genes. With $p = 2$, this is the usual Euclidean distance. Other distance measures use centered and uncentered pearson correlation ρ between continuous random variables $\mathbf{X}, \mathbf{Y}$ and their realizations $\mathbf{x} = (x_1, x_2, \ldots, x_n), \mathbf{y} = (y_1, y_2, \ldots, y_n)$.[6] The empirical coefficient of correlation is defined as:

$$\rho = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \ ,$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ denote the arithmetic means of $\mathbf{x}, \mathbf{y}$.

There are many different ways to find a good approximation to the optimal grouping of objects. The existing algorithms can be roughly characterized by assigning them to one of four groups:

Hierarchical methods construct a hierarchy of group relationships between individuals, which can be represented as a binary tree.

Partitioning methods assign each object to a distinct group label from a previously defined number of different labels. Together with hierarchical clustering, partitioning methods are most frequently used for microarrays. Both approaches are therefore treated in more detail below.

Probabilistic methods use assignments of objects to classes, weighted by some measure of probability. Often, the method attempts to find an optimal model of the distribution of the input data. A well-known approach of model-based cluster analysis is presented in the Mclust method developed by Yeung *et al.* (2001).

Node based methods assign objects to individual nodes which are inter-connected and thereby try to model the high-dimensional topology of the input space. The most well known method is the Self-Organization Map (SOM) by Kohonen (1995) with two initial applications to microarray data by Tamayo *et al.* (1999) and Törönen *et al.* (1999). The neural-gas algorithm of Martinetz *et al.* (1993) can be seen as an intermediate between a k-means approach and SOMs.

Network based methods construct a graph structure of nodes representing genes and edges representing interactions. Graph theoretic methods are used to

[6] The differences in definitions of vectors in $\mathbb{R}^l$ and realizations of continuous random variables are ignored for this purpose

find tight groups in the graph such as in the CLICK algorithm (Sharan *et al.*, 2003). Such networks could also contain cycles and genes can be connected to multiple other genes. Relevance networks are an interesting representative method of this category (Butte *et al.*, 2000). In a relevance network, edges between genes are constructed based on their pairwise correlations. Such networks may also contain cycles and genes can be connected to multiple other genes.

Hierarchical Methods

Hierarchical methods have the common property of constructing a hierarchy of groupings which can be represented as a binary tree. The hierarchical tree order of object groupings is also called an ultra-metric on the input-space. Hierarchical clustering operates on a distance matrix, such that the original data are not available in the algorithms. This provides the potential to use arbitrary measures of distances between objects. The hierarchy can be constructed by starting with each object in a separate cluster, joining two similar clusters in each step until only a single cluster is left over; this is called agglomerative hierarchical clustering. The opposite method is called divisive analysis; the algorithm starts with a single cluster containing all objects. Clusters are split in every step in a way that minimizes an error criterion for each split.

Hierarchical methods can be characterized as beeing greedy heuristics; an optimal cluster solution is approximated by repeatedly making a decision which achieves the maximal gain or minimal loss for the next step. The objective functions are defined in terms of an inter-cluster distance measure. This extends the notion of a distance between two vectors to a measure of distance between clusters. In agglomerative clustering, two clusters are selected for joining them, when their inter-cluster distance is minimal. Therefore, the inter-cluster distances are also called linkage methods.

Some popular linkage methods are:

Single linkage The distance between two clusters is defined as the distance between their closest members (Florek *et al.*, 1951; Sneath, 1957). Given a distance function $d(\mathbf{x}, \mathbf{y})$ and two Clusters C, C':

$$\mathcal{D}_{\text{single}}(C, C') := \min_{\mathbf{x} \in C, \mathbf{y} \in C'} d(\mathbf{x}, \mathbf{y}) \ .$$

Single linkage tends to form deep branching trees of weakly related neighbors. For this reason, it is not well suited for microarray data with lots of outliers.[7]

Complete linkage The opposite of single linkage is complete linkage. The inter-cluster distance is defined as the distance between the most distant members

[7] In bioinformatics, it is very useful for clustering sequence fragments (Expressed Sequence Tags, or ESTs) using string edit distance.

(McQuitty, 1957).

$$\mathcal{D}_{\text{complete}}(C, C') := \max_{\mathbf{x}\in C, \mathbf{y}\in C'} d(\mathbf{x}, \mathbf{y}) \ .$$

Average linkage (UPGMA) Unweighted Pair Group Method with Arithmetic Mean (UPGMA) is the average distance between all elements between both clusters (Sokal and Michener, 1958),

$$\mathcal{D}_{\text{average}}(C, C') := \frac{1}{|C||C'|} \sum_{\mathbf{x}\in C, \mathbf{y}\in C'} d(\mathbf{x}, \mathbf{y}) \ .$$

WPGMA Weighted Pair Group Method with Arithmetic Mean (WPGMA) differs slightly from UPGMA in computation of the linkage criterion; the inter-cluster distance is computed recursively using inter-cluster distances of previous joining steps. Let the new cluster $C = A \cup B$ be formed by joining clusters A and B, then the distance between clusters C and C' is computed as

$$\mathcal{D}_{\text{W}}(C, C') := \frac{\mathcal{D}_{\text{W}}(A, C') + \mathcal{D}_{\text{W}}(B, C')}{2} \ .$$

This approach differs from UPGMA in that the cluster size does not play a role, and that it is computationally less complex.

Centroid linkage The centroid method is defined as calculating the centroid (mean vector) for each cluster and defining inter-cluster as the distance between their centroids. It can be applied only, if the notion of a centroid is justified by the distance metric. Correlation coefficients are not suited, because they do not guarantee that a centroid vector exists.

$$\mathcal{D}_{\text{Centroid}}(C, C') := ||\bar{C} - \bar{C}'|| \ ,$$

where $\bar{C}, \bar{C}'$ are the cluster centroids: $\bar{C} = \frac{1}{|C|} \sum_{\mathbf{x}\in C} \mathbf{x}$

Ward's method Ward's minimum variance method also involves the cluster sizes into the inter-cluster distance (Ward, 1963):

$$\mathcal{D}_{\text{Ward}}(C, C') := \frac{2|C||C'|}{|C| + |C'|} (\bar{C} - \bar{C}')^2 \ .$$

Ward's method selects those clusters for joining which lead to a minimal increase in the overall error sum of squares. This method also tends to form more equally sized clusters if the error distribution is relatively constant for all data. The same limitation for the choice of a distance measure as for centroid linkage applies.

A widely used implementation of agglomerative clustering is the HCLUST Fortran routine in the STATLIB library described by Murtagh (1985). Another agglomerative algorithm is the AGNES routine implemented in Fortran by Kaufman and Rousseeuw (1990). DAISY is a divisive routine developed by the same authors.

The use of hierarchical clustering for the analysis of microarray data was popularized by Michael Eisen and colleagues, who have used a combination of agglomerative hierarchical clustering with pearson correlation distance and the UPGMA method (Eisen *et al.*, 1998). An appealing method for visualization of the results of the hierarchical clustering is also presented in this publication. The expression values are represented by color codes; a red–green representation is used which resembles the approximate colors of false-color microarray images. Negative log-ratios are projected on green values and positive on red values, yielding black for values close to zero change.

The expression matrix is then reordered according to the dendrogram and plotted as a heatmap beside the dendrogram. Due to the dendrogram ordering, it is easy to spot regions of similar expression profiles within the data by human inspection. The graphical display can also serve to estimate the approximate number of groups in the dataset. Often, the grouping of similar objects by dendrograms can be visually appealing; on the other hand there are other ways of ordering objects, which can be more appropriate. A good example of an alternative approach is the ordering of time-series data by the time of occurrence of the peak expression level. This approach has been used by Spellman *et al.* (1998) for depicting alterations of gene expression during the yeast cell cycle.

Partitioning Methods

Partitioning methods of cluster analysis assign a number of observations into a predefined number of distinct groups. Each object is assigned to exactly one cluster, therefore partitioning methods are also called ‘hard clustering’ methods. The k-means method constantly adjusts a fixed number of cluster centroids to the dataset until convergence (MacQueen, 1967). The algorithm requires a Euclidean distance to be able to compute a centroid that is a representative point having minimal distance to all points in a cluster.

Kaufman and Rousseeuw (1990) have proposed another approach which is based on cluster representatives from the set of data objects, called medoids. The algorithm is called Partitioning Around Medoids (PAM). The complexity for calculating the medoid from a matrix of distances is much higher (quadratic in each step with respect to the number of elements in the cluster) than the computation of a centroid directly from the data (which is linear). On the other hand the algorithm can use arbitrary distances. To be able to handle large numbers of objects, the authors have also developed a heuristic extension to PAM named Clustering Large Applications (CLARA). CLARA relies on a randomly sampled subset of points for each cluster to approximate a medoid. Kaufman and Rousseeuw have also developed a fuzzy k-means clustering algorithm (FANNY). Instead of making a hard assignment of each point to a cluster, FANNY uses a weighted assignment to cluster centroids.

3.5.2 Supervised Machine Learning

The machine learning methods presented so far are used to generate hypotheses about the structure of the observation space. As no feed-back from already existing knowledge is required, for example, to label observations as false or true or to assign an error quantity to the predictions, they are called unsupervised methods. If additional knowledge about class assignments can be provided, it is possible to use supervised methods for classification or regression.

Classification can be defined as a discrete prediction process, such that each data point is assigned to one out of n classes $\omega_1, \ldots, \omega_n$. The term regression is used for quantitative predictions delivering a numerical response. Most machine learning experiments with microarrays involve classification, as the experiments analyze discrete classes such as different tumor or tissue types, different species or strains. That way, mostly expression vectors of all genes of a full experiment are classified, but there are also few approaches that try to assign individual genes into classes, for example based on results from a preceding cluster analysis (e.g. Méndez *et al.*, 2002).

Classification methods (classifiers) have been developed for multivariate data. Learning is based on presenting labeled examples to the method in a training phase; after that, the trained classifier can be used on unlabeled novel data for classification. To validate the performance of the classifier, the available data with known class labels are divided into two disjoint sets, the training set and the validation set; using the validation set to assess its performance. To further increase the reliability beyond that of a single pass, a n-fold cross-validation approach is used. The training and validation process is repeated n times, while choosing n pairwise disjoint validation sets of size N/n (N is the size of the labeled data) to use in each run.

One of the most simple but highly useful methods is the k nearest-neighbor (kNN) classifier which makes class predictions for an unknow object based on the majority of the k closest labeled examples (Cover and Hart, 1967). Wu *et al.* (2005) have used a kNN classifier to compare the merit of different normalization methods; Golub *et al.* (1999) have developed a similar method based on representative expression profiles to classify two different types acute leukemia.

Other classifiers are based on decision functions that aim at optimization of class predictions. Among such methods optimizing a linear decision function, linear discriminate analysis (LDA) (Fisher, 1936; Ripley, 1996).

Vapnik (1998) has developed the Support Vector Machine (SVM) approach for classification. SVMs are suited for finding optimal separating functions in high-dimensional spaces and allow for linear and non-linear decision boundaries. Examples for applications of SVMs to microarray data are frequently found in the literature (see for example Liu *et al.*, 2005; Pavlidis *et al.*, 2002) and it seems that SVMs are a highly popular method for predictions.

3.5.3 Towards Systems Biology

Qualitative models of biological processes are well established, but the question has been raised how quantitative models can be carved out. These models should serve to develop and validate quantitative hypotheses on biological systems. To base model validations on a solid foundation, large datasets of quantitative measurements are required.

Systems biology is an emerging research field of biology, which is fostered by the availability of high-throughput technologies (Kitano, 2002b; O'Malley and Dupré, 2005). In contrast to classical molecular biology it tends to follow a more holistic approach: while in molecular biology, single molecules or genes and their interactions with a few other molecules are studied in a rather qualitative way, systems biology tries to achieve quantitative knowledge on the complete biological system (e.g. a cell) or many interacting components. To stress this contrast, classical molecular biology is sometimes termed a 'reductionist' approach by system biologists (Friboulet and Thomas, 2005).

Systems biology is an interdisciplinary approach grounded in systems theory (von Bertalanffy, 1968; Luhmann, 1994). Objects of study for systems biology are living systems, sub-systems, and components thereof, furthermore their interactions in a dynamic manner (Kitano, 2002a; Williamson, 2005). Other fields of particular interest are robustness of living systems against perturbations of essential biochemical parameters in the environmental conditions. Systems biology approaches have recently been applied to a manyfold of bacteria and archea (See for example Baker *et al.*, 2006; Wendisch *et al.*, 2006) and eukaryote organisms (Li *et al.*, 2006; Gutiérrez *et al.*, 2005), as well as in biomedical research (Mattoon *et al.*, 2005; Fraunholz, 2005, among many others), and the study of cell communities (Goryachev *et al.*, 2006; Grabe and Neuber, 2005).

Three central tasks can be identified for systems biology:

- generating hypotheses from experimental data,
- building quantitative models of the systems being studied,
- evaluating the models by comparing the model predictions with more measured data and improving it.

These steps are performed in an iterative manner. The benefits that can be taken from this iteration are an improved model, which could serve to make predictions about unobserved states of the system and, by building and improving the model, one might get a deeper insight into the function of the process.

A data-driven approach of iterative model improvement relies heavily on the availability of high-throughput data, in particular gene expression data from microarrays. Some approaches combine gene expression data with metabolome or proteome data or try to make inferences about regulatory binding sites by analyzing the promoter regions of genes (Baitaluk *et al.*, 2006; Downer *et al.*, 2005; Hayes *et al.*, 2005). Other groups use protein microarrays to elucidate protein-protein or protein-DNA interactions on a genomic scale (Mattoon *et al.*, 2005).

Datasets arising from these studies are highly complex and can be interpreted only in the light of the experimental context. Easy access to these datasets is essential for any subsequent analysis and modeling steps. Taking into account that many individuals from diverse scientific backgrounds are involved in the model building and data collection process, the most challenging task for systems biology might be to establish good collaborations between them and therefore to provide the technical foundation of collaboration (Williamson, 2005).

CHAPTER 4

State of the Art in Systems and Standards for Microarray Data

Software systems developed for managing and analysing microarray data have evolved concurrently with the maturing experimental technology. In the beginning, there have been only few software system tailored specifically for microarray data. Images and intensity measurements were stored in flat files of varying formats. Data were published as HTML-documents containing genes and expression values, sometimes accompanied by the raw data. Experimental protocols were provided as unstructured electronic documents or only within the publication of that experiment.

4.1 Standardization and Specification

In 2001, an editorial text in Bioinformatics depicted the importance of standardization as a consequence of the introduction of high throughput techniques into life sciences (Brazma *et al.*, 2001). A good example of bad practice is the problem of gene annotations which have developed historically. Gene names and annotations are often assigned inconsistently within genome annotations of different organisms and hinder easy retrieval of gene products or gene function and their comparison between organisms (Schulze-Kremer, 1997). In order to improve this situation, systematic classification schemata have been developed to make genome annotations more precise and consistent (for example: the Gene Ontology (Ashburner *et al.*, 2000) and the COG/KOG database (Tatusov *et al.*, 2000, 2003)). The strategy to introduce systematic nomenclature is also applicable to the annotation of microarray experiments.

In the context of microarray experiments, the notion of standardization refers to the introduction of standards into biological research for the representation and exchange of data resulting from such experiments as well as sufficient annotation about the experiment itself. In fact, the term *standard* is often used synonymously for *open standard*. Wikipedia[1] defines open standards as "...publicly available and implementable standards. By allowing anyone to obtain and implement the standard, they can increase compatibility between various hardware and software components, since anyone with the necessary technical know-how and resources can build products that work together with those of the other vendors that base their designs on the standard." The term is contrasted by proprietary standards which are not publicly documented and are exclusive to their owner or require expensive license agreements. Open formats constitute a subset of open standards, which are important for data-interchange between applications.

Standardization organizations play a key role in developing and disseminating open standards. A prominent example is the W3 consortium (W3C) which specifies standards for web technology[2][3]. The technical reports published on the web-site of the W3C are the primary source of documentation on these standards. The Bio Ontologies consortium is another example dedicated to standardization of representation and sharing of diverse biological knowledge.

The main rationales for introducing open standards for microarray data, mostly stated implicitly in the microarray literature, can be summarized as:

Interpretability As Stoeckert *et al.* (2002) point out, microarray data are highly context dependent; they can be interpreted only in the context of experimental conditions, parameters, and laboratory protocols. Standards should define the content of an appropriate annotation of such an experimental context enabling researchers to interpret or reproduce the experiment. Re-analysis of microarray experiments serves to assess data quality or to perform an independent evaluation of findings. Interpretability makes demands on the quality and content of experimental annotations.

Interoperability The development of public repositories has created the need to exchange experimental data and experimental annotations between heterogeneous software including local and public repositories. A common data representation makes data exchange between applications from different vendors possible. Standardization implies to define a document structure for experimental annotations and data.

Even formally defined standards of document structure leave room for individual

[1] `http://en.wikipedia.org/wiki/Open_standard`. The fact that Wikipedia follows an open concept of contribution lets it appear an ideal resource for such a definition.

[2] `http://www.w3.org`

[3] Although in the strict sense and as the W3C states these are only recommendations, they can be seen as *de facto* standards.

interpretation, as can be seen, for example, from the recommendations for web-technology. Individual interpretation can become an issue, in particular when the content of experimental annotation is concerned. As experimental settings and conditions cannot be known beforehand, new techniques or experiments not covered by the specification can arise. The following suggestions about the content of experimental annotations can be seen only as free-text guidelines for the required level of detail.

Nevertheless, a formal syntax or document structure to convey microarray related content is required for an appropriate experimental annotation. It has to be flexible, to be easily adapted to evolving research techniques and settings. On the other hand, the formalism needs to be precisely defined to be accessible for automated processing algorithms.

4.1.1 MIAME

Compared to sequence data, microarray data and their interpretation are much more complex in structure. A single genomic sequence per organism, strain, or individual is the result of a sequencing approach. The interpretation of the resulting single genome sequence does not (at least in current practice) depend very much on the applied sequencing technology and the genome is assumed to be invariant during the study.

In contrast to the invariant genomic sequence, arbitrarily many gene expression experiments can be performed. Brazma *et al.* (2001) state that there can be as many transcriptomes as there are cell types multiplied by environmental conditions. Thus, data are only valuable for independent research and evaluation when sufficient additional annotation about an experiment is provided.

For the first publications of microarray experiments, experimental data and annotations were made available to the public in a varying level of detail or even not at all. Additional experimental protocols were, at best, provided in free text form in the resulting publications. Good examples of how data were published as supplementary material are the yeast cell cycle experiments of Spellman *et al.* (1998) and the leukemia data of Golub *et al.* (1999). Raw data and images are provided on web pages as well as processed data and lists of gene annotations. The experimental setup is usually described in the Materials and Methods section.

Over the following years, several organizations and working groups created their own databases to store and publish microarray data, such as *ExpressDB* at Harvard University (Aach *et al.*, 2000) and *ArrayDB* at The National Human Genome Research Institute (Ermolaeva *et al.*, 1998). As a result, effective search and comparison of microarray experiments was difficult, as the data was spread over a growing number of databases and web sites in a variety of formats.

In 1999 the Microarray Gene Expression Data Society (MGED)[4] was founded by researchers from several different interest groups, for example sequence database

[4] `http://www.mged.org`

providers, research groups employing microarrays on a large scale, and commercial providers of microarrays, reagents, and related hardware and software. The primary interest of MGED is to define standards for communicating microarray experiments. The first major result of that effort was the publication of the MIAME (Minimum Information About a Microarray Experiment) recommendations (Brazma *et al.*, 2001). The intention behind MIAME is to provide a guideline on information to be recorded for interpreting of experimental results and performing independent verification. The MGED group also laid down a MIAME-checklist which can be used by authors and publishers of microarray related publications to check whether information is complete[5]. An open letter, sent to major journals by MGED members, suggested to require submission of MIAME-compliant data to a public repository prior to publication of microarray related articles. These recommendations were consecutively adopted by many journals, among which are the well-renowned medical journal *The Lancet*, as well as *Bioinformatics* and *Nucleic Acids Research.*

MIAME compliant annotation of microarray experiments can be divided into the following categories:

Array design information Each individual array used for the experiment has to be annotated, as well as its design. An array needs as a minimum a unique ID to reference it in the experiment table and the name of the array design used, as there can be many arrays with different designs within a single experiment. The array design describes the physical layout of a set of microarrays. For each design a name as well as contact information of the vendor are needed. Technological information includes the microarray platform, the type and origin of the reporter molecules, the surface type, the number of features and physical dimensions, production protocol and date. For every individual element on the microarray, physical location, sequence, and sequence type, as well as many other properties of the corresponding biomolecules have to be reported.

Experimental design The experiment can be seen as the main grouping unit for hybridizations. It groups all related hybridizations for the analysis of a scientific question. According to the MIAME-checklist, an experimental annotation should include contact information for the experimenter or lab, a short free-text description of the experiment and findings, bibliographic references and a description of the type of experiment using predefined terms such as 'normal vs. diseased comparison', 'time course' or 'dose response'. Experimental variables, defined as the quantities changing during the study, have to be specified. The grouping and connection of all hybridized arrays to the experimental variables and the corresponding extracts have to be explained. Furthermore, quality control steps such as technical and biological replicates have to be specified.

[5] `http://www.mged.org/Workgroups/MIAME/miame_checklist.html`

Samples and extract procedures A sample is the biological material from which RNA, DNA or proteins used in the experiment are extracted, for example a cell culture or tissue. Further, the organism, cell type, and tissue have to be recorded. The method of RNA, DNA, or protein extraction from the sample material has to be described in detail as well as the labeling procedure preceding a hybridization.

Hybridization information For each hybridization a detailed protocol description has to be provided. This protocol should describe reagents and quantities of labeled extract used during hybridization and washing, if applicable. Also, the instruments used for hybridization, for example hybridization chambers, have to be described.

Measured data and procedures The measured data from each scan of a hybridized microarray include the original images from the scanner and datafiles resulting from applying an image analysis software to each image. Due to the huge space requirements it is not clear if the original images have to be included in a MIAME-compliant submission. Additionally, hardware and software (e.g. microarray scanner, scanner software, and image quantification software) have to be described. This is especially important for the scanner settings. Datasets which summarize measurements of related spots (e.g. computing a mean value over replicates) which are used to gain the results of the study have to be provided together with a description of the method and parameters used for transformation.

Data transformation procedures As the choice of a normalization procedure may affect the outcome of the subsequent analysis, the normalization strategy, background correction and further transformations have to be specified; further, on which set of features the normalization method relies, e.g. housekeeping genes or all spots on the array. The normalization algorithm and parameters have to be documented. The MIAME specification also requires to state the control elements on the array and if external controls have been added to the labeled extract.

Although the MIAME-recommendations aim at standardizing the information content to describe a microarray experiment, it leaves space for subjective interpretation. For example, the provision of the original scanned images is optional due to the high storage requirements of image data, and resulting from that high costs for public repositories. On the other hand, re-evaluating the quality of image quantification and also assessing the quality of the signals is impossible without the original images. The use of freetext descriptions of experimental protocols, substances, and procedures can become another point of ambiguity. The level of detail may vary between submitters. In addition, the availability of the original reporter sequences of custom arrays might be an issue.

4.1.2 MAGE

Although it is possible in principle to include MIAME-compliant annotation in a free-text format, the need for a standardized format for reasons of interoperability between systems still persists. The development of a standardized format for microarray data was mainly driven by the need to send data to public repositories and to interchange data between institutions. A joint effort with the Object Management Group (OMG) [6] was undertaken to establish a standardized format to capture the content prescribed by MIAME. Independently Rosetta Inpharmatics developed the GEML (Gene Expression Markup Language) format and MGED developed the MAML language which can both be seen as forerunners of a common standard.

In 2002, version 1.0 of MAGE-ML was defined by an MGED working group and the standard was published by the OMG in a public document.[7] MAGE-ML is an XML application[8], as such MAGE-ML documents must be written using XML (eXtensible Markup Language)[9] markup. XML provides a structured way to encode documents. A markup language consists mainly of elements (sometimes called 'tags') which provide a means of structuring a document and allow automated processing of documents.

A certain document syntax can be defined using Document Type Definition (DTD) documents or an XML-schema, which provides a more flexible way of describing valid XML syntax. The syntactical validity of a XML, and in particular of MAGE-ML documents, can be automatically verified by an XML-parser.

Unlike GEML and MAML, MAGE-ML was not developed by simply defining a DTD from scratch. The design of MAGE-ML followed an object oriented approach. An object model, named MAGE-OM, was specified using the Unified Modeling Language (UML). Using UML, class diagrams of the structure and interconnection of language elements can be provided. UML-based models can be specified as eXtensible Metadata Interchange format (XMI), which is also an XML application.

XMI allows for automated parsing, processing and conversion of the model. This way, the object model was transformed into a DTD by a set of Java programs. A simple API, the MAGE-STK, was generated to enable development of software reading and writing MAGE-ML documents. XML-Schema validation is not supported with MAGE-ML.

Version 1.0 of the MAGE object model consists of 182 classes which are highly connected including self-references. MAGE-OM classes are organized in 15 distinct packages. These packages group the classes logically for structured annotation of experiments. The main packages of MAGE-OM correspond roughly to the annotation categories of MIAME. In addition to the core packages, there are utility packages which are intended to aid the annotation by providing classes which are often used throughout the whole annotation. These include the *BDQ* package

[6] `http://www.omg.org`

[7] `http://www.omg.org/technology/documents/formal/gene_expression.htm`

[8] An XML application is a formal definition of a document syntax by means of XML elements.

[9] `http://www.w3.org/TR/xml/`

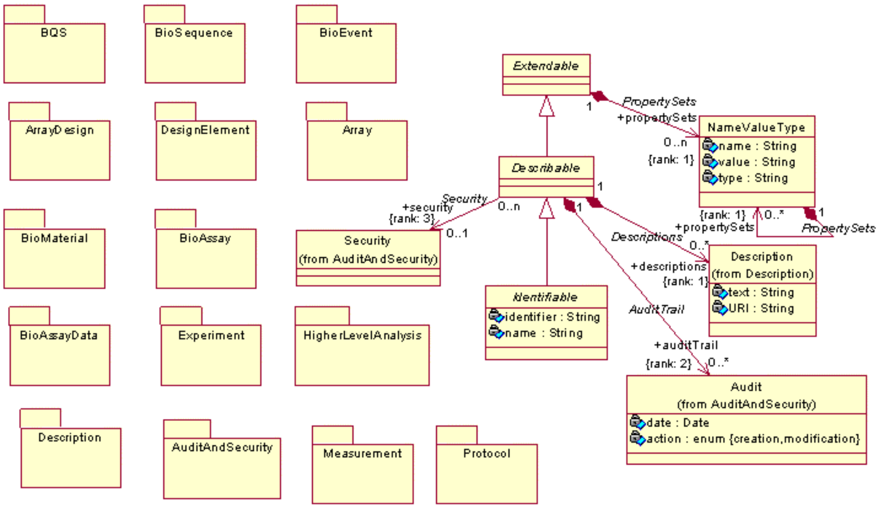


Figure 4.1: Overview of the three base classes of the MAGE-OM hierarchy and all packages. All other classes are derived from one of *Extendable*, *Describable* or *Identifiable*. The folder icons denote the 15 packages into which all classes except the base classes are divided. Diagram taken from `http://www.ebi.ac.uk/arrayexpress/-Schema/MAGE/MAGE.htm`

for providing bibliographic references, the *Description* package for providing free text and structured descriptions including database and bibliographic references. The *Measurement* package provides a class hierarchy for measurement units. The Protocol package allows to define protocols for laboratory, hardware and software applications. The protocols function as prototypes with each individual application of a protocol requiring values for specific parameters of the protocol. The *Audit and Security* package should provide means to specify who has access to specific objects and to record object creation and changes. The object model has a deep inheritance structure for many classes. The class hierarchy relies on a hierarchy of three base classes depicted in Figure 4.1.

Extendable objects allow for arbitrary annotation in a 'Name-Value-Type' format. It is stated that this type of annotation should not be used for standardized information that also fit into other classes. *Describable* objects can have added free-text descriptions and bibliographic references. *Identifiable* objects are used to specify objects which need a unique identifier. Identifiable classes subsume content to be identified and referenced uniquely within a database or document, for example experiments, arrays or sequences. The Identifiable class provides *name* and *identifier* attributes where the identifier has to be unambiguous, which means it has to be unique within a document or within a repository but not necessarily worldwide. The attribute *name* can be any possibly ambiguous name.

Although MAGE-ML is intended to capture MIAME compliant annotations, it contains mostly optional associations and attributes. This seems to be in contradiction to the MIAME specification but in fact is stringent, as often during the analysis process not all data is already available, e.g. while designing the layout of the microarray the actual sequences to spot on the microarray are not available. Also the MIAME specification might change over time and adoption of required and optional fields by changing the object model and thus the language syntax might render existing documents or databases incorrect.

The MAGE object model has been designed to be not restricted only to DNA microarrays, but to be suited for protein arrays and other types as well. In summary, the MAGE object model has been designed with flexibility in mind as it is able to capture far more details than specified by MIAME.

This flexibility, on the other hand, is also one of the biggest trade-offs of MAGE-OM, as by increasing the flexibility of a model its complexity is also increased. From the point of database design, the inherent complexity has the disadvantage of introducing ambiguity. This means that there are several possibilities of how to encode a MIAME compliant annotation into MAGE. This issue has been addressed by a document by MGED describing a standard way of encoding an annotation into MAGE. This problem might also be due to a flaw in the design process of the object model as this document appeared lately after the standardization of MAGE.

Another drawback resulting from model flexibility is that the model uses uncommon names. Naming of classes is often based on abstract terms different from concrete technical terms used in the laboratory. This approach serves a correct naming of different techniques. The *Hybridization* class, representing the process of concurrent hybridizations of labelled extract with the reporter molecules on the microarray, is a subclass of *BioAssayCreation*. This is a more generally valid term, as the process of binding the labelled protein extract to a protein microarray does not involve hybridization of complementary DNA. This class is described as "The process by which an array and one or more biomaterials are combined to create a BioAssayCreation". So far *Hybridization* is the only subclass of *BioAssayCreation* and has no attributes on its own. This can be seen as unnecessary complexity in the model.

An additional disadvantage of MAGE is its weak support for data mining techniques. In the *HigherLevelAnalysis* package classes are contained that can represent a tree structure of hierarchical clustering and a set of distinct clusters for e.g. k-means clusters but are restricted to hard cluster assignments. Unlike for a transformation event, neither the algorithm nor its parameters for generating these clusters can be stored.

Despite these disadvantages, the success of the MAGE-ML standard is evident. All three major public microarray databases support it, it is supported and distributed by a standardization organization, and reasonably stable. There has been only one minor revision of the model since its first publication from version 1.0 to 1.1. Furthermore, it seems that the need for expressive power of MAGE outweights need for simplicity, because the model is not primarily intended for direct human editing but for automated data-interchange and for use with software providing a simplified view on the data-structures. The need for interchanging the results of higher-level analyses can be neglected, too, because it should be possible to reconstruct them from the data given the parameters. In summary, MAGE-ML seems to be sufficient to annotate current and future microarray experiments.

4.1.3 Ontologies

The MAGE-OM model provides an instrument for creating structured documents of MIAME compliant information. This is already a vast improvement for data mining but it could be even more simplified if standardization was also applied to natural language terms. The model does not prescribe terms for the description of experimental annotation. Free-form annotations are problematic: It is hard to search for content in a free-text database where descriptions of the same process or material may be described with different terms. It is also hard for the experimenter to infer the meaning of different terms and to know how they should be used within an experimental annotation. Therefore, a way of defining terms for experimental annotations is required.

One way of reducing free-text descriptions is to represent a hierarchical odering terms in a hierarchical structure of classes directly in the data model. Therefore, classes for each possible value (e.g. an organism part or instrument) are required. This approach is inflexible, because it introduces a large number of classes having no attributes and methods to differentiate them, except their name. This leads to an inconsistent data representation as naming conventions change. As an example, a taxonomy of organisms could be added to the data-model resulting in a class for every organism. If a new organism is discovered, the hierarchy has to be changed, affecting data representation. By re-arranging an embedded taxonomy or removing branches, formerly valid documents would become invalid.

The problem of embedded term definitions is often addressed by so called controlled vocabularies. A controlled vocabulary is a set of terms which can be used in a specific context, being defined separately from the data-model. For a simple controlled vocabulary, there is no assumption of a structure of terms or dependence relations between terms.

An *ontology* can be used as a special case of a controlled vocabulary adding relations between terms. The term ontology stems from philosophy where ontology is the branch of metaphysics concerned existence of things. Ontologies are an attempt to categorize existing things in a way that represents knowledge about them. Within computer science, the term ontology has been adopted for an application used in knowledge based systems.

Gruber (1993) defines an ontology as an explicit specification of a conceptualization. A conceptualization is defined as “the objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them” (Genesereth and Nilsson, 1987). For such systems only those concepts exist that can be represented. The ontology thus defines the terms with which software (in this case called ‘agents’) can communicate about a given domain of interest without necessarily sharing the same knowledge base. The ontology then should contain

- the names of entities in the domain
- human readable textual descriptions
- and formal constraints for the use and interpretation of terms

Although this definition has been made with artificial intelligence systems in mind, it also seems suitable for the special case of providing a controlled vocabulary for the annotation of experiments.

Each application is likely to operate on its own distinct domain of application while a portion of shared information needs to be exchanged with other applications. The only difference between Gruber's definition and experimental annotations is the level of optimism for automatic generation of data and queries between applications. The main use case of an ontology is formal annotation and interpretation by a person that chooses terms from the ontology. Database queries for experiments can also be based on ontology terms. Still formal constraints are important to enable software to assist the user in the correct usage of the ontology. A prominent example is the Gene Ontology which provides a hierarchy of classes to annotate the function of genes (Ashburner *et al.*, 2000).

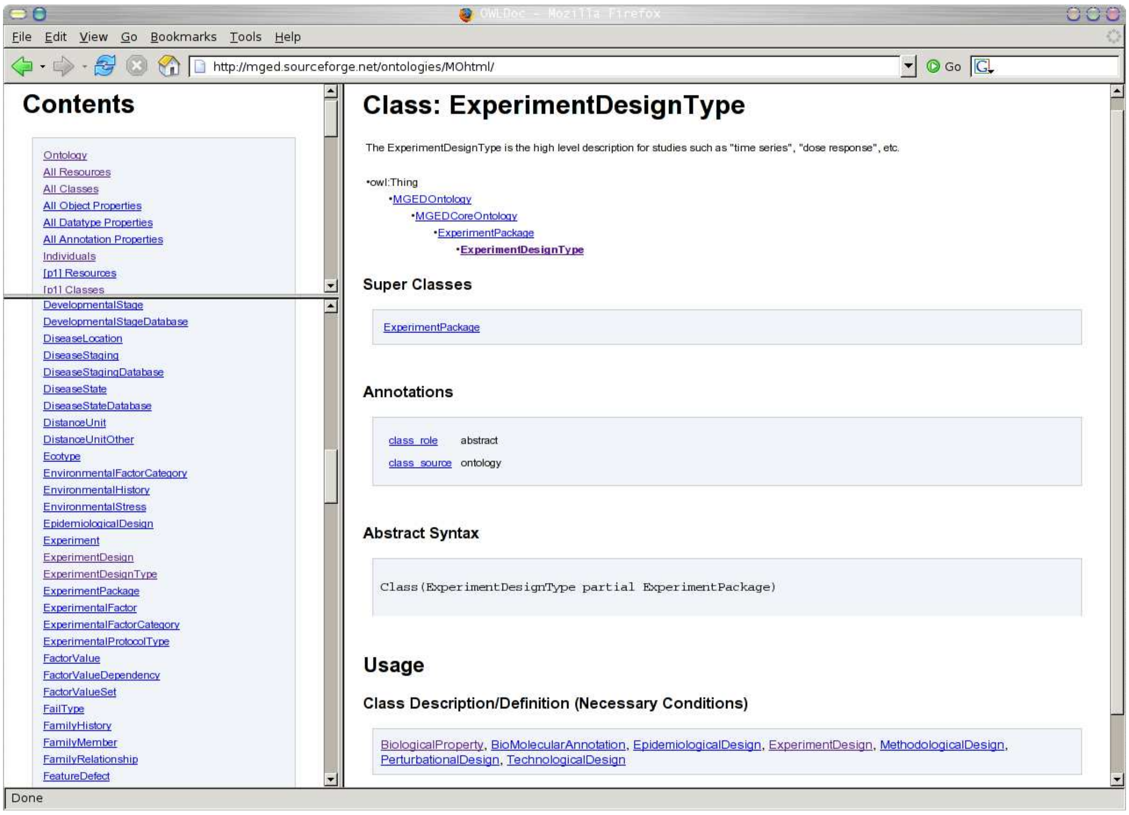
To annotate microarray experiments, a customized ontology is needed. The creation of such an ontology was undertaken within the MGED by the ontology working group. In 2002 a preliminary ontology was published (Stoeckert *et al.*, 2002). This ontology contains a hierarchical structure of classes and terms for all attributes in the MAGE-OM where an ontology entry can be referenced. The MGED ontology is a reduced ontology in that it has a hierarchical structure of classes and individuals allowing only inheritance relations.

The MGED ontology has also been enriched by formal constraints on the proper use of ontology terms. These constraints are also realized implicitly by MAGE-OM. There exists a one-to-one relation between the names of associations to *Ontology Entry* classes and the names of the classes within the ontology. Therefore, MAGE-OM provides the constraints or formal syntax for the use of the MGED ontology terms.

The MGED ontology has been originally implemented using the XML-application DARPA Agent Markup Language (DAML+OIL)[10], which can be used for specification of ontologies. The resulting files can be easily parsed by computer programs. The MGED ontology is now publicly available in several other XML-formats, including the Web Ontology Language (OWL)[11], which defines an open standard for the representation and the algorithmic processing of ontological data provided by the W3C. There is also web-browsable version of the MGED ontology (see Figure 4.2 on the facing page).

[10] `http://www.daml.org/2001/03/reference.html`

[11] `http://www.w3.org/TR/owl-features/`


OWLDoc - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://mged.sourceforge.net/ontologies/MOhtml/
Go
Done

Contents

Ontology
All Resources
All Classes
All Object Properties
All Datatype Properties
All Annotation Properties
Individuals
[p1] Resources
[p1] Classes

DevelopmentalStage
DevelopmentalStageDatabase
DiseaseLocation
DiseaseStaging
DiseaseStagingDatabase
DiseaseState
DiseaseStateDatabase
DistanceUnit
DistanceUnitOther
Ecotype
EnvironmentalFactorCategory
EnvironmentalHistory
EnvironmentalStress
EpidemiologicalDesign
Experiment
ExperimentDesign
ExperimentDesignType
ExperimentPackage
ExperimentalFactor
ExperimentalFactorCategory
ExperimentalProtocolType
FactorValue
FactorValueDependency
FactorValueSet
FailType
FamilyHistory
FamilyMember
FamilyRelationship
FeatureDefect

Class: ExperimentDesignType

The ExperimentDesignType is the high level description for studies such as "time series", "dose response", etc.

- owl:Thing
  - MGEDOntology
    - MGEDCoreOntology
      - ExperimentPackage
        - **ExperimentDesignType**

Super Classes

ExperimentPackage

Annotations

class role	abstract
class source	ontology

Abstract Syntax

```
Class(ExperimentDesignType partial ExperimentPackage)
```

Usage

Class Description/Definition (Necessary Conditions)

BiologicalProperty, BioMolecularAnnotation, EpidemiologicalDesign, ExperimentDesign, MethodologicalDesign, PerturbationalDesign, TechnologicalDesign

Figure 4.2: Screenshot of the MGED ontology web pages. On the left, the page shows a list of all available classes in the ontology. In the main frame, detailed information of a class (in this case ExperimentDesignType which is referenced in the corresponding MAGE-OM entry to annotate an Experiment) is displayed. The class hierarchy is depicted as a tree list. The subclasses which contain further sub-classes of the ExperimentDesignType class are depicted at the bottom under the caption 'Usage'.

4.1.4 Other Standardization Approaches

The apparent success of the standardization and specification effort of the MGED has demonstrated the relevance of defining and disseminating open standards for the annotation of biological experiments and data. Conclusively, this first approach has served as an example for the process of standardization of functional genomics data and has been repeated in other 'Omics sciences.

For proteomics, the Human Proteome Organization (HUPO) has taken the lead in formation of workgroups for standardization. The Proteomics Standards Initiative (PSI) has proposed the Minimum Information About a Proteomics Experiment (MIAPE) (Taylor, 2006) recommendation and the PEDRo object model (Garwood *et al.*, 2004) following the development of MIAME and MAGE. The PEDRo model was developed with the intent to trigger discussions within the proteomics community and has not yet reached the status of an adopted standard. The design of a related ontology is still discussed. For other areas like metabolomics the process of standardization has just begun.

All approaches mentioned so far, cover only a distinct type of experiment in functional genomics, despite the fact that microarray experiments often play only a limited part in the analysis of gene regulation and the study of whole regulatory networks. Often, such studies comprise analyses on almost all levels: genomic sequence, transcription profiling, proteome, and metabolome analysis. It seems reasonable to annotate experiments within the context of global scientific hypotheses instead of having experimental annotations separated by technical issues.

To foster the discussion about integration of standards, Jones *et al.* (2004) propose the FGE-OM. It constitutes mainly a combination of three models: MAGE-OM for transcriptomics experiments, PEDRo for proteomics experiments, and Gla-PSI for annotations common to every functional genomics experiment.

The aforementioned concept of data integration has, despite its accordance with the current trend towards integration of data-sources, severe drawbacks. It is still not complete, which means that specifications for metabolomics are still not existing. The included components have a variable state of maturity. While MAGE is an adopted standard, PEDRo is only a draft standard and Gla-PSI is an unsettled proposal. A change in one component might affect all experiments, even those which would be otherwise unaffected by a change.

Complexity is the other main hurdle, besides lack of maturity, for introducing software based on such a system, because each additional data model which is added to FGE-OM also adds additional complexity.

4.2 Categorization of Microarray Software

There exists a variety of software applications aiming at the analysis of microarray data, MIAME compliant annotation, storage and retrieval of experiments and data. The domains of these applications often overlap as public databases such as ArrayExpress are equiped with tools for cluster analysis. Image analysis software often has basic normalization and visualization capabilities. For practical purposes it seems still reasonable to classify microarray applications by the main use-cases covered. This is in accordance with most publications and web-sites on this topic.

- Microarray scanner and image analysis software
- Pure data analysis software
- Microarray Laboratory Information Management Systems (LIMS)
- Organism specific databases
- Public Repositories
- General purpose analysis and storage systems

Additionally, microarray software can be grouped into software which adds analysis capabilities to another software (e.g. spreadsheet software) or statistical packages such as the statistical environment R (R Development Core Team, 2005) or the computational environment Matlab. The categories could be further sub-divided; for example data analysis software can be grouped by the functionality (normalization, data-mining, cluster analysis or classification) it can perform.

4.3 Scanner and Image Analysis Software

The main task of scanner software is to provide the user with an interface to control the process of image acquisition. They provide functionality to preview the scanned images and often also false color images for multi channel technologies. As an output they return an image in TIFF format for each supported wavelength. This type of software is specific for the scanner hardware and therefore beyond the scope of this overview.

Image analysis software serves for the analysis of the resulting images, which come from the scanner software. This process is usually twofold. First, a segmentation step is performed to identify the locations of the hybridization signal and its boundaries within the image corresponding to spots of the microarray. Depending on the software, this step can be carried out manually, semi-automatically, if predefined gridding information is used, or completely automatically. In the second step signal intensities are calculated for each segmented spot. For spotted multichannel microarrays it is common to all programs to provide also an estimate of the background intensities which is the area around a spot where no DNA should be present. Additionally other statistics for each spot are computed like standard deviation of pixel intensities, quality of shape of a spot and others. The result is a quantification matrix that includes a number of measurements for each feature on the microarray.

The GenePix software is bundled with the microarray scanners from Axxon instruments. GenePix is integrated with the scanner software. The ImaGene software is another example of an image analysis software. It allows for half-automatic segmentation if a sample gridding file is provided. The AIM software has been developed at Bielefeld University for automated grid finding and spot segmentation. For the purpose of verification of automated results, a limited amount of user interaction is still required. Most of the image analysis softwares (such as GenePix and ImaGene) require a proprietary computer system and therefore lack the possibility of automated batch analysis which could be remotely controlled by another system. The commercial software Spot is implemented as a package for the statistical environment R. It can be installed on any computer system.

4.4 Pure Data Analysis Software

4.4.1 Eisen's Cluster and TreeView

The first standalone software specialized on the analysis of microarray data was the Cluster and TreeView software developed at the University of California at Berkeley (Eisen *et al.*, 1998). The software has been developed as an approach to analyze and visualize data from several microarray experiments with *Saccharomyces cerevisiae* and human fibroblasts at the same lab. The Cluster and TreeView software is implemented as a C++ application running on windows. Cluster performs hierarchical clustering and filtering while the results are visualized with tree view. Algorithms for k-means clustering, self-organizing maps and principle component analysis have been added to Cluster.

By providing these tools hierarchical clustering methods have been established for microarray data. Visualization uses heat-maps to display an ordered data matrix with red, black and green colors.

4.4.2 J-Express

J-Express was the first data-mining application for microarray data written in Java (Dysvik and Jonassen, 2001). The software was developed at the Department of Informatics, University of Bergen. Like Eisen's Cluster, J-Express contains methods for hierarchical cluster analysis, self organizing maps, k-means clustering, and principle component analysis. In fact, the only specialization of J-Express to microarray data is its user interface, where the clustered elements are named 'genes' and colors for displaying a heatmap are chosen according to Cluster and TreeView. J-Express also contains a visualization component, which is able to produce three-dimensional scatter plots and heatmaps. The software does not contain methods for normalization or filtering nor does it have database functionality. The development of J-Express has stopped already in 2001 after the product was turned into the commercial software J-Express Pro.

4.5 Public Repositories and Databases

Re-evaluation of experiments, mining existing data for additional knowledge, and using datasets for evaluation of new data-mining methods are important tasks in microarray related research. The main purpose of public repositories is to make data and protocols from microarray experiments accessible to the public. Data-mining capabilities of a repository software are of major importance. In addition, there often exists a structured workflow for data entry or upload serving the standardization and integrity of the data. Often, manual effort of a human data curator is required for obtaining protocol text and data annotations which conform to a desired structure and format. This is also valid for the terms used in the experimental annotations.

4.5.1 ArrayExpress

The ArrayExpress system is developed and maintained by the European Bioinformatics Institute (EBI) (Parkinson *et al.*, 2005). Since its deployment in 2002 it has become a quasi standard for public repositories and among GEO and CIBEX is one of the three repositories recommended by the MGED for making microarray data public. The amount of data in this repository is growing fast. In July 2005 it contained 22214 microarray hybridizations organized in 750 different experiments. Within 18 months (December 2006) these figures have more than doubled (52854 hybridizations in experiments: 1775).

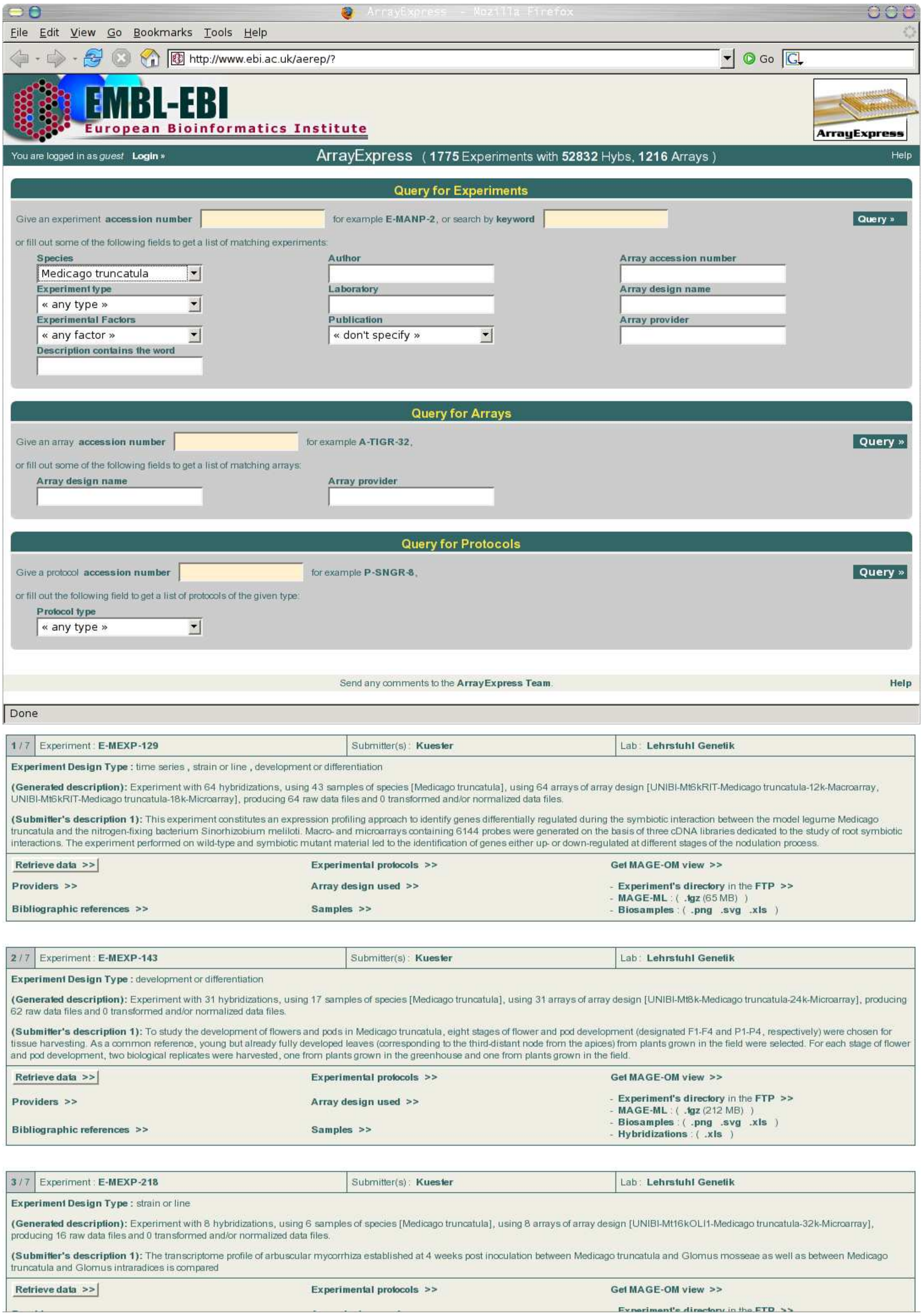
The system consists of three interacting applications: MIAMExpress for data submission, ArrayExpress for querying the repository and Expression Profiler for cluster analysis. A data warehouse is under development to support queries for experiments and expression data.

The ArrayExpress system is implemented in Java using an Oracle application server as relational database backend. Measured data are not stored in the database for the purpose of scalability by keeping the table size low. Instead, data are stored in a binary file format (NetCDF) which supports numeric array data. The database is accessed by a proprietary object-relational layer called Castor.

The development of ArrayExpress was aimed at supporting community standards. As a result ArrayExpress developers were also involved in the process of creating the MAGE object model. The system supports import and export of MAGE-ML files and stores data in a MIAME compliant annotation. The data base scheme supports this purpose by resembling the MAGE object model closely. In fact the data-base scheme is auto-generated from MAGE-OM with some added optimizations. ArrayExpress provides some levels of access control by employing password protection for selected data sets. No further level of access control is implemented at present.

MIAMExpress is a web-based application that allows users to enter experimental annotations and data in a MIAME compliant way. The annotation process is divided into several steps including sample generation and hybridization descriptions. Annotation involves the interaction with database curators located at the EBI. The curators manually check the integrity of the annotations and its compliance with MAGE terms and structure. This submission track is aimed at institutions with no bioinformatics resources to format the data. Alternatively submission of ready made MAGE-ML files is possible. This track is used mainly by large institutions employing automated analysis pipelines. Storage of raw images from the scanner is not implemented in ArrayExpress due to storage space restrictions.

The Expression Profiler component is aimed at *a posteriori* analysis of already uploaded datasets. Therefore, it is required to select datasets in ArrayExpress and transform them into a GeneExpression matrix. The only analysis algorithm implemented in Expression Profiler at present is a seemingly very fast implementation of hierarchical cluster analysis. It provides different ways to compute the distance matrix as well as for the linkage criterion. Unfortunately, Expression Profiler is


ArrayExpress - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://www.ebi.ac.uk/aerep/?
Go
EMBL-EBI
European Bioinformatics Institute
ArrayExpress

You are logged in as *guest* **Login »** ArrayExpress (**1775** Experiments with **52832** Hybs, **1216** Arrays) Help

Query for Experiments

Give an experiment **accession number** for example **E-MANP-2**, or search by **keyword** **Query »**

or fill out some of the following fields to get a list of matching experiments:

Species: Medicago truncatula
Experiment type: « any type »
Experimental Factors: « any factor »
Description contains the word:
Author:
Laboratory:
Publication: « don't specify »
Array accession number:
Array design name:
Array provider:

Query for Arrays

Give an array **accession number** for example **A-TIGR-32**, **Query »**

or fill out some of the following fields to get a list of matching arrays:

Array design name:
Array provider:

Query for Protocols

Give a protocol **accession number** for example **P-SNGR-8**, **Query »**

or fill out the following field to get a list of protocols of the given type:

Protocol type: « any type »

Send any comments to the **ArrayExpress Team**. **Help**

Done

1 / 7 Experiment : **E-MEXP-129** Submitter(s) : **Kuester** Lab : **Lehrstuhl Genetik**

Experiment Design Type : time series , strain or line , development or differentiation

(Generated description): Experiment with 64 hybridizations, using 43 samples of species [Medicago truncatula], using 64 arrays of array design [UNIBI-Mt6kRIT-Medicago truncatula-12k-Macroarray, UNIBI-Mt6kRIT-Medicago truncatula-18k-Microarray], producing 64 raw data files and 0 transformed and/or normalized data files.

(Submitter's description 1): This experiment constitutes an expression profiling approach to identify genes differentially regulated during the symbiotic interaction between the model legume Medicago truncatula and the nitrogen-fixing bacterium Sinorhizobium meliloti. Macro- and microarrays containing 6144 probes were generated on the basis of three cDNA libraries dedicated to the study of root symbiotic interactions. The experiment performed on wild-type and symbiotic mutant material led to the identification of genes either up- or down-regulated at different stages of the nodulation process.

Retrieve data >> **Experimental protocols >>** **Get MAGE-OM view >>**
Providers >> **Array design used >>**
- **Experiment's directory** in the **FTP >>**
- **MAGE-ML** : (**.tgz** (65 MB))
- **Biosamples** : (**.png .svg .xls**)

Bibliographic references >> **Samples >>**

2 / 7 Experiment : **E-MEXP-143** Submitter(s) : **Kuester** Lab : **Lehrstuhl Genetik**

Experiment Design Type : development or differentiation

(Generated description): Experiment with 31 hybridizations, using 17 samples of species [Medicago truncatula], using 31 arrays of array design [UNIBI-Mt8k-Medicago truncatula-24k-Microarray], producing 62 raw data files and 0 transformed and/or normalized data files.

(Submitter's description 1): To study the development of flowers and pods in Medicago truncatula, eight stages of flower and pod development (designated F1-F4 and P1-P4, respectively) were chosen for tissue harvesting. As a common reference, young but already fully developed leaves (corresponding to the third-distant node from the apices) from plants grown in the field were selected. For each stage of flower and pod development, two biological replicates were harvested, one from plants grown in the greenhouse and one from plants grown in the field.

Retrieve data >> **Experimental protocols >>** **Get MAGE-OM view >>**
Providers >> **Array design used >>**
- **Experiment's directory** in the **FTP >>**
- **MAGE-ML** : (**.tgz** (212 MB))
- **Biosamples** : (**.png .svg .xls**)
- **Hybridizations** : (**.xls**)

Bibliographic references >> **Samples >>**

3 / 7 Experiment : **E-MEXP-218** Submitter(s) : **Kuester** Lab : **Lehrstuhl Genetik**

Experiment Design Type : strain or line

(Generated description): Experiment with 8 hybridizations, using 6 samples of species [Medicago truncatula], using 8 arrays of array design [UNIBI-Mt16kOLI1-Medicago truncatula-32k-Microarray], producing 16 raw data files and 0 transformed and/or normalized data files.

(Submitter's description 1): The transcriptome profile of arbuscular mycorrhiza established at 4 weeks post inoculation between Medicago truncatula and Glomus mosseae as well as between Medicago truncatula and Glomus intraradices is compared

Retrieve data >> **Experimental protocols >>** **Get MAGE-OM view >>**

Figure 4.3: Screenshot of an ArrayExpress query for all experiments using *Medicago truncatula* and the query result. The list consists of seven supplementary datasets belonging to experiments carried out at the Lehrstuhl Genetik, Bielefeld University.

not freely available for download at present. As the ArrayExpress database stores previously normalized data, there is no need for a normalization pipeline within the system. As a consequence there is no possibility to apply different normalization methods to already uploaded data within ArrayExpress.

The software requirements of ArrayExpress are a major hurdle for installing the software in an open source environment. Recently, the system has been ported to MySQL as an alternative database backend (Mainguy *et al.*, 2004).

At first glance, it seems promissing to use a local installation of ArrayExpress also as an inhouse analysis system, but due to the different focus of the system as a global repository, it lacks basic functionality required. There is for instance no easy way to edit the data and descriptions once entered into the system via the web. Moreover, there is no specified way of uploading raw data to the repository and performing filtering, normalization and analyses solely with built-in tools.

4.5.2 Stanford Microarray Database

The Stanford Microarray Database (SMD) is developed at Stanford University in 1999(Sherlock *et al.*, 2001). It was initially intended as a resource for microarray research at Stanford University and external collaborators. The first version of SMD was restricted to data from two-color spotted DNA arrays. Data files from GenePix and ScanAlyze were supported. In consecutive versions, support for more image analysis programs and Affymetrix arrays was added (Ball *et al.*, 2005). The latest version also supports data export as MAGE-ML.

SMD has a web-interface for queries and data-upload. The experiments can now be annotated in a MIAME compliant fashion. SMD supports only the MGED ontology for a sub-set of all annotation fields. Terms from the ontology may be used to categorize experiments and experimental variables. The query interface (see Figure 4.4 on page 67) allows queries for organisms, experimenters, and experimental categories. The user can view experimental raw data, images, and normalized data from the results pages. Access control in SMD is realized with two different levels of access. An experiment can either be public or disclosed and thereby accessible only to the submitter. It is also possible to form groups of users to further refine access control.

Data processing methods include normalization with constant factors and linear correlation regression. Intensity dependent normalization or location dependent normalization is not implemented. Hierarchical and k-means clustering, self-organizing maps and singular value decomposition have been implemented as data analysis techniques using Eisen's Cluster. The results are browsable with the GeneXplorer application (Rees *et al.*, 2004). Quality control methods include a false-color representation of the expression ratios mapped on the array layout and a one-way ANOVA (see Section 3.4), testing for the effects of spot-location on the array and for the the effects of the origin of the spotted material on the spotting-plate (Gollub *et al.*, 2003).

SMD does not link gene annotations to the original sources but stores them locally. To keep local gene annotations up to date, a scripted update mechanism is used. The sequence databases for each organism are queried and downloaded in regular interval. The local annotations are then updated with the new information. The main disadvantage of this procedure is the high network load it creates. Also, the annotation information is up-to-date in SMD, only after a new run of the updates. The associations of organisms and databases are coded directly into the application, making it hard to configure the data sources.

SMD is implemented in Perl. It uses the Oracle software as a database and the Apache web-server. The original installation at Stanford University is running under the Sun Solaris operating system. To address high software requirements and hardware requirements resulting from the use of Solaris, SMD was ported to the Linux operating system with PostgreSQL as database. This port, named Longhorn Array Database (LAD), requires only open-source software for operation (Killion *et al.*, 2003).

The architecture of SMD does not follow a classical n-tier approach. Inspecting the Perl source-code, it can be concluded that the architecture is, in fact, a flat one. The implementation consists of Perl CGI-scripts generating the web-pages, while accessing the database directly using SQL-statements. For data-analysis external applications are called and data is interchanged via files with them. This architecture is rather inappropriate for extending and maintaining the application. Also, SMD lacks a structured interface for accessing the data or adding more analysis functions.

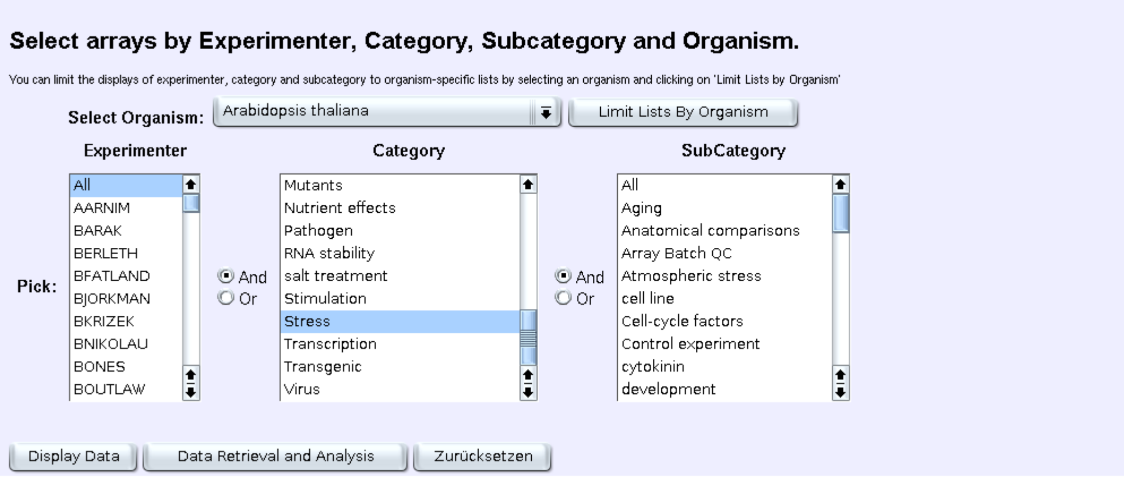
In summary, it seems that the SMD software is rather inappropriate to set up a local microarray database. The main reason is the lack of adherence to modern paradigms of software design. As a result, it will require a much larger effort to adapt the system to individual requirements than necessary.

4.5.3 CIBEX

The CIBEX database has been developed at the Center for Information Biology and DNA Data Bank of Japan (Ikeo *et al.*, 2003). CIBEX supports MIAME compliant experimental annotations by using web-based submission tools. Moreover, it has a rather simple web-interface for queries. The interface allows queries for experiments and arrays. CIBEX is a web-based application, implemented in Java using the Tomcat4 application server and a MySQL database. A Java application provides scatter plots and hierarchical clustering on the client side. The database stores quantification data and normalized data, but no images due to storage requirements. Up to now, it does not support MAGE-ML or the use of ontologies.

The authors point out that the submission process can be done only after personal communication between the submitter and the database curators, in order "to clarify every detail of the the submission process". This is driven by the intent to increase the level of data consistency, but the need for human effort is increased on both sides.

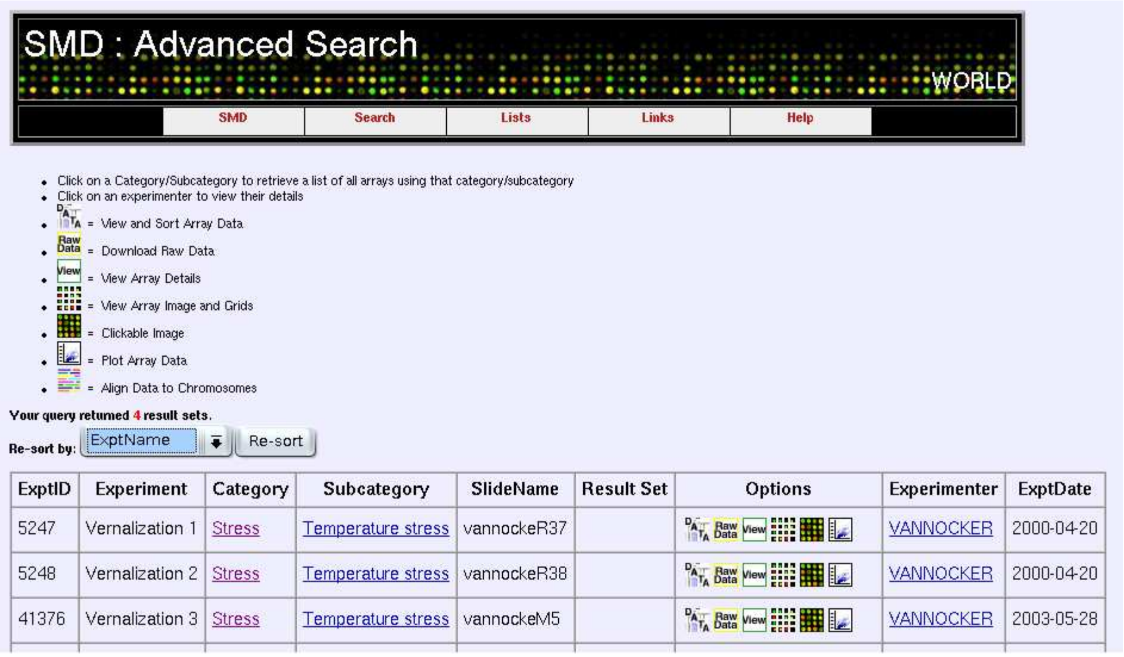
The CIBEX database does not appear to be used very actively. As of November



Select arrays by Experimenter, Category, Subcategory and Organism.

You can limit the displays of experimenter, category and subcategory to organism-specific lists by selecting an organism and clicking on 'Limit Lists by Organism'

Select Organism: Arabidopsis thaliana



- Click on a Category/Subcategory to retrieve a list of all arrays using that category/subcategory
- Click on an experimenter to view their details
- DATA = View and Sort Array Data
- Raw Data = Download Raw Data
- View = View Array Details
- ▦ = View Array Image and Grids
- ▩ = Clickable Image
- 📈 = Plot Array Data
- ☰ = Align Data to Chromosomes

Your query returned 4 result sets.

Re-sort by: ExptName Re-sort

ExptID	Experiment	Category	Subcategory	SlideName	Result Set	Options	Experimenter	ExptDate
5247	Vernalization 1	Stress	Temperature stress	vannockeR37		DATA Raw Data View ▦ ▩ 📈	VANNOCKER	2000-04-20
5248	Vernalization 2	Stress	Temperature stress	vannockeR38		DATA Raw Data View ▦ ▩ 📈	VANNOCKER	2000-04-20
41376	Vernalization 3	Stress	Temperature stress	vannockeM5		DATA Raw Data View ▦ ▩ 📈	VANNOCKER	2003-05-28

Figure 4.4: Screenshots of the query interface of the SMD database (top) and the resulting list of experiments for the category stress (bottom).

2005 it contained only 3 publicly available experiments, 5 different types of arrays, and 448 hybridizations. The low activity compared to the other repositories might be a result of the rather cumbersome submission process.

In contrast to ArrayExpress and SMD, the repository software is not freely available, and thus CIBEX cannot be considered as a solution for local installations. In summary, the low activity of CIBEX and its limited capabilities make it rather unfavorable for submitting microarray data.

4.5.4 NCBI GEO

The Gene Expression Omnibus (GEO) database has been developed at the National Center for Biotechnology Information (NCBI) of the USA (Edgar *et al.*, 2002). It is intended to serve as a public repository for molecular abundance data with a focus on mRNA and genomic DNA. Submissions to GEO can be made either interactively via web-based forms, as a bulk data upload via FTP of text files, or as a relatively new feature via upload of a MAGE-ML file.

The submission forms follow the MIAME guidelines, but responsibility is left to the submitter to provide meaningful MIAME compliant annotations for all three submission paths. The manual curation effort is limited to checks of syntactic correctness and correctness of data organization. This principle can be seen as the opposite of the curation concept of CIBEX and also ArrayExpress. Both involve a higher level of human interactions during the submission process.

Like the other public repositories, GEO holds quantification data of experiments, but it does not allow to store the original image files.

Data retrieval was at first possible only by accession numbers of experiments, or by categories of experiments. Recently, some tools for data-mining for individual expression levels of specific genes, for species and experimental conditions have been added (Barrett *et al.*, 2005). Access control is provided only for complete datasets. The current policy of GEO is to keep submissions private up to a maximum period of six months.

Very few information is available on the actual design and implementation details of the underlying software. Data is stored in a mixture of relational databases and raw text files. No statement is made about the availability of the software, but it is not distributed under any form of open source license.

In summay, GEO is a very actively used public repository for microarray data. It is at present impossible to evaluate the quality of the database software as it is unavailable and no other installation of the GEO software exists.

4.6 LIMS Software

The Microarray Database of Gene Expression (MADGE) has been developed at the University of Florida, Gainsville (McIndoe *et al.*, 2003). It is a representative example for one of the few open-source Laboratory Information Management Systems (LIMS) for microarrays. As pure LIMS systems lack processing and analysis features, they can only become a part of a larger microarray system.

This software is specialized for storage of data from two color spotted microarray experiments. It relies on a relational database which models the laboratory workflow of array production and microarray hybridization. The system is designed as a 3-tier architecture that relies on Microsoft SQL Server. The application layer and the middle layer are written in Visual Basic using the Microsoft .COM architecture. The middle tier provides an API. The application layer uses Active Server Pages to generate the web-interface.

The system does not support MAGE-ML or ontologies and it seems unclear how it handles textual protocols of experimental steps. MADGE is restricted to two-channel spotted microarrays, and cannot be applied for the widely used single channel techniques.

4.7 General Purpose Analysis and Storage Systems

The following section will introduce systems that combine storage and analysis features in one system. Such a system can come in the form of a monolithic system such as BASE or Rosetta Resolver, or as a loosely coupled suite of tools, namely TIGR TM4. There are large differences in how the various tools interact with a database and integrate with each other. None of them provides integration with external data sources for the purpose of data analysis.

There exist more tools of this type, which will not be covered in detail. This is mainly for the reason that they resemble representative tools, for example the maxd suite (Hancock *et al.*, 2005) and Nomad, or that they are no longer actively maintained; such a case is found with GeneX-Lite, which has been preceded by the commercial application GeneX professional. A rather dated comparison of such open-source tools is given by Gardiner-Garden and Littlejohn (2001). Expressionist is another interesting product developed by GeneData, Basel, that requires a commercial closed-source license, and therefore could only be inspected in a sketchy way.

4.7.1 TIGR TM4

The TIGR TM4 application suit (Saeed *et al.*, 2003) has been developed at The Institute for Genome Research (TIGR). It consists of 5 separate applications: Microarray Data Manager (MADAM) which stores microarray raw data and provides LIMS functionality, TIGR Spotfinder for image analysis, Microarray Data Analysis System (MIDAS) providing normalization and filtering, and the Multi Experiment Viewer (MeV), which can visualize microarray data by heatmaps and perform cluster analysis. MADAM, MIDAS and MeV are implemented in Java making them platform independent while TIGR Spotfinder is written in C and is running only on windows systems.

Raw data is stored in a relational database by MADAM which supports MIAME compliant annotation. MADAM can also export data in MAGE-ML format but MAGE-ML import is not implemented. MADAM allows to enter information about RNA production and samples and hybridization protocols. No ontology can be employed automatically up to now to provide annotation for some fields. Instead the software relies on some hard coded controlled terms for data entry.

At present there is no component handling information about array production. For interoperation with other programs of the suite, the data have to be exported into a proprietary flat-file format.

MIDAS allows for normalization and filtering of the raw data. A lowess-regression normalization can be performed as well as global intensity normalization. A local intensity normalization based on the position of spots in a grid is not implemented. The user may define filters based on several statistics of the measured values. MIDAS seems to be applicable to dual-channel microarray data. Data import and normalization of Affymetrix data is not supported.

Normalized and filtered data from MIDAS can be exported into a file and opened with MeV. This tool has a large set of implemented clustering algorithms like k-means clustering, hierarchical clustering, Self Organizing Maps and gene shaving. It also supports dimensionality reduction with PCA and classification with Support Vector Machines. Figure of merit graphs support the user to chose parameters for the clustering algorithms. Also the clustering information can be mapped onto metabolic pathways with MeV. There is no data integration in the opposite direction, which results in that one cannot use external sources of genome annotation for half-supervised cluster analysis or visualization automatically.

MeV has a Java API for extending the software with additional analysis functionality, but at the time being no module which uses this API is known to the author. The other software packages lack a well defined API, thus extending them, requires modifying the existing source code.

MIDAS and MeV do not provide means for MAGE-ML export of the chosen analysis methods, parameters and resulting data sets. To achieve a complete MAGE-ML file describing all analysis steps performed, the user would have to code this information by hand. All TM4 software is available free of charge under an open source license.

As a conclusion, the available analysis methods for dual channel systems seem to be rather comprehensive in TM4. Especially in MeV we find a large collection of state of the art algorithms. On the other hand, the level of integration between the applications of the suite as well as the integration with external sources could be improved.

4.7.2 BASE

The BioArray Software Environvent (BASE) has been developed at the University of Lund, Sweden (Saal *et al.*, 2002). At present it seems to be the most wide-spread open source system for centralized storage and analysis of microarray data. BASE 1.0 is a web-based application server, that uses a relational database management system and an Apache web-server. The main functionality is based on server side PHP scripts. The server is designed as a Linux solution. The computational tasks of data analysis are performed by executables written in C and C++. The web-interface of BASE uses a mixture of HTML, Java-applets, and JavaScript and requires only a web-browser.

The LIMS component of BASE allows for a MIAME compliant annotation of experiments, providing the possibility to specify biological samples, RNA extraction, labelling, and hybridization, as well as array designs. Additionally, BASE LIMS captures information about the array production process and the molecules spotted as well as their sequences. Quality control measurements can also be added to the database. The use of ontologies is not supported, but simple controlled vocabularies can be defined.

The system provides basic access control and the possibility to share data objects with other users, whereas it does not provide explicit user definable roles to manage access privileges for individual tasks. Administrative tasks for a project are restricted to administrators and cannot be passed in total or partially to other users.

The analysis pipeline is based on so called PlugIns. By PlugIns, the authors describe executable programs running under the server operating system. The PlugIn architecture provides the data as a file in a standardized format which the PlugIns must be able to parse. The output of the external programs consists also of files in that format which are then recognized and registered in the database by the PlugIn system. The advantage of that system is that external programs are self-contained and the relative simplicity of having a single interchange format. The disadvantage is that every program to be included as a PlugIn needs a parser and output routines for this format.

The system includes PlugIns for normalization, filtering, and multidimensional scaling. Other functionality, like cluster analysis and significance tests, have been added by external contributors. Moreover, a MAGE-ML exporter has been added as a PlugIn, while an import functionality is not available. As for TIGR TM4 the MAGE-ML exporter does not include annotation of data normalization and analysis.

From the point of view of a programmer, BASE 1.0 has a very important drawback: the system lacks a structured and documented programming interface. All the server scripts communicate directly with the underlying database. It is also not possible to invoke server functionality from remote systems or to link to data other than by using the web interface to generate queries.

In order to address shortcomings in the architecture of BASE 1.0, the authors have recently released the BASE 2 system. It is a complete redesign using a 3-tier Java-based architecture supporting a structured API and remote access to database objects. BASE 2 is designed as a web-application using Java Enterprise Java Beans for client server communication. Unfortunately, neither ontologies nor MAGE-ML support are yet implemented. Analysis PlugIns for single-channel data are also missing in the new version. Data integration with other resources is not found within the current web-interface as well as methods for cluster analysis are missing. Moreover, the system is only partially MAGE-compliant, because its database model represents only a subset of the MAGE-OM classes.

In summary the BASE system appears comprehensive in the LIMS component and extensible in data analysis while its major drawbacks are lack of visualization components and interoperability with other 'Omics software. BASE is available under a free open source license.

4.7.3 MARS

The Microarray analysis, retrieval, and storage system (MARS) was developed at the Graz University of Technology (Maurer *et al.*, 2005) and it is the most recent system of those described here. MARS has a 3-tier architecture based on the Java 2 Enterprise Edition (J2EE). As most other applications it relies on a relational database management system as backend and the JBoss as the middle tier. On top of that it has web interface using Java Servelets and Java Server Pages.

MARS has components providing LIMS capabilities for the array production process, sample preparation and hybridization workflow. These processes can be described in a MIAME compliant way by employing free text, numerical scoring and ontologies. Also experiments can be described in a MIAME compliant way by using terms from the MGED-Ontology to classify the experimental design. This seems a promising approach towards integration of ontologies and apart from MIAME-Express/ArrayExpress MARS has the highest level of ontology integration. However, it does not provide means to change terms in the ontology or exchange the ontology with other customized ontologies. This is due to the fact that MARS does not have an independent ontology database allowing for the definition of user defined ontologies.

In addition to the standard LIMS features MARS also offers storage and analysis of external quality control measures (e.g. to assess RNA quality), gel images of the labeled extracts to assess labelling quality, and to store and retrieve output generated by these methods.

Apart from the quality control analysis, MARS has no analysis capabilities on its own. Instead it relies on two additional softwares: ArrayNorm for data transformation and Genesis for cluster analysis. ArrayNorm can perform normalization of datasets and Genesis can perform the usual cluster analysis algorithms. The programs have to be installed on the computers of the users and data need to be transferred back to MARS after analysis. This approach does not allow for standardized pipelines of analyses and it has the disadvantage that software needs to be deployed on the client machines. Also it delegates computationally intensive tasks to the clients. This can be seen as an advantage as it saves load from the server, but requires properly installed client systems with computational resources.

The architecture provides an API for programming and extensions. The LIMS system also seems to be comprehensive compared to other packages. It provides a basic ontology interface and partial MAGE-ML export.

On the other hand, the analysis interface is rather poor and MAGE-ML import is not implemented. As all other systems, MARS provides only partial MAGE support. Integration with genome annotation databases has also not been implemented except by hyperlinks.

4.7.4 Rosetta Biosoftware Resolver

In contrast to the applications described so far, Rosetta Resolver is a fully commercial product developed by Rosetta Biosoftware. Due to the lack of freely available documentation or evaluation licenses, the facts described in this evaluation were gathered by working with a Resolver installation at the Max Plank Institute for Infection Biology, Berlin.

Resolver has been implemented as a client-server Java application using Oracle as DBMS. Resolver contains MIAME compliant experiment annotation features and allows for data export with MAGE-ML. MAGE-ML import is possible only for a small restricted subset of the language. All computations are carried out on the resolver server machine, allowing for relatively small client systems. The resolver server carries out all computationally intensive tasks. As it does not allow for the use of a batch-queuing system or other mechanisms of distributed computing, a single large multiprocessor system is required.

This solution is expected to scale poorly for high data volumes. Data analysis is carried out using predefined analysis modules or R scripts. Numerous preprocessing and analysis modules are available. Within the Resolver software, individual microarrays are grouped in so called 'profiles' consisting of replicate sets of microarrays. These are further grouped in 'experiments' consisting of a set of profiles. 'Experiments' can further be organized in 'experiment definitions'. Every experiment has a single data table which is populated with normalized ratio or intensity values during data upload. p-values of a statistical test are also computed during data upload. As there is only a single measurement table, it is not possible to compute and compare the effects of different normalization or preprocessing approaches within a single experiment. The grouping of profiles is also invariant for all computations on an experiment.

A very promising feature is the use of so called 'Broadcasts' of selected sequences between multiple windows, containing lists or visualizations. For example, while selecting a region of a scatterplot containing a set of interesting genes, this selection can be broadcasted into all other windows. If another window contains a table of filtered expression values, not necessarily from the same experiment, the intersect of the genes selected in the plot and the genes listed in the table will also be selected in this table. The selection can then be used as input for further analyses like clustering. This feature allows for very flexible and intuitive data-analysis, on the other hand there is a substantial risk of loosing objective criteria for data-analysis and filtering.

Rosetta Resolver is a product certified by the United States Food and Drug Administration (FDA). The certification states that the software is in accordance with FDA regulations for the deposition of experimental data from a drug-design process. Hence, it is practical to apply Rosetta Resolver in life-science companies for large-scale screening projects for potential drugs for the United States market. The certificate does not state the correctness of the software, the accuracy of its results, or the absence of software errors (which is, of course, infeasible to prove).

In summary, the Resolver software offers promising features, like interactivity and broadcasting of selected items to support intuitive selection of expressed genes. The downside is a intuitive workflow which may direct the user from objective criteria such as p-values to subjective criteria such as visible spots. Hardware and sotware requirements, incomplete handling of normalized data, and lack of data-integration and extensibility with respect to data-analysis make the software unfavorable in an academic environment.

4.8 Extensions to Existing Programming Environments

4.8.1 Bioconductor and R

The Bioconductor project is an open source project for the development of bioinformatics tools (Gentleman *et al.*, 2004). The projects main goal is to provide extensions for the statistical environment R for the analysis of genomic data (R Development Core Team, 2005). There are several research areas in functional genomics which are covered by Bioconductor like visualization methods for DNA and amino acid sequences, gene ontology classification and general functional annotation.

A wide range of packages within Bioconductor is dedicated to the analysis of microarray data. The *affy* package, for instance, covers import and export of Affymetrix file formats. Several more packages for normalization and significance testing of in-situ synthesized and spotted microarrays can be found in the Bioconductor package list. Other packages add functionality for import, export and manipulation of MAGE-ML file. Another strength of the Bioconductor project is its wide range of visualization options, which are based on the R plotting engine.

Integration with the statistical programming language R provides a level of flexibility which can hardly be achieved by a standalone application without a programming language interface. The results of array specific analysis methods could be passed to the general purpose methods implemented in R like cluster analysis, statistical tests and visualization options. An experienced user might also employ additional functionality from a large collection of general purpose R packages or even implement own analysis algorithms in R. R can be further extended as there are bindings to many other programming languages, for example C, C++, Perl and Fortran. Bioconductor also contains repository functions like interfaces to database management systems.

It is unavoidable to learn the syntax and data structures of a programming language with a very rich type system providing contradictory paradigms of programming languages[12], but without type-safety. Basic functionality is often hard to find within all the R packages, even with a firm background in statistics and experience in using the R language, its manuals, and online help.

In summary, R and Bioconductor provide extremely flexible, function rich, extensible and computationally efficient tools. The number of methods for microarrays is almost complete with respect to the published methods. They are hard to learn even for computer-scientists, and therefore absolutely inappropriate for unexperienced users and also for centralized repositories and data management.

[12] Most relevant and complicating paradigms are providing a functional and object-oriented language with two incompatible approaches to object-orientation.

CHAPTER 5

Requirements and Specification

The existing software systems depicted in the previous chapter all have specific limitations. These are mainly located in the field of software engineering like failure to employ design patterns and to provide defined interfaces. They also lack completeness and extensibility of analysis functionality and data models, data integration with other applications, and seamless analysis pipelines with automated annotation of transformations.

It is not desirable to maximize the sheer amount of functionality of the user-interface but to provide the functions that are required. Otherwise, unnecessary and thus unused functions could distract the user's attention, make the interface overly complex, and finally leave the programmer with unused and untested code fragments.

Lack of important functionality without being able to extend the software later on can lead to low acceptance of the whole effort. As a consequence, requirement analysis and specification are the first steps in the development process of a software. This analysis can also be seen as the attempt to achieve the largest possible intersection between three sets of functions: the functions the users require, the functions the software finally has, and the functionality that is most beneficial to accomplish the tasks the users really need to perform[1].

5.1 Use Cases

During a use case analysis, the tasks a software system has to perform are defined from the users' perspective. A use case model can be described using the Unified

[1] In case of conflict one might maybe want concentrate on the latter two.

Modeling Language (UML) (Rumbaugh *et al.*, 1999).

To gain a rough impression of the system, the functions such a system will have to perform are collected in an informal way. The required functionality of the software was gathered from two sources: First, in multiple discourse sessions with potential users having no specific background in programming, the basic requirements from the point of view of regular standard analyses were acquired. Second, it was also important to gather requirements from software developers. This is unusual for business-oriented software, which tends to follow mostly requirements or aspects of merchantability, for academic software it seems reasonable.

For this project, two aspects are important: the software shall serve as a tool for biologists to perform routine tasks of data-analysis efficiently, and it shall be used by computer-scientists as a tool to develop and evaluate and deploy new methods. The second aspect is very important, as it provides the potential to be creative and to provide functionality that is unparalleled in other software.

The functional requirements were classified into distinct groups to gain a more structured overview:

User interaction The user interface has to support the user in handling the software and must be robust against errors. The user interface also has to be complete in that it provides all functionality to perform advanced analysis tasks.

Data handling An application for microarray data has to deal with the large amount of high dimensional datasets this techniques create. To design a consistent and efficient system, good insight into the structure of that data is of primary importance.

Data integration functionality This class contains functionality responsible for any communication, linking and data exchange with other software. The decision to follow established open standards, namely MIAME and MAGE-ML, is the most important requirement in this class.

Data analysis Data analysis functionality is directly associated with required analysis methods, successive organization of methods, and their modular extension. Analysis functionality was kept separate from user interface functionality, backend functionality and data integration functionality, as it relies on many aspects of all of the other fields and consequently would have been present in all of them.

Administrative functions These functions subsume tasks like user and account management and database maintenance.

Technical aspects describe general technical requirements resulting from the existing software environment like the need to be able to have an easy to install version of the software or the expected software environment of potential users.

According to the intended user groups, a set of roles for user interaction have been defined and afterwards used in the modeling of user interactions. These roles comprise:

User The role of a user describes the main everyday experimenter who imports data and runs analyses.

Project administrator A project administrator has to manage the project by doing the account management, setting up available analysis tools and predefining parameters.

Maintainer Often the task of setting up analysis parameters and adding information about available microarray layouts to the repository should be delegated by the project administrator. A maintainer keeps track of changes to the database and delete or edit falsely annotated or incorrect data.

Guest A guest user is able to only view at existing data in the database. For a guest, the entered data should appear as a static repository, that can be browsed and searched but not altered in any way.

Lab-employee A lab-employee is defined as a person working in the laboratory with samples, microarray hybridizations and data acquisition but not necessarily with data analysis.

The roles have been used to define corresponding actors in the use-case diagram. Figure 5.1 depicts an example of such a use case for a LIMS component of a data storage system. The diagram contains three actors: a normal user, a lab-employee, and a project-administrator. The possible use-cases are depicted in the center of the diagram while the actors perform actions with respect to each use-case.

Use-case diagrams can represent only rather simple operations such as directed interactions. Due to this limitation, each use-case model requires additional textual descriptions. Especially in the functional category of data integration, use-case diagrams did not appear to describe complex interactions sufficiently. An example for such a complex interaction is given in the following textual description: A user initiates data analysis. Class labels for the each expression vector can be derived from an external source, depending on the functional classification in another application, for example a genome annotation system.

5.2 User Interface

The proper design of a user interface is the major aspect for usability, and hence acceptance of a software system; a graphical user interface is the only visible portion of the system for its users. The steps taken during the conduction of microarray experiments have a high level of complexity. This applies equally to laboratory workflows as well as consecutive analysis steps. The user interface has to support

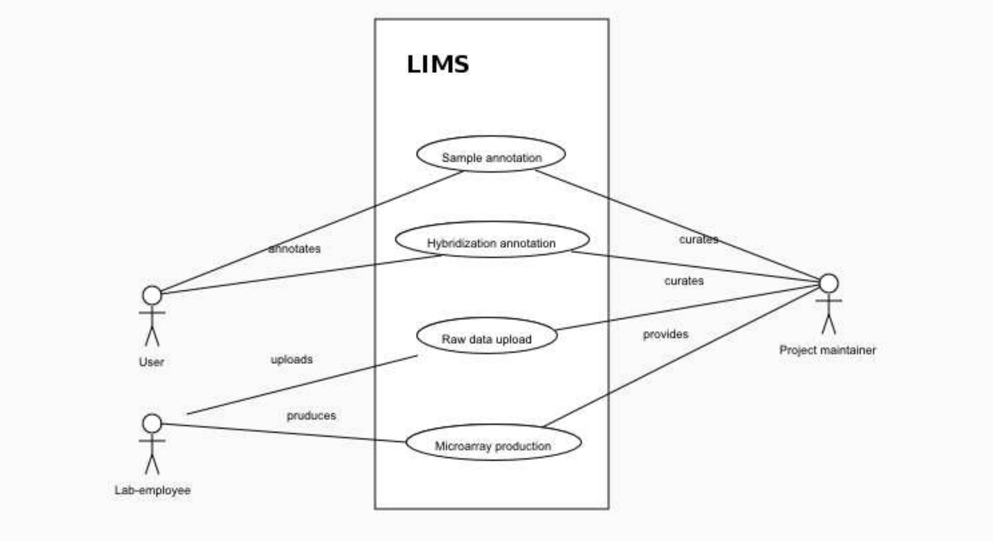


Figure 5.1: Use-case for the annotation and data acquisition process of the LIMS system.

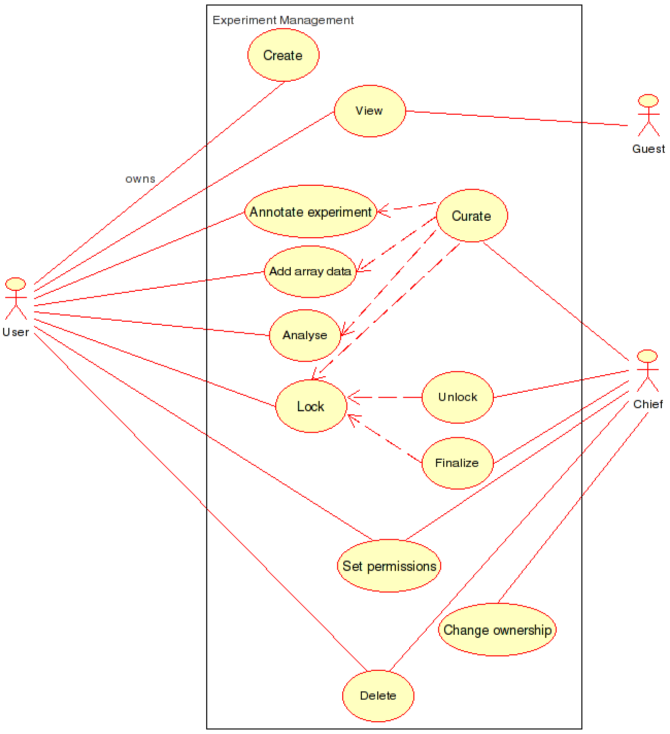


Figure 5.2: Another more complex use-case for the management of experiments and data. Several steps in the life-cycle of an experiment depend on each other. For example, the curation of and experiment by an administrator depends on the preceding finalization of the experiment (depicted by dashed arrows).

users in entering complex laboratory parameters and resulting data. When appropriate, the software should resemble the workflow users are accustomed with while it should at the same time provide help on which steps to take next.

In order to completely capture the wide variety of possible protocols and their variations, the underlying data model of the software is likely to be itself highly complex. It should not be exposed to the user, but worksteps of analysis should be rather decomposed into more easy to comprehend steps.

Explorative analysis of acquired and transformed data may benefit from a high level of interactivity of the system. This means that for every relevant detail like genes and data fields it should be possible to increase the level of background information displayed by simple mouse interactions. This additional level of information should always be available with the same representation, regardless of where in the interface a hybridization, biological sample, specific gene or data entry occurs.

Furthermore, the user interface has to be readily available without complex software installation procedures, to reduce software maintenance costs. The users work with highly heterogeneous software and hardware and require a high level of compatibility with the different infrastructure.

Regarding different modes of interaction, such as graphical user interfaces and command-line interfaces, the optimal combination of flexibility and ease of use would be an interface, as flexible as a command-line solution (for example R) and as easy to use as a graphical user interface. It is almost certain there has to be a trade-off between flexibility and ease of use in a real-world application.

5.3 Technical Requirements and Preconditions

5.3.1 Data Handling

As a high-throughput method microarrays create large amounts of data which have to be handled in an efficient and reliable way. Any step of data analysis usually involves the creation of more derived transformed datasets. The amount of data is further multiplied by addition of an increasing number of new experiments carried out over time. This creates the necessity to use scalable technologies which can keep data manageable while the underlying database grows and allow the database to be searched.

Redundancies can occur during use of the system by adding identical datasets many times and by repeated application of analysis methods with the same parameters. That way, the amount of stored data is multiplied without any added information. Making existing data reusable for other users is a way of addressing this problem. The user needs to be able to use already uploaded microarray datasets for different types of analyses and re-organize them into virtual experiments. If there is already an appropriate normalized dataset for a specific microarray, there should be no need to recompute it. Adding more normalized datasets to check effects of different methods or parameters should remain possible and datasets based on previous normalization steps should be preserved.

Security by authentification and authorization for each individual dataset is a high-priority requirement, especially in large distributed projects where data privacy is often a serious concern. Another rationale is observed during data collection and analysis; it is often unclear if data quality justifies further analysis efforts. Therefore, the experimenters tend to make only data of good quality visible to other project members.

It is possible to set up individual repositories for each project, but this has a severe impact on database maintenance. With the number of individual repositories, the administrative effort increases as well. Data scattered over several repositories is an obstacle for data mining as the data mining process has to be repeated for each repository. It is therefore required to be able to reduce the number of repositories by forming common data repositories. In addition, a way of interchanging data between repositories is required to create federated repository from several independent ones.

5.3.2 Data Analysis Capabilities

The available methods for pre-processing and analysis should include most algorithms presented in Chapter 3 for which their applicability to microarray data has been shown in the corresponding publications. Every existing system, presented in Sections 4.4 and 4.7, tends to implement at least a subset of this functionality.

To be able to carry out self-contained analyses within the system, methods for normalization and preprocessing, data filtering, statistical tests, and machine learning should be integrated. Additional visualization methods are required to enable researchers to interactively locate interesting genes, experiments and groups thereof. For the purposes of publishing the experimental results, these visualizations should also be exported in printable formats.

The analysis methods examined so far can be applied in sequential steps. While not every possible arrangement of such an analysis sequence makes sense, there are still many possible combinations of methods. Each method accepts data of specific type and produces data of another type. It is important to be able to store data of consecutive analysis steps for reference, but if this was mandatory even for all preliminary steps, storage requirements would grow excessively. Accordingly, it should be possible to combine methods without keeping intermediate results.

Apart from the possibility of having multiple options of arranging methods in many different ways, almost every analysis method has a set of parameters controlling its behavior. Hierarchical cluster analysis, for example, requires to specify the method for computing distances between genes (e.g. Euclidean, manhattan, correlation coefficient) and the method for computing the inter-cluster distances (e.g. average linkage, complete linkage, Ward's method). Pre-filtering steps are necessary to compute a gene-expression matrix, where repeated measurements must be consolidated by computing an empirical location statistic, namely mean, median, or a trimmed mean. It can also be reasonable to scale the resulting gene-expression matrix to have equal variance and means of columns, and for time-series to scale the mean of each row.

Exploring different parameters might result in an inefficient try-and-error approach. It is therefore important to provide guidelines on how to set up a consecutive series of analysis methods, and to provide unexperienced users with a set of standardized parameters to start with. All parameters used in an applications of methods need to be recorded. The choice of some methods depend on the microarray platform: normalization of two-color cDNA arrays differs from normalization of single channel arrays (see Section 3.3.2). The system should support the user in this decision process based on the type of microarray. As many new methods are constantly being developed, the software needs a mechanism to incorporate new methods to extend its analysis capabilities without further programming effort.

5.3.3 Data Integration

Some analysis methods can benefit from or even require the presence of prior knowledge. Supervised learning methods, as an example, require class labels for the purpose of training and performance evaluation. Cluster analysis methods could also benefit from the presence of external labels: Functional classification of genes from genome annotation systems can be used to assess the quality of clusters. Another possible use-case is the projection of measured data on the chromosomal location of genes. This approach can be used for tiling microarrays to detect transcribed regions of the chromosome. The exact location of representative sequences on the chromosome can be retrieved from specialized genome annotation systems such as GenDB (Meyer *et al.*, 2003).

Allowing the retrieval of microarray data by remote software is also an important feature, for instance in order to map the expression levels of genes to metabolic pathways. That way, the specialized pathway software can display the metabolic network including displays of the actual expression level for each involved gene. Gene expression information may also serve as additional annotations in genome annotations, stating that a specific gene was differentially expressed under a specific experimental condition. This requirement raises a new problem, both motivated by the complexity of microarray data.

The technical aspect to solve is about appropriate interfacing. Neither can the remote application (client) be required to support MAGE-ML, nor should the complex structure of MAGE-ML be exposed to the client; this would be an unwarranted overhead for simple queries. On the other hand, there can be more complex queries which combine many different aspects of the experimental annotations with external data-sources. For this purpose it seems justified to provide full interoperability for the software programmers of remote applications.

There can exist a very large number of measured and derived single-datum points for a gene within the system. It is a non-trivial task to decide which single-point measurement or which expression vector should be exported to external applications. Hence, the external user has to be able to further restrict queries for expression data not only to genes, but to a defined set of experimental conditions and data-types. Although, applications often require singular measurements for a gene, one has to keep in mind that this is a very reductionist view on gene-expression data. This problem needs to be solved in the design of an appropriate query interface.

In conclusion, it is necessary to provide at least two views on data-interoperability:

- A simplified access model by which simple queries for single datum-point can be processed and
- a complex access model, by which complex queries can be implemented, which are not specified at this time and may use all information in the repository.

5.4 Existing Systems Revisited

After having acquired use-cases the software has to fulfill, and having evaluated the technical and problems that need to be solved, the existing systems have to be re-evaluated with respect to these findings. The core question can be formulated as: can existing software be used and to which extent do these systems require and justify modifications.

First, the desired system can be classified within the categorization of microarray related software in Section 4.2. Pure analysis software can be excluded immediately, because a centralized data repository is required for the system. Necessary modifications to provide database and communication functionality would result in building the database functionality from scratch. It is still important to provide interfaces to some external analysis solutions, because some of these tools provide valuable visualization and analysis methods.

Pure LIMS solutions lack analysis capabilities, and, even more important, data structures to store processed results. Together with pure analysis software LIMS systems seem to be the least optimal choice to implement a comprehensive system. Nevertheless, LIMS capabilities remain an important aspect of the specification and justify the inclusion of a LIMS component.

Public repository software systems, such as ArrayExpress, bear the advantage of being almost complete with respect to their internal data representation. Some of them encompass some built-in analysis methods. Repository functionality is required to store and share data in collaborative functional genomics projects. After completion of a microarray study, it should be made publicly available. A transmission to a public repository can provide a way to make data publicly available in a MIAME compliant format, which is often mandatory. Notwithstanding their relevance for publishing a completed analysis, the repository software is not suited to support data acquisition and exploratory analysis.

The class of general purpose storage and analysis systems appears to be the most appropriate category. These systems combine data storage with analysis capabilities, as well as LIMS functionality. A rather easy to extend plug-in architecture is found only in BASE. The other applications either contain a fixed set of tools or rely on other analysis tools installed on the user's local computer.

Most of the systems do not follow basic paradigms of software development and lack object-oriented architectures and structured programming interfaces. Without such interfaces, software is hard to extend with required functionality. As a result, communication interfaces for interchanging expression data and annotations with other systems are hard to implement. To add supplementary interfaces requires to restructure the software completely.

The same holds true for full support for the complete MAGE-ML format. Some provide partial implementations restricted to data export. Enhancing the MAGE-ML capabilities of such software to the level of full support is impossible without major re-implementation of their storage-backend, because not all annotation data encoded in MAGE-ML can be persistently stored in their database models. Changing the database-model of a software is a very deep intervention in the system and may require changes throughout many other components of the software.

The proposition to implement a MAGE-compliant system from scratch appears sound, given the large extent of required reformations and the consequences of having to re-structure large portions of the software. Provided existing code, such as R and Bioconductor and existing communication infrastructure can be used, the ratio of expenditure of human labor and the adequacy of the system building a well-designed architecture from scratch can be much lower than refurbishing an inappropriate system.

CHAPTER 6

Object-Oriented System Design

Requirement analysis delivers a set of specifications of a desired system. The specifications consist of UML use-case diagrams, lists of analysis functions required and textual natural language descriptions of the desired functionality. As a look at the existing systems has shown, there are many alternative ways to design a microarray storage and analysis system.

The main development paradigms were chosen from the point of view of the developer of the software: the functionality should be modular, and by that, allow to start out with a small set of functions for a prototype and then to add additional functionality afterwards. A modular system has advantages over a monolithic system in being easier to test and maintain.

The system was therefore designed using an object-oriented approach. This involves to use object oriented methods during each step of the design process. Classes of the application logic, persistent classes, and the software component architecture were defined using UML-based design tools.

6.1 Architecture

A so called three-tier architecture is commonly used in software development today (Shaw and Garlan (1996), see also Section 4.7). This approach applies mainly to large distributed software systems involving a database storage component for data. The three-tier approach divides in three layers or tiers:

The database tier or backend layer provides mechanism for storage and retrieval of the data. This layer is often implemented using a database management system, but could also be built using files, especially XML-documents. Often an abstraction layer like ODBC is used to encapsulate the database.

The application tier or business layer contains the actual application logic and also distributes data between the database and the clients.

The presentation layer is responsible for interacting with the user of the software. It receives user input, interacts with the application tier and presents the results to the user.

An important principle of general multiple-tier architectures is, that all communication passes linearly through the layers. There exist no short-cuts to access the database directly from the presentation layer. Structured interfaces have to be provided by every layer to communicate with each other.

The three-tier approach, as presented here, has some limitations, and thus has to be modified and extended into a multi-tier approach. Not only has the information of the centralized server to be sent to distributed clients. Additionally, it is stated in the specification of EMMA2, that a bidirectional communication with other software is required. Therefore, it was decided to provide the complete functionality of the business logic to external applications via an extra layer. This intermediate layer communicates with the application tier and passes objects and messages to other applications.

Database invocation is based on queries and not on message passing. Direct invocation of underlying relational database can pose problems within an object-oriented approach. The table structures might be very different from the application logic. Moreover, additional documentation is required on the database structures. Thus, using a simple relational database management system (RDBMS) would result in leaving object-oriented design at this point. Object oriented database systems do not have these drawbacks. But there could arise other difficulties for time critical applications, which would benefit from SQL like queries. A suitable combination consists of an object-oriented database management system based on an RDBMS (Alagic, 1989). To provide object-oriented features, an RDBMS can be encapsulated in an object-oriented abstraction layer, still allowing low-level access to SQL-queries for a small number of time-critical operations (Blaha and Premerlani, 1998).

The previous considerations result in the following modified multi-tier architecture for the specified system (see Figure 6.1 on page 90):

The backend layer consists of a relational database management system together with an object-relational mapping, which should provide a structured interface, accessed by the other layers.

The application layer provides the necessary application logic. It also provides interfaces to other applications which are used as embedded applications for the purpose of performing computational tasks.

The communication layer provides a bidirectional interface to other software applications for data integration. The complete interface of the backend layer is exposed by this layer. This allows other software to transparently integrate microarray data into their application logic. Additionally, a selected public set of methods from the application layer can be exposed by the interface. The application layer will have to communicate also with the communication layer, whenever it needs data from external programs for data analysis.

The presentation layer will perform user interaction tasks. As such, it has to communicate solely with the application layer and the communication layer. Accessing the communication layer is essential, only if the user requests information not stored in the system but linked to external data-sources.

There can be diverse presentation-interfaces for the data. As an example, the presentation layer provides web-pages for human-computer interaction, or alternatively a graphical user interface (not implemented in this system), and a machine-machine interface for other software applications. These presentation-modes will have a distinct set of functionality while other functionality is shared between them. Examples of shared functionality independent of the mode presentation are validation and safety checks of user input, retrieval of data by certain criteria, and storing data. Therefore, all presentation layers use common functionality which is implemented in the facade layer (Gamma, 2004).

The application and communication layer are located on the same level within the hierarchy. These layers have to communicate with each other and with the backend layer. While the application layer communicates with the presentation layer it transparently encapsulates the communication layer against the presentation layer. Thus, for the presentation layer the communication functionality appears as normal functions of the application layer.

Although the communication layer plays an important role for data integration, in some cases, communication might not be required or be technically unavailable. This implies, that the system should not rely on the continuous presence of external data-sources, and be fault tolerant as to provide autonomous operation in absence of external data.

6.2 Object Model

The specification of the object model serves the purpose of defining the functional entities of a software system and how they interact. In object-oriented design this serves the purpose of breaking down the functionality of the system into smaller parts, the classes, and to define a functional hierarchy on them, using inheritance relations. Interactions between instances of the classes can be defined. It is assumed, that interacting objects exchange information only by invoking methods of other objects.

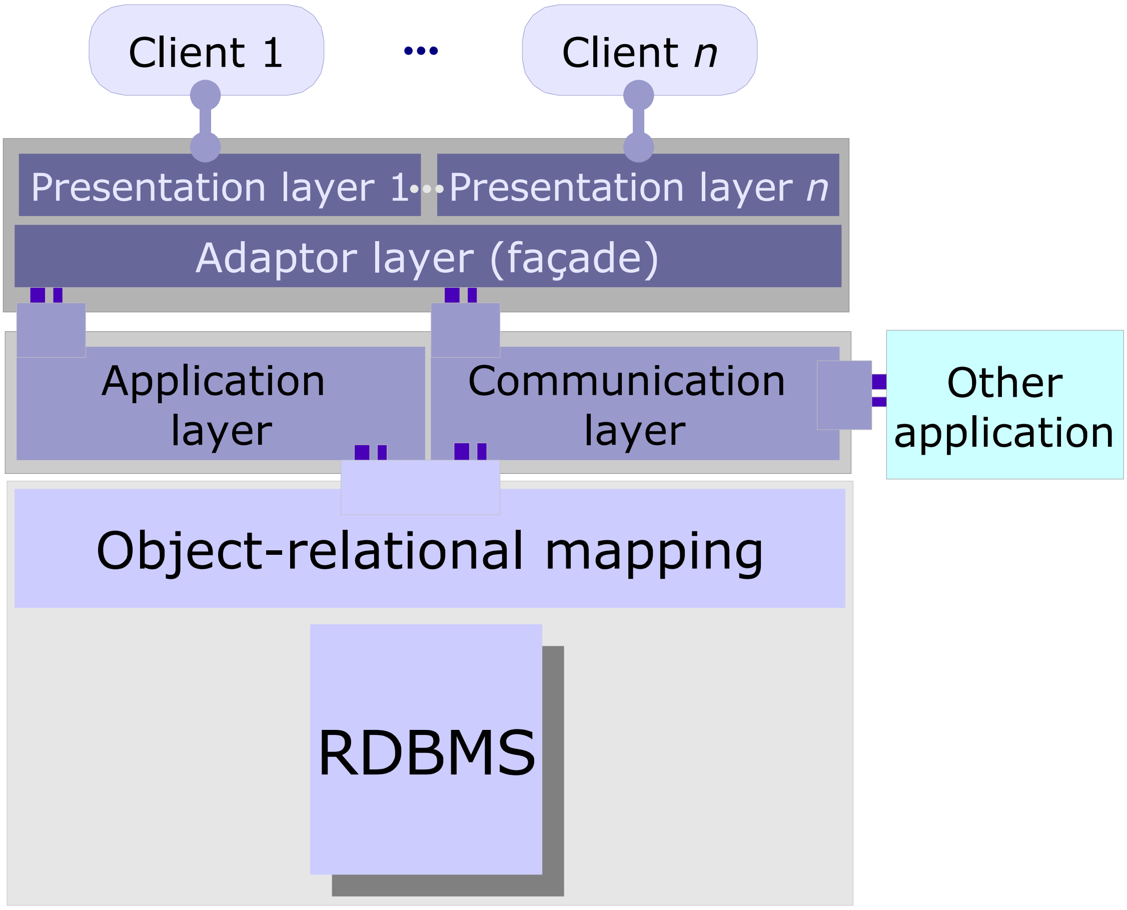


Figure 6.1: Proposed multi-tier architecture for the specified system. RDBS denotes the relational database backend; it is presented to the middle tier by its object-oriented API. The middle tier is split into the application layer which provides core functionality and the communication layer, which is used for interchanging data with other applications. The topmost tier consist of the facade and a collection of presentation modules. Client 1 . . . Client n denote different types of clients (e.g. web-interface, command-line-interfaces etc.).

From the perspective of the application-logic it can be concluded, that the object-model can be divided into two types of classes: persistent and non-persistent classes. As the persistent classes will be mapped to objects, anyway, from the point of view of theuser of the API, there is no difference between persistence and non-persistence classes; persistence is handled transparently. There is still a difference for the design process, because persistence-classes have to be mapped to a database model.

6.2.1 MAGE-OM Classes as Core Persistence Model

For the database component a database scheme is required to model the content and its logical interconnection. This model has to be able to store information related to microarray experiments and their annotations. In accordance with the requirements of MIAME-compliant annotations and to process data in MAGE-ML format, it was decided to use the complete MAGE object model as the object model for the persistence classes of EMMA2. A MAGE-OM based system is capable of storing data from any valid MAGE-ML file. MAGE-OM is also developed using UML class-diagrams which allow for automated transformation into a database scheme.

The original MAGE object model was designed solely to store and transfer data. There are only rudimentary classes which represent functionality of data-driven analysis methods and access control. Nevertheless, they require persistence, too, because it is necessary to record information of analysis steps taken. Classes representing access control functionality are also needed within the persistence model to provide mandatory access control.

Following the preceding considerations, the object model of persistence classes is divided into three portions:

- the core data-representation classes based on MAGE-OM,
- extensions not contained in MAGE-OM handling data analysis,
- extensions to MAGE-OM governing access control.

MAGE-OM has been describe in detail in Section 4.1.2. The extension classes containing additional application logic are specified using UML class-diagrams as well. They are further described in the following sections.

6.2.2 Access Control Model

As stated in section 5.3.1, providing a secure level of access control is essential for data privacy and thereby acceptance of the software system. A software component providing access control has to be integrated into the described framework. An access control component has to fulfill two authorization tasks. First, access to the individual repository or project has to be controlled, and second, access to each individual object of interest within this repository has to be controlled on a per-user basis. Additionally, it should be possible to restrict the available functionality for each user in such a way that users can be excluded from performing computationally intensive or possibly dangerous tasks, for example deleting objects or adding new functionality.

A role-based approach was chosen to accomplish this tasks. As such it has several advantages over a simple per-user approach. While providing a high level of granularity to control access, it also keeps the system manageable by providing sets of predefined access rights for objects. A role provides a set of rights within the domain. By defining a role of a user, the possible actions a user can perform within a system are defined.

A primary problem with role-based access control systems is the dynamic nature of the objects. Objects may be created, modified and destroyed during normal operation of the system. In order to keep up with the dynamics of the changing repository, either the access rights of every role would have to be constantly updated or could only work on classes of objects and not on individual instances. Both approaches seem rather unsatisfying, as the former makes the role contents unpredictable and involves a large management overhead, and the latter does not fulfill the requirements.

Access control lists provide a convenient way of customizing the access rights of individual users on a specific object. Thus it was decided to expand the role-based approach by adding user and group-based access control lists (ACL) to the system. The ACLs provide the ability to specify access rights for individual objects in the repository. Users can be organized into intersecting groups, having their own ACLs. As a user can be assigned to only a single role within the role-based access model, this concept helps to organize access rights within large projects.

It is necessary to define the set of access permissions controlled by the ACLs. The permission set consists of the following actions:

Read The user may look at the object and view the information provided, like name, descriptions, individual data, etc.

Edit The user may change values, names and descriptions.

Reference The user may re-use this object (mostly important for arrays and array layouts) and link to it. For example, an array can be referenced in another experiment.

Delete The user may remove the object permanently.

View permissions The user may view the ACLs assigned to this object.

Change permissions The user may alter the permission settings by adding and removing ACLs.

Each permission can be set to one out of three values, which can be:

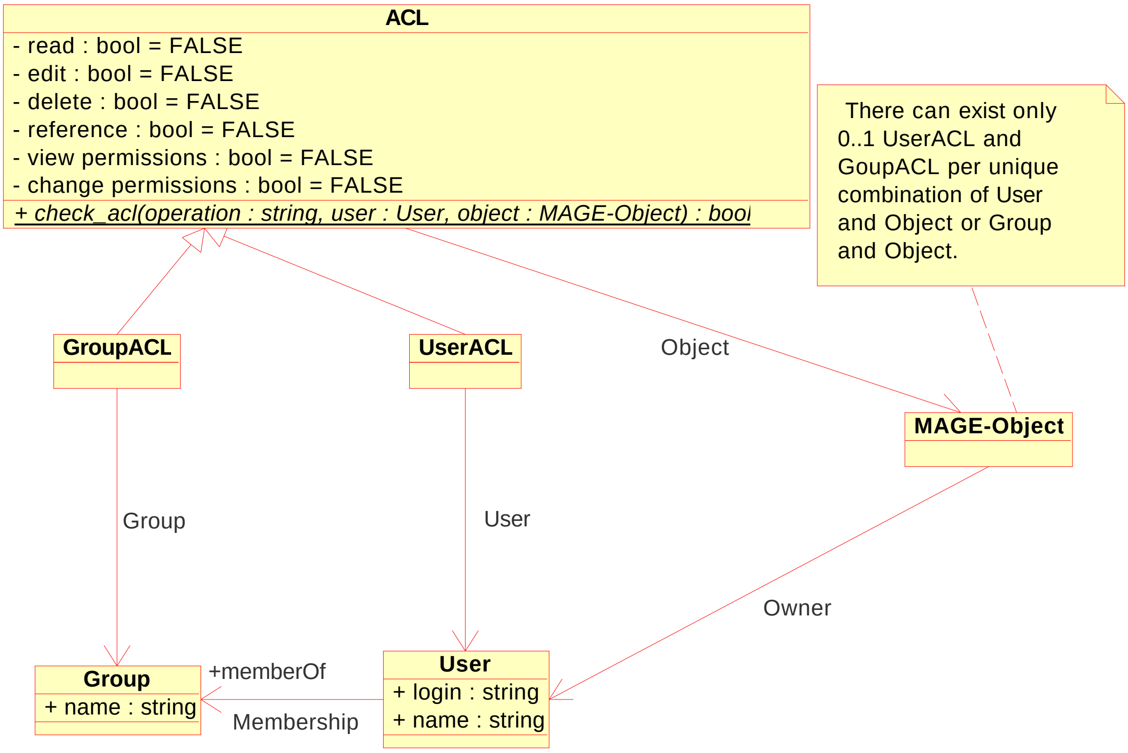

ACL
- read : bool = FALSE
- edit : bool = FALSE
- delete : bool = FALSE
- reference : bool = FALSE
- view permissions : bool = FALSE
- change permissions : bool = FALSE
+ check_acl(operation : string, user : User, object : MAGE-Object) : bool
There can exist only 0..1 UserACL and GoupACL per unique combination of User and Object or Group and Object.
GroupACL
UserACL
Object
MAGE-Object
Group
User
Owner
Group
+ name : string
+memberOf
Membership
User
+ login : string
+ name : string

Figure 6.2: Simplified UML-model of the database classes representing the access control system. The user and group ACL classes represent the access control lists for users and groups of users and are both derived from the ACL superclass. The ACL class can control access to any object within the MAGE-OM.

Yes Grant the permission, equivalent to a logical TRUE value.

No Forbid the action, never allow to perform it, equivalent to a logical FALSE value.

Undefined Do not grant or forbid the action, ignore this setting equivalent to a logical missing value.

A combination of all permissions with an assigned value, a user or group, and an object constitutes a complete ACL. The resulting permission is derived from a combination of all ACLs assigned to the user for the given object; ACLs can be either directly assigned to a user or assigned by group memberships. As a result, many ACLs may apply for a given combination of user and object. In case of a conflict, the permissions are computed from all ACLs using a logical AND operation. Undefined values are simply ignored; if there are only undefined values the *default access policy* is used.

The default access policy needs to be defined for each action in the action set. If the permission settings for an action consist only of missing values or if there are no applicable ACLs, the default ACL is substituted. For reasons of data security, without a specific default policy, a deny-by-default policy is applied. This results in an often desired behavior where a user cannot perform any action until it is explicitly allowed.

6.2.3 Data-Analysis Model for PlugIns

A modular analysis system, that is flexible to combine different analysis algorithms, is a key component in a microarray analysis system. Data structures representing the different functional elements of an analysis process have been included into the data model. There is no representation of specific algorithms or implementations within the data model, but representations of generic functions operating on the MAGE datastructures.

The representation of analysis functionality is divided into three classes which correspond to program code in the backend of EMMA2:

Functions are the basic building blocks of the analysis process. Functions represent the transformation algorithm implemented in a programming language. A function can be implemented in either R, Perl or as an arbitrary executable program. Functions can have parameters controlling the behavior of the function, which need to be set by the user. The data types of parameters are also stored within the database. So called import and export functions form a special class of functions which serve as a means to store the results of an analysis step in the database.

Tools can be seen as parameterized functions with defined values for the parameters. As the user needs to set actual values for the parameters of a function, the parameters have to be stored in the database. There is a special type of tool, which serves as a means to combine several analysis methods in a single consecutive execution. These tools are termed 'Queue' within the model (in the presentation to the user they are called 'pipelines' in resemblance of an assembly-pipeline). Queues may contain a sequence of other tools and also other queues, which are then executed consecutively. A queue is independent of the actual data set but often restricted to a specific data type, which is determined by the first function in the pipeline.

Jobs are a combination of tools and actual data sets to be analyzed. A job can be executed via a scheduling mechanism to dispatch jobs to a multi-host compute cluster. Jobs reference the designated input data and the output they produce. They can be assigned to an experiment if appropriate.

Three different types of functions can be specified which serve different purposes:

General analysis functions are R or Perl functions or binary executables performing computations.

Export-functions get additional annotation information from MAGE-OM, BRIDGE or a web-service. This can be for example pathway information from GenDB.

Writer functions put back data into the MAGE-OM. They can also be used to store images or graphics generated within a pipeline.

Importer functions put data into the database, like reading in raw-data files or array-layout definitions. Unlike writer functions, they do not require data from a previous computation.

Data structures were designed to represent the analysis modules within the database. The corresponding classes serve to set up pipelines and parameters of the contained functions. An overview of the extension classes that supplement MAGE-OM is given in UML in Figure 6.3 on the next page.

The ability to add new functions to the system is a key feature of the data analysis system. As such it allows to add functions which have a behavior not known during the implementation phase. To avoid meaningless pipelines, a type system has been included in the model. An example for the application of a type-systemcan be given as follows: a normalization function operates on the measured data that have been imported from the image analysis software; the application of a normalization function on already normalized data or on the results of a significance test would not be sensible. As a consequence, creating such a pipeline configurations has to be prevented.

The first step for the definition of a type system is to identify the types. From a computational perspective, the data type of microarray data is represented by multidimensional arrays of numeric and factorial values. The atomic datatypes and also the dimensionality of the arrays do not provide a suitable type system for

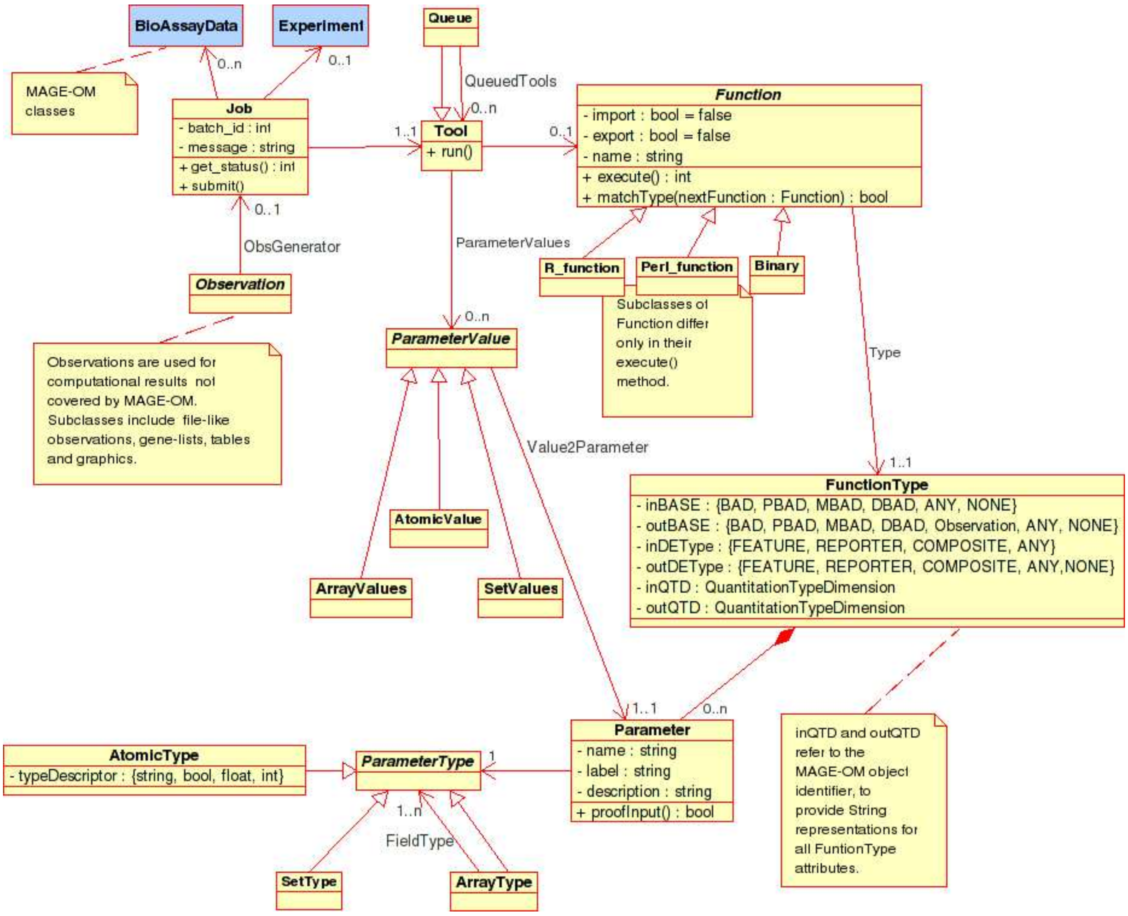

BioAssayData
Experiment
Queue
MAGE-OM classes
0..n
0..1
QueuedTools
0..n
Job
- batch_id : int
- message : string
+ get_status() : int
+ submit()
1..1
Tool
+ run()
0..1
Function
- import : bool = false
- export : bool = false
- name : string
+ execute() : int
+ matchType(nextFunction : Function) : bool
0..1
ObsGenerator
ParameterValues
R_function
Perl_function
Binary
Subclasses of Function differ only in their execute() method.
Observation
Observations are used for computational results not covered by MAGE-OM. Subclasses include file-like observations, gene-lists, tables and graphics.
0..n
ParameterValue
Type
Value2Parameter
1..1
FunctionType
- inBASE : {BAD, PBAD, MBAD, DBAD, ANY, NONE}
- outBASE : {BAD, PBAD, MBAD, DBAD, Observation, ANY, NONE}
- inDEType : {FEATURE, REPORTER, COMPOSITE, ANY}
- outDEType : {FEATURE, REPORTER, COMPOSITE, ANY,NONE}
- inQTD : QuantitationTypeDimension
- outQTD : QuantitationTypeDimension
AtomicValue
ArrayValues
SetValues
1..1
0..n
Parameter
- name : string
- label : string
- description : string
+ proofInput() : bool
inQTD and outQTD refer to the MAGE-OM object identifier, to provide String representations for all FuntionType attributes.
AtomicType
- typeDescriptor : {string, bool, float, int}
ParameterType
1
1..n
FieldType
SetType
ArrayType

Figure 6.3: Simplified UML diagram of the supplementary persistence classes of the analysis model. The diagram depicts the core classes Job, Tool, and Function together with derived classes. The observation class is introduced to store all resulting data that do not fit in MAGE-OM. The diagram is simplified for clarity and readability: only the most important subclasses are depicted, and not all associations are named and show their cardinalities.

functions as they do not provide a logical classification of the data types. A numeric array of given size might be the result of many different analysis algorithms.

In particular, the dimension indicating the number of spots, genes or other design elements of a microarray should be neglected by the type system, as a function like a normalization function should be applicable to datasets regardless of the specific array design used. On the other hand, the choice of a normalization function might reflect the technological platform of the array, because multi-channel microarrays and one-channel microarrays require different processing. In addition, different image quantification softwares produce diverse quantitation types.

It is required to base the type system on a higher-level of logical annotation, because MAGE-OM plus extensions provides for the complete view of the world for the software. Any analysis functions has to operate on the datastructures which exist within the model. Therefore, the possible types of the analysis functions must reflect the data-structures in MAGE as close as possible.

Four classes representing steps of the data analysis serve as input and output of data analysis functions. These classes are descendants of the BioAssayData class and are used to determine the basic data type of a function in Analysis model:

PhysicalBioAssayData(PBAD) represent data measured with hardware equipment like scanners. The only possible data type found with PBAD is images. Image analysis is currently not incorporated into EMMA pipelines but carried out by external applications. The presence of the PBAD data type provides the possibility to include image analysis directly into a pipeline.

MeasuredBioAssayData (MBAD) are data resulting directly from a image quantification software. MBAD consist of tabular data containing raw intensities and quality statistics for each feature on the array. These data have to be further processed by normalization.

DerivedBioAssayData (DBAD) represent the output of a transformation process. A transformation process takes as input MeasuredBioAssayData or DerivedBioAssayData and creates one or more numerical datasets as output which are of type DBAD. Normalization is an example for a transformation from MBAD to DBAD. Other functions like significance tests operate on normalized data which are of type DBAD and give a table of significance statistics also of type DBAD for each gene.

BioAssayDataClusters (BADC) are the results of a higher-level analysis which provide a grouping of individual design elements or also microarrays on the basis of MeasuredBioAssayData or DerivedBioAssayData. A typical function is a cluster analysis algorithm calculating a grouping of data into clusters. Also, the results of a classification algorithm producing a mapping of design elements into disjoint classes may be represented by this data type.

The BioAssayData classes are not complete, as specific analysis tasks require more sophisticated data-types than found in MAGE; images, clickable maps, files, and lists of gene names are the most frequently required examples. The Observation class hierarchy is introduced to set forth such supplementary data types. It is also suited to eventually derive further sub-classes for new analysis methods without affecting the core MAGE-OM classes.

There are more preconditions that can help to classify the type of a data matrix, in particular the type of rows in a data matrix. As an example, the rows in a quantification table from an image analysis software represent measured values for each individual spot on the microarray. Also, normalization produces normalized intensities or intensity ratios for each spot, represented by the Feature class in MAGE-OM. Individual spots may be seen as repeated measurements for a common polymer of nucleotides physically present on the arrays, called Reporter. Reporters may be further grouped into a logical sequence, representing a genomic region. A common example are genes represented by different oligonucleotide sequences, which are subsequences of its coding region. These logical sequences are called CompositeSequence in MAGE terminology.

A normalization function solely operates on the Feature-level of the data matrix, while a function which computes an expected expression value, e.g. the mean, over all replicates for a sequence operates on Features and its output is based on the Reporter or CompositeSequence assignment of the Features. In conclusion, the DesignElement type of a function may be one of *Feature*, *Reporter*, *CompositeSequence* or *any*. The type *any* may be applicable for functions which are indifferent on the type of design element present, such as plotting functions.

The third categorization of data in the analysis process relies on the data types of the columns in a data matrix, called QuantitationType. In MAGE it is mandatory to assing a QuantitationTypeDimension to a data set defining the type of measurements found in the columns. For a measured data set, the QuantitationTypes correspond to the column headers of the quantification table. An example for quantitation types in measured data are the foreground intensity and background intensity of the estimated spot intensities. The names of the quantitation types may vary between different image quantification software and analysis functions, and are not known to the implementation of the analysis system. Therefore, the quantitation types have to be specified for each function present in the database during run-time of the system.

In summary, the type of each function in EMMA2 is defined by tuple of the input and output type, in such a way that each function constitutes a mapping:

$$f : (B_i, D_i, Q_i) \longmapsto (B_o, D_o, Q_o) \tag{6.1}$$

where $B \in \{\mathrm{BAD}, \mathrm{PBAD}, \mathrm{MBAD}, \mathrm{DBAD}, \mathrm{BADC}\}$ is the class of the BioAssayData in MAGE, $D \in \{\mathrm{Feature}, \mathrm{Reporter}, \mathrm{CompositeSequence}, \mathrm{Any}\}$ the type of DesignElement and Q the QuantitationTypeDimension in MAGE. The type of a function is modeled in the supplementary class

CHAPTER 7

Implementation

In the previous Chapters, the required functionality of EMMA2 has been specified, and from that the structural components and their interactions have been derived. Now, it is time to assemble them into an operational piece of software. This can be accomplished by using programming languages, code libraries and data-base management systems. A prototype version of the software has been built from scratch to refine and improve the specification and design.

An overview of the component structure of EMMA2 is given in Figure 7.1 on the following page. The implementation and adaptation of these data-structures, algorithms and novel visualization methods are explicated in the following Chapter.

7.1 Choice of Core Development Tools

There are manifold development environments, database tools, and programming languages to support the process of implementing software. A major criterion for selecting these tools is their reliability and efficiency; another is free availability to guarantee the system is distributable and extensible within the academic community for everyone. For microarrays, additional problems such as dealing with very large and noisy datasets and very complex data structures need to be addressed.

7.1.1 Database Backend

The named aspects of efficiency and reliability are of particular relevance for the choice of the database backend to store persistent data. Database tools should support the implementation of the architecture and data structures developed

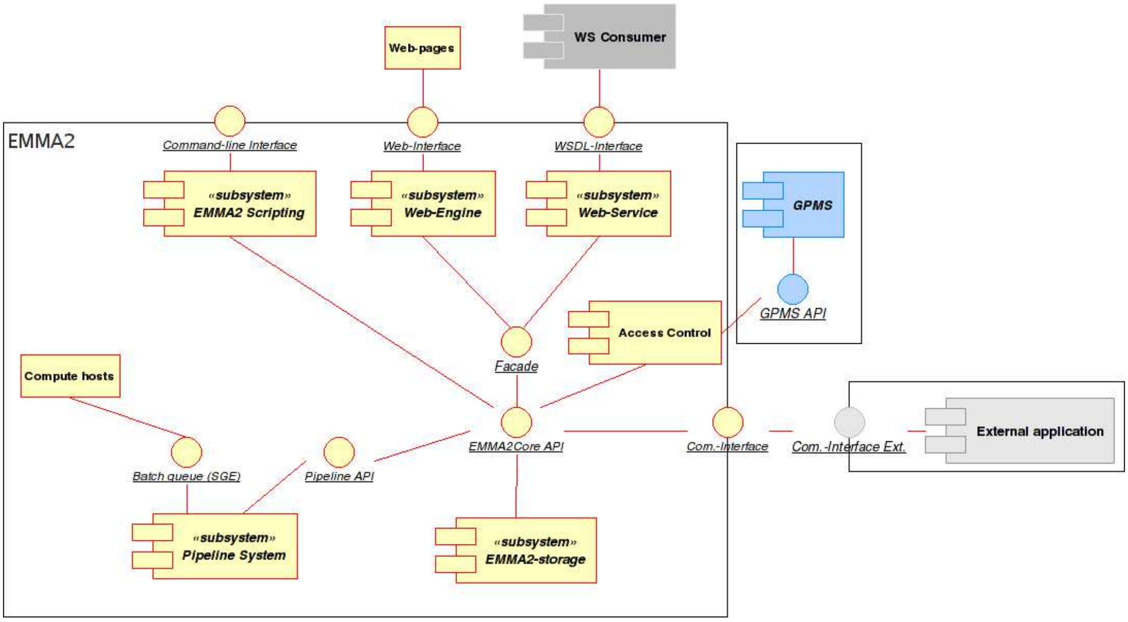


Figure 7.1: The component model of EMMA2. An UML component graph provides an overview of functional subsystems and their communication by using defined interfaces.

during the design phase. Therefore, it is favorable to utilize an object-oriented database management system that can directly resemble the object-oriented persistence classes, and also supports client-server applications. Unfortunately, there are no efficient open-source implementations of such systems.

On the other hand, many relational database management systems (RDBMS) exist, such as MySQL, MaxDB, Microsoft SQL Server, Oracle or PostgreSQL, some of which are distributed under open source licenses. At present one of the most widely used open-source RDBMS is MySQL. Therefore, the current implementation uses MySQL as RDBMS in connection with a object-relational mapping providing persistent objects. Wherever possible, only the subset of the SQL language compatible with MySQL, PostgreSQL and Oracle is used. This provides the possibility of porting the application to a different RDBMS without changing the backend code.

7.1.2 Programming Languages and Tools

The choice of appropriate programming languages depends on many criteria. In the case of the EMMA2 software, the most important considerations support of the object-oriented development paradigm, the support of a client-server architecture and internet communication, as well as the availability and extensibility of existing code-libraries for microarray data analysis. Furthermore, efficiency and accuracy of numerical operations for data-analysis, XML-support, the availability of interfaces to RDBMS are relevant aspects, and the ability to generate interactive web-applications. Finally, the employed programming tools and code libraries should be available under free open-source licenses to allow for easy code extensions.

Perl meets most of the above criteria and is especially suited for database connectivity and web-applications (Wall *et al.*, 2000). On the other hand, the R/Bioconductor environment includes a much larger number of implementations of data-analysis algorithms; it is also more efficient in programming numerical operations as it allows for direct operations on vectors and matrices.

The RSPerl library provides a means to transfer data between both programming languages. Thus, the current implementation uses Perl for most components and R for efficient data-analysis. To further enrich the interactivity of web-pages, Java-applets are used. For the task of automated code-generation and for generation of APIs and documentation from UML specifications a transformations language (XSLT) is used; this process is detailed later in this Section (7.1.4).

7.1.3 Object-Relational Mapping With O2DBI

The database backend layer has to provide structured access to the underlying data repository for structured queries and creation of new datasets. The data-model contains a large and highly connected set of persistent classes, constituting the repository of the application. These classes have to be represented in a relational database scheme. Therefore, a mapping of objects on relational tables is required. In order to provide the application programmer with a structured interface to the stored data, an API in the target programming language (Perl in this case) has to be implemented.

An API-code and the corresponding relational database scheme could be implemented by hand. As a downside, this would necessitate manual writing and adjusting of source-code for every change of the data model. The data-model developed in the desing phase is not static but is likely to be modified during development cycles. Because of this, a completely automated process of creating the object-relational mapping would serve for flexibility and speed up of the implementation process.

The O2DBI software is a system that can be used to create object-relational mapping interfaces from a definition of classes, their attributes and methods. It creates database table descriptions in SQL for RDBMS and the corresponding API code and documentation. O2DBI can provide class inheritance and auto-generated accessory methods for each attribute of a class. O2DBI has an XML-based class definition format, and it also provides a graphical user interface to define the object model.

7.1.4 Mapping of MAGE-OM Classes on O2DBI Classes using XML Transformations

The object modeling format of O2DBI is not directly compatible with the XMI format in which the MAGE-OM is defined and which is often used by UML modeling tools. While the UML class model encloses class hierarchies class attributes, associations, and methods, O2DBI resembles the class structure of Perl more closely. To work out some of the core differences of O2DBI and UML class models, some examples will be given.

UML makes a distinction between attributes and associations; attributes can have only atomic types while associations connect two classes. In O2DBI as in Perl there are only attributes of a class which are of a defined data-type which could be atomic or reference to another class. In UML associations can be uni-directional or un-directed, where in O2DBI, there are only uni-directional attributes. Multiplicity constraints for attributes and associations can be defined in UML, while in O2DBI, one can define one-to-one or one-to-many associations by using Perl arrays. Backward directed multiplicity constraints cannot be directly resembled in O2DBI. Instead, O2DBI provides means of using features like the indexing functions of the database engine. Unique and non-unique indices can be defined on single and multiple attributes of a class.

UML class diagrams and O2DBI definitions are functionally not equivalent, hence the process of implementing an UML class diagram with O2DBI requires a mapping step, leading from a logical model representation to a source code representation. This representation will also have features which are closer to the programming language and the employed DBMS. UML and O2DBI use XML files for model specification, and the transformation process can be implemented by defining a transformation function from UML to O2DBI. The transformation function should also make use of the optimizations the DBMS allows, wherever feasible.

The following rules have been applied to bridge between features having no direct representation in the other system:

- UML-datatypes are mapped to their closest O2DBI atomic datatypes.
- UML-associations are tranformed into O2DBI-attributes of the required O2DBI-class.
- Forward multiplicity is transformed into array-valued attributes, if the upper multiplicity constraint is greater than 1, and to a class reference attribute otherwise.
- Backward and forward multiplicity is translated into unique indices, if both ends of the association have multiplicity 1. This implies that a exclusive one-two-one relation exists between all objects of these classes.
- If backward multiplicity is greater than zero, a non-unique index is created. This is the case when class A has an association on another class B and should refer to exactly one object of class B, while the object of class B can be referenced by many instances of class A. In this case, there will be many references from class A to class B making a unique index impossible.
- Bidirectional UML associations are mapped into two O2DBI attributes.

The transformation rules could in principle be applied manually by defining the O2DBI datastructures. As this is inconvenient for the large number of classes and associations in MAGE-OM, an automated transformation was implemented. The eXtensible Stylesheet Language for Transformations (XSLT)[1] is a programming language optimally suited for this purpose. It combines completeness and easy recursive function definition within a functional programming language operating on XML tree structures. For each relevant element in the XMI document, a transformation rule was defined giving O2DBI XML elements as output.

The transformation algorithm has been applied to the unmodified MAGE-OM, the access control model, and the data analysis classes as comprehensive persistence classes set forth in Section 6.2. Documentation within the UML-model has been converted into the corresponding O2DBI documentation. The resulting two XML-files have been merged to become the primary core model of the application. The Perl-classes and the backend code has been automatically generated using the O2DBI code generator. The code generation process results in the backend Perl-classes, the data scheme as SQL table definitions and the documentation files. The implementation of code generation is a multi-step process which is fully automated; an overview is given in Figure 7.2. During the automatic transformation, the algorithms also create the database indexing structure for single associations as described before, such that an immediately functional database model is derived. To speed-up frequent queries, the SQL definition has been complemented with manually defined optimizing index structures for the most common and most

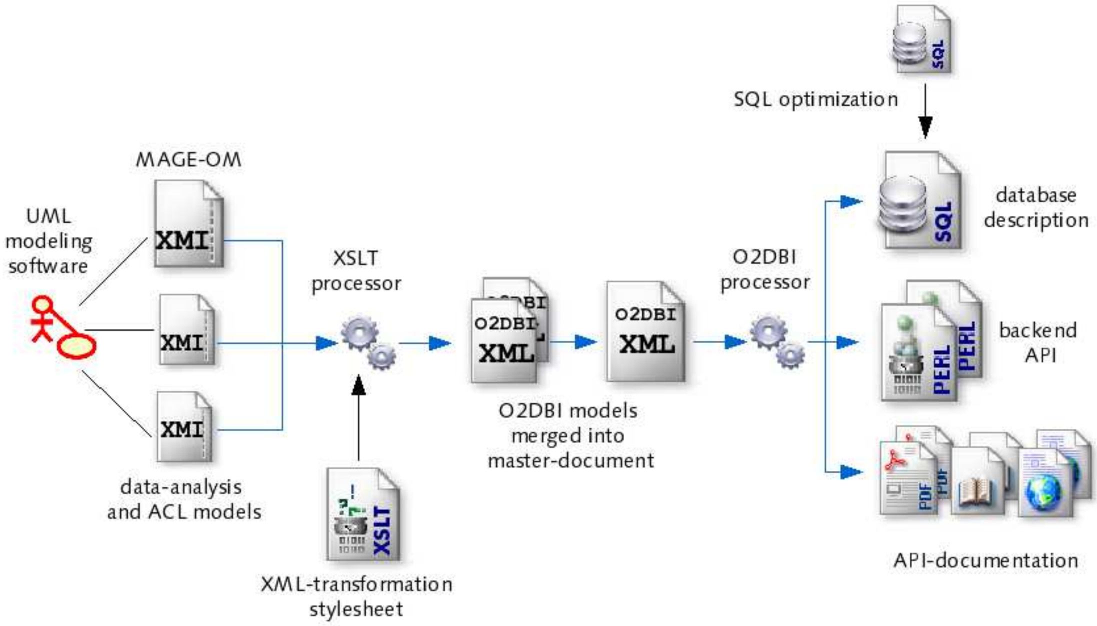


Figure 7.2: Overview on the code generation process of the EMMA2-API.

[1] http://www.w3.org/TR/xslt

time consuming queries. The supplementary SQL-optimizations are automatically merged into the process.

7.2 Web-Interface

A web-interface provides a high level of compatibility with heterogeneous computing environments. The only software necessary to interact with a web-interface is a web-browser. No deployment of client software is required.

The web-interface of EMMA2 is implemented in a highly modularized way, to separate the layout of the web-pages from the underlying functionality. Code-generation of web-pages was attempted to create parts of the web-interface automatically from the MAGE object model. A prototype implementation was set up, consisting of automatically generated pages.

As a result, a generated interface was found to be counter-intuitive. The prototype provided a view on the data structures and the possibility to edit the data, but the level of interactivity was low. Therefore, a manually optimized interface was built using dynamic HTML pages with higher interactivity.

7.2.1 The Web-Interface as an Implementation of a Presentation Layer

As the specified software requires interactive exploration of the data by tables, scatter-plots and other visualization techniques, it was decided to provide a manually adapted interface. This interface was implemented on the basis of the automatically generated presentation code, but could be adapted to needs of potential users. During the implementation of the interface, an iterative process of building prototypes was used. Prototype web-pages with minimal functionality were presented to potential users of the system to get feedback on the layout and further proposals on required functionality.

Each functional component of the web-interface is divided into three sub-components. The first sub-component implements the actual screen-layout of the presentation using HTML-template files. The screen-layout of the interface is further divided into cascaded templates defining the overall layout and navigational items, common to all pages.

The second component provides interactive functionality of the presentation layer, like filling the pages with content, receiving and filtering user input. These modules are all subclasses of the class *BasicTemplate* from which they inherit a common set of functions to list objects, view and edit data and to create new data objects. These functions are called by a central dispatcher CGI-script depending on navigational parameters.

To decouple the presentation from the overall structure of the API, this layer does not contain direct calls to the MAGE-OM API. All API calls are implemented within the facade layer. The modules in this layer are all derived from the *BasicTemplateDB* class. Sub-classes provide methods to verify data input from the web-interface including access control restrictions and functions which operate on the API-level to store and retrieve data in complex queries. The functions in the facade layer are called from the BasicTemplate presentation classes.

There exists a single multiplexer CGI-application, called `emma2.cgi`, that handles parameters, database authorization, template and request-handling which are common to all web-server based applications. Extension modules in the presentation layer, which implement the presentation logic, are loaded on demand. Methods from the interface common to all the modules of the presentation layer are called by the multiplexer.

The modular approach makes it easier to develop and deploy new functionality. In particular there is no need to write new cgi-scripts. Instead, a presentation module, an adaptor module and a HTML-template have to be defined implementing the common defined interface as depicted in Figure 7.2.1.

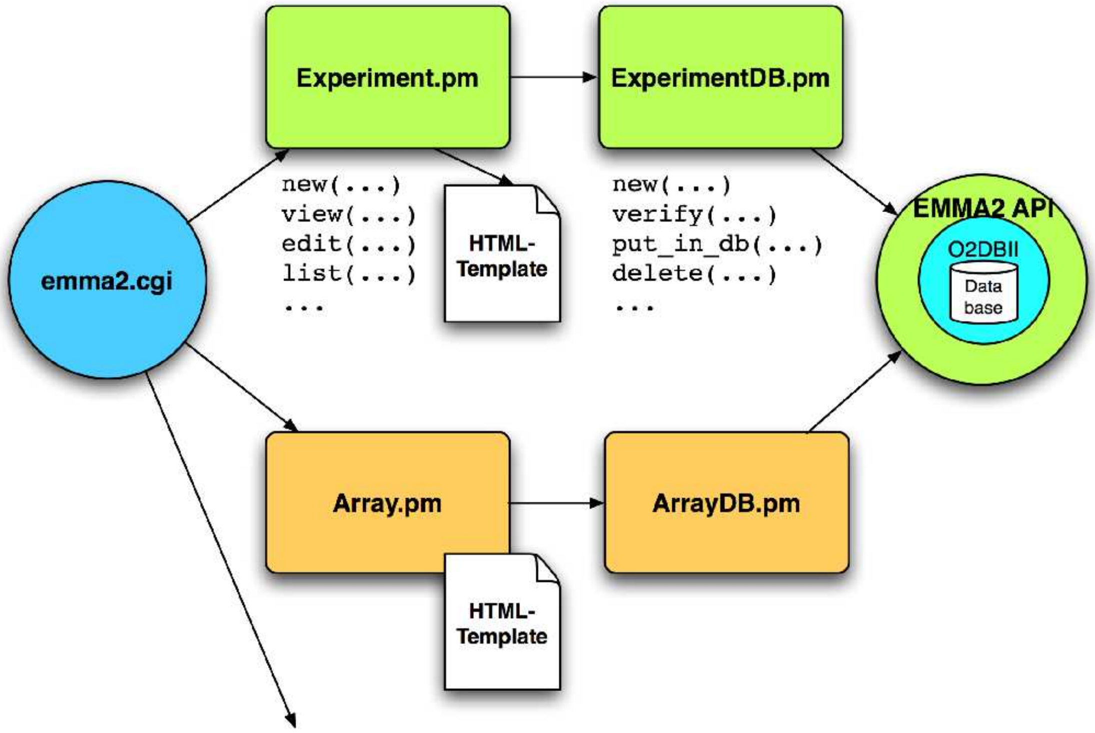
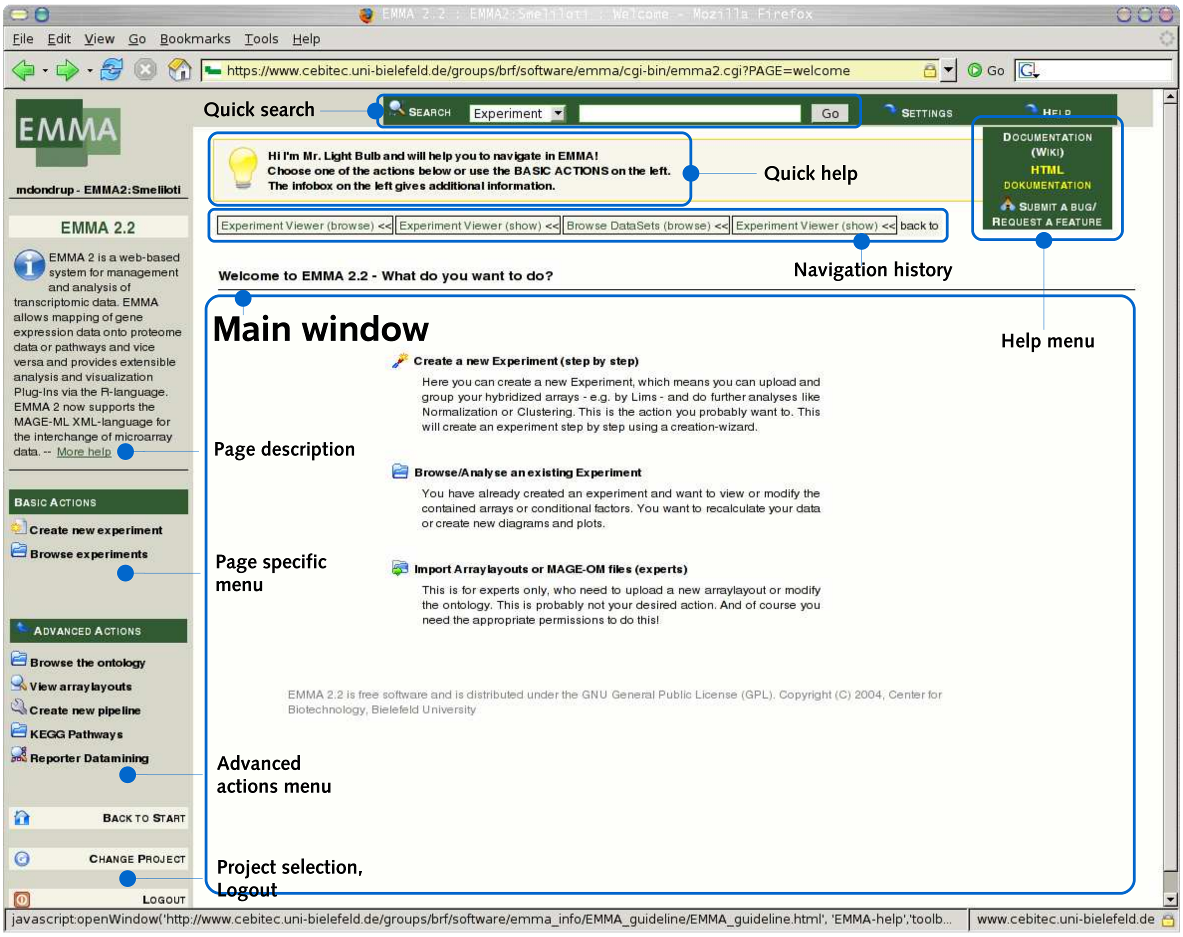


Figure 7.3: Structure of the implementation of the web-interface of EMMA2. The implementation consists of a single multiplexer script, presentation modules (Experiment.pm, Array.pm) which use adaptor modules (ExperimentDB.pm, ArrayDB.pm) to access the EMMA2 API.


EMMA 2.2 : EMMA2:Smeliloti : Welcome - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
https://www.cebitec.uni-bielefeld.de/groups/brf/software/emma/cgi-bin/emma2.cgi?PAGE=welcome
Go
EMMA
mdondrup - EMMA2:Smeliloti
EMMA 2.2
Quick search
SEARCH
Experiment
Go
SETTINGS
HELP
DOCUMENTATION (WIKI)
HTML DOKUMENTATION
SUBMIT A BUG/ REQUEST A FEATURE
Quick help
Experiment Viewer (browse) << Experiment Viewer (show) << Browse DataSets (browse) << Experiment Viewer (show) << back to
Navigation history
Help menu
Main window
Page description
Page specific menu
Advanced actions menu
Project selection, Logout
BASIC ACTIONS
Create new experiment
Browse experiments
ADVANCED ACTIONS
Browse the ontology
View array layouts
Create new pipeline
KEGG Pathways
Reporter Datamining
BACK TO START
CHANGE PROJECT
LOGOUT
www.cebitec.uni-bielefeld.de

EMMA 2 is a web-based system for management and analysis of transcriptomic data. EMMA allows mapping of gene expression data onto proteome data or pathways and vice versa and provides extensible analysis and visualization Plug-Ins via the R-language. EMMA 2 now supports the MAGE-ML XML-language for the interchange of microarray data. -- More help

Hi I'm Mr. Light Bulb and will help you to navigate in EMMA! Choose one of the actions below or use the BASIC ACTIONS on the left. The infobox on the left gives additional information.

Welcome to EMMA 2.2 - What do you want to do?

Create a new Experiment (step by step)

Here you can create a new Experiment, which means you can upload and group your hybridized arrays - e.g. by Lims - and do further analyses like Normalization or Clustering. This is the action you probably want to. This will create an experiment step by step using a creation-wizard.

Browse/Analyse an existing Experiment

You have already created an experiment and want to view or modify the contained arrays or conditional factors. You want to recalculate your data or create new diagrams and plots.

Import Arraylayouts or MAGE-OM files (experts)

This is for experts only, who need to upload a new arraylayout or modify the ontology. This is probably not your desired action. And of course you need the appropriate permissions to do this!

EMMA 2.2 is free software and is distributed under the GNU General Public License (GPL). Copyright (C) 2004, Center for Biotechnology, Bielefeld University

javascript:openWindow('http://www.cebitec.uni-bielefeld.de/groups/brf/software/emma_info/EMMA_guideline/EMMA_guideline.html', 'EMMA-help','toolb...

Figure 7.4: Annotated screenshot of the EMMA2 web-interface.

7.2.2 The ArrayLIMS Interface

The ArrayLIMS system has been implemented to capture the whole laboratory process of a microarray experiment. ArrayLIMS provides web-forms for lab-employees to be used as an electronic laboratory book. It allows to upload and manage protocols for each laboratory step as well as quality control information like images of gel separations.

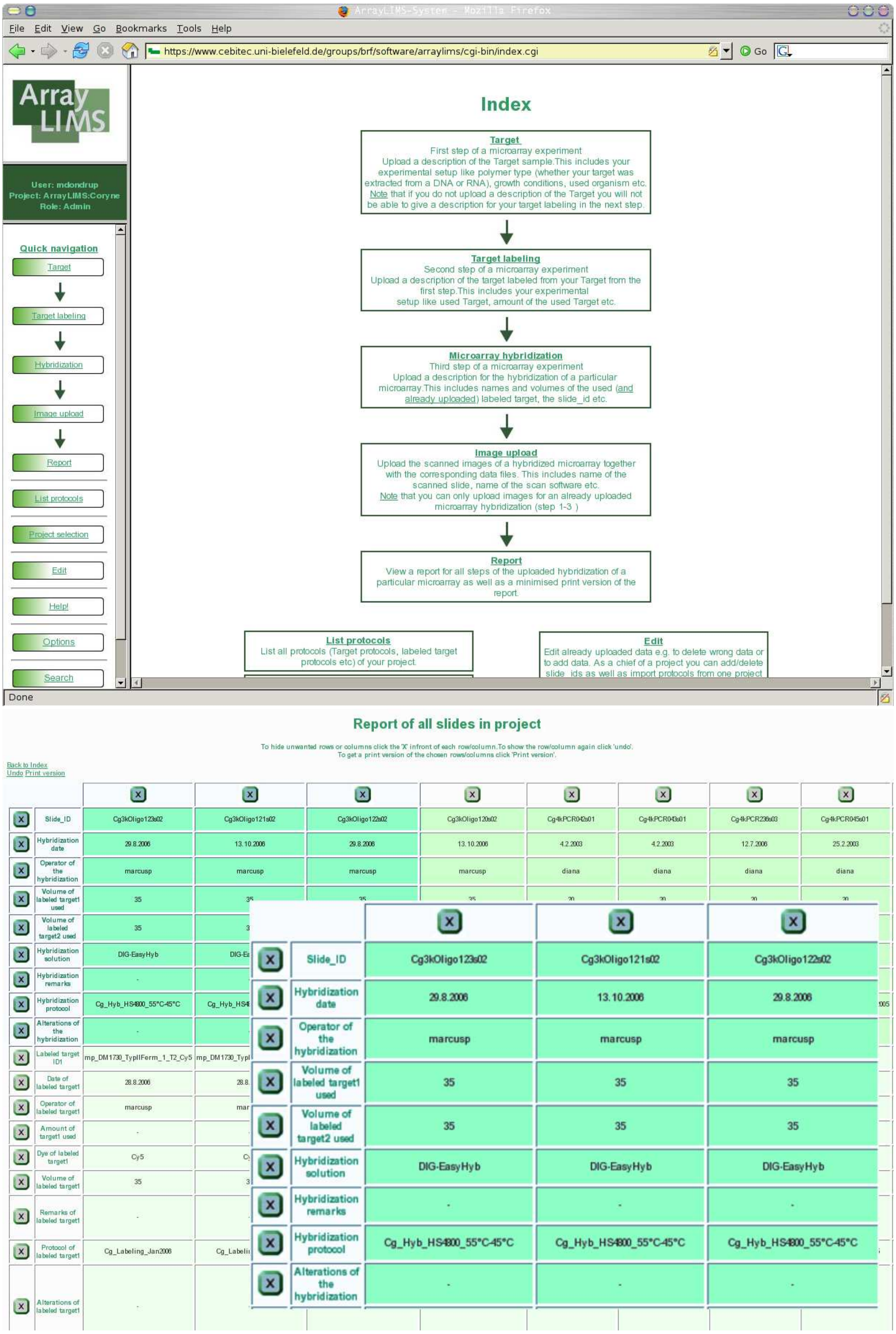
ArrayLIMS has been implemented as standalone application with its own database and web-server. This is due to reasons of data-security where a certain level of redundancy can be tolerated. Raw image data cannot be reproduced from the microarrays; hence it absolutely necessary to store image data in their native format, prior to any successive analysis or transformation. Another reason for building a stand-alone application is the use of ArryaLIMS as a laboratory book. This requires a very minimalistic user interface he web-interface has to be compatible with many web-browsers and laboratory computer systems.

The web-interface of ArrayLIMS resembles a linear workflow of the laboratory process during experiment conductions. Data entry starts with the sample preparation and RNA extraction steps. Growth conditions, species, strain and tissue information can be provided as the first step of the workflow. For two color experiments, it is possible to specify two RNA extractions in a single step.

Protocol steps for producing the labeled targets and their hybridization to the microarray follow. As the final step, image files and data files of different formats can be uploaded and attached to a slide. Each slide can have many image files from multiple scanning procedures. A screenshot of the front-page depicting the workflow is found in Figure 7.5 on the next page.

Additional functions of the web-interface contain summary pages for each microarray and search functions. Laboratory work involves many routine steps with fixed parameters. To support large batches of arrays, ArrayLIMS allows to predefine protocol parameters for faster data upload. Detailed information on the use of the ArrayLIMS web-interface is available in an extensive manual and online help.

After a batch of protocols and hybridizations have been uploaded to the LIMS, data can be transferred to EMMA2 in an import process and are transformed into the corresponding MAGE-OM objects. Collections of arrays can afterwards be assembled into collections by organizing them within experiments.


ArrayLIMS-System - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
https://www.cebitec.uni-bielefeld.de/groups/brf/software/arraylims/cgi-bin/index.cgi
Go
Array LIMS
Quick navigation
Target
Target labeling
Hybridization
Image upload
Report
List protocols
Project selection
Edit
Help!
Options
Search
Done

User: mdondrup
Project: ArrayLIMS:Coryne
Role: Admin

Index

Target
First step of a microarray experiment
Upload a description of the Target sample.This includes your experimental setup like polymer type (whether your target was extracted from a DNA or RNA), growth conditions, used organism etc. Note that if you do not upload a description of the Target you will not be able to give a description for your target labeling in the next step.

Target labeling
Second step of a microarray experiment
Upload a description of the target labeled from your Target from the first step.This includes your experimental setup like used Target, amount of the used Target etc.

Microarray hybridization
Third step of a microarray experiment
Upload a description for the hybridization of a particular microarray.This includes names and volumes of the used (and already uploaded) labeled target, the slide_id etc.

Image upload
Upload the scanned images of a hybridized microarray together with the corresponding data files. This includes name of the scanned slide, name of the scan software etc.
Note that you can only upload images for an already uploaded microarray hybridization (step 1-3)

Report
View a report for all steps of the uploaded hybridization of a particular microarray as well as a minimised print version of the report.

List protocols
List all protocols (Target protocols, labeled target protocols etc) of your project.

Edit
Edit already uploaded data e.g. to delete wrong data or to add data. As a chief of a project you can add/delete slide_ids as well as import protocols from one project

Report of all slides in project

To hide unwanted rows or columns click the 'X' infront of each row/column.To show the row/column again click 'undo'.
To get a print version of the chosen rows/columns click 'Print version'.

Back to Index
Undo Print version

		☒	☒	☒	☒	☒	☒	☒	☒
☒	Slide_ID	Cg3kOligo123s02	Cg3kOligo121s02	Cg3kOligo122s02	Cg3kOligo120s02	Cg4kPCR042s01	Cg4kPCR043s01	Cg4kPCR236s03	Cg4kPCR045s01
☒	Hybridization date	29.8.2006	13.10.2006	29.8.2006	13.10.2006	4.2.2003	4.2.2003	12.7.2006	25.2.2003
☒	Operator of the hybridization	marcusp	marcusp	marcusp	marcusp	diana	diana	diana	diana
☒	Volume of labeled target1 used	35	35	35	35	20	20	20	20
☒	Volume of labeled target2 used	35	3						
☒	Hybridization solution	DIG-EasyHyb	DIG-Ea						
☒	Hybridization remarks	-							
☒	Hybridization protocol	Cg_Hyb_HS4800_55°C-45°C	Cg_Hyb_HS4						005
☒	Alterations of the hybridization	-							
☒	Labeled target ID1	mp_DM1730_TypIIFerm_1_T2_Cy5	mp_DM1730_TypI						
☒	Date of labeled target1	28.8.2006	28.8.						
☒	Operator of labeled target1	marcusp	mar						
☒	Amount of target1 used	-							
☒	Dye of labeled target1	Cy5	C						
☒	Volume of labeled target1	35	3						
☒	Remarks of labeled target1	-							
☒	Protocol of labeled target1	Cg_Labeling_Jan2006	Cg_Labeli						
☒	Alterations of labeled target1	-							

		☒	☒	☒
☒	Slide_ID	Cg3kOligo123s02	Cg3kOligo121s02	Cg3kOligo122s02
☒	Hybridization date	29.8.2006	13.10.2006	29.8.2006
☒	Operator of the hybridization	marcusp	marcusp	marcusp
☒	Volume of labeled target1 used	35	35	35
☒	Volume of labeled target2 used	35	35	35
☒	Hybridization solution	DIG-EasyHyb	DIG-EasyHyb	DIG-EasyHyb
☒	Hybridization remarks	-	-	-
☒	Hybridization protocol	Cg_Hyb_HS4800_55°C-45°C	Cg_Hyb_HS4800_55°C-45°C	Cg_Hyb_HS4800_55°C-45°C
☒	Alterations of the hybridization	-	-	-

Figure 7.5: Screenshot of the ArrayLIMS web-interface. The ArrayLIMS provides a linear workflow of web-forms to enter information about each step of the experiment (top). The report page provides an overview of all arrays in the repository and their annotations (the colored portion is enlarged for readability). The arrays are represented by the columns of the table.

7.2.3 Access Control

Following the extended ACL model developed in Section 6.2.2, the definition of user rights has to be defined on a per-object level. An ACL constitutes the unique combination of a user or group with an object and the associated access rights to that object. As some ACLs would otherwise be redundant, there can only be a single ACL for each combination of user or group and object.

MySQL, as many other RDBMS, does not provide direct access control on a per-row basis. The data structures are also scattered over multiple tables. Thus, the access control mechanism is implemented using O2DBI-objects. The implementation in EMMA2 is generic, using only four classes representing users, groups of users, and the group and user ACLs. The rights, contained in the ACLs can be easily modified. The storage related part of the access model does not make assumptions about the controlled objects, and can be used in other softwares components as well.

The access control model is completely respected within the web-interface. The system will not list or display objects for which a user has no permissions. The central access control mechanism is implemented on the level of the multiplexer script of the web-interface and within the EMMA2 API of the system architecture. This guarantees that access restrictions are enforced under all circumstances, irrespective of the implementation of the presentation components.

This mechanism operates as follows:

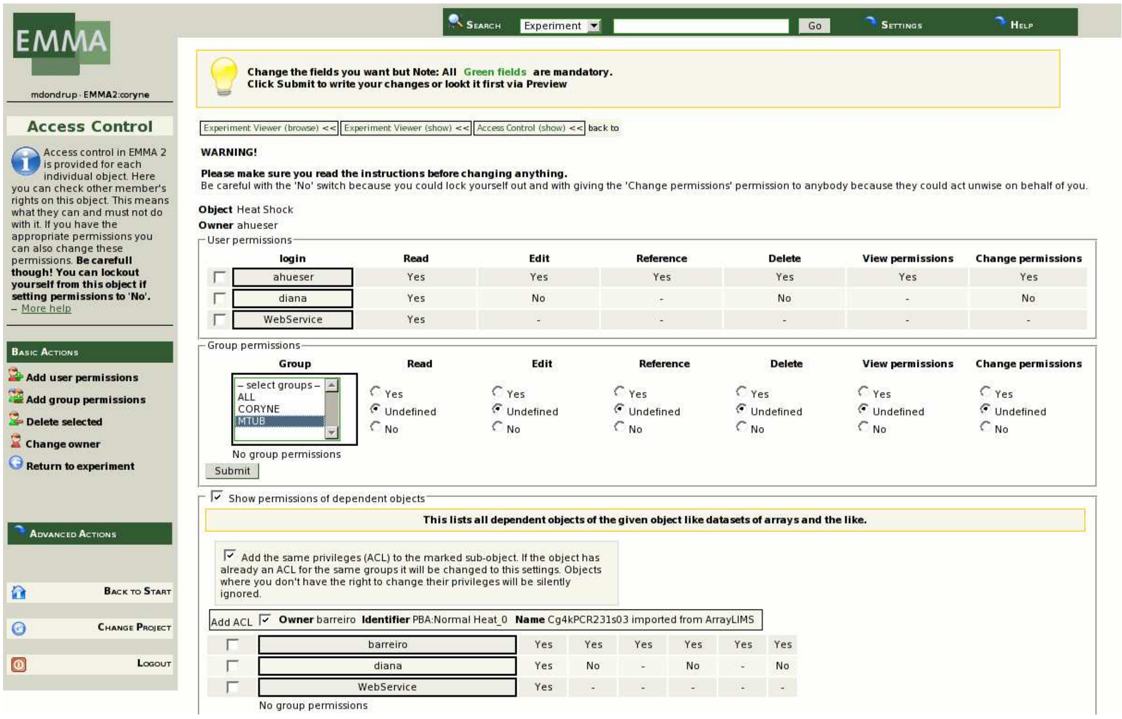
1. The desired action and optional MAGE objects, referenced by their unique O2DBI object-ids, are passed as parameters to the multiplexer.

2. Based on the role-based access control model, the multiplexer decides whether the current user is allowed to perform this action at all.

3. The multiplexer initializes a list of objects by their unique ids if provided; or it retrieves a list of all objects of a given class depending on the desired action. By using EMMA2 API-calls, the multiplexer can retrieve only those objects the user may read.

4. If the desired action involves modification of the object, the multiplexer checks preemptively whether this action is allowed for each object found. This prevents the multiplexer from raising software errors by invoking a forbidden API-call on an object erroneously.

5. If the action is allowed, the appropriate method of the presentation module that can handle objects of the given class is called and the resulting web-page is displayed. Otherwise, the multiplexer issues an 'access-denied' message.



mdondrup · EMMA2:coryne

Access Control

Access control in EMMA 2 is provided for each individual object. Here you can check other member's rights on this object. This means what they can and must not do with it. If you have the appropriate permissions you can also change these permissions. **Be carefull though! You can lockout yourself from this object if setting permissions to 'No'.** – More help

Change the fields you want but Note: All Green fields are mandatory.
Click Submit to write your changes or lookt it first via Preview

WARNING!

Please make sure you read the instructions before changing anything.
Be careful with the 'No' switch because you could lock yourself out and with giving the 'Change permissions' permission to anybody because they could act unwise on behalf of you.

Object Heat Shock
Owner ahueser

User permissions

	login	Read	Edit	Reference	Delete	View permissions	Change permissions
□	ahueser	Yes	Yes	Yes	Yes	Yes	Yes
□	diana	Yes	No	-	No	-	No
□	WebService	Yes	-	-	-	-	-

Group permissions

Group	Read	Edit	Reference	Delete	View permissions	Change permissions
– select groups – ALL CORYNE MTUB	○ Yes ● Undefined ○ No	○ Yes ● Undefined ○ No	○ Yes ● Undefined ○ No	○ Yes ● Undefined ○ No	○ Yes ● Undefined ○ No	○ Yes ● Undefined ○ No

No group permissions

☑ Show permissions of dependent objects

This lists all dependent objects of the given object like datasets of arrays and the like.

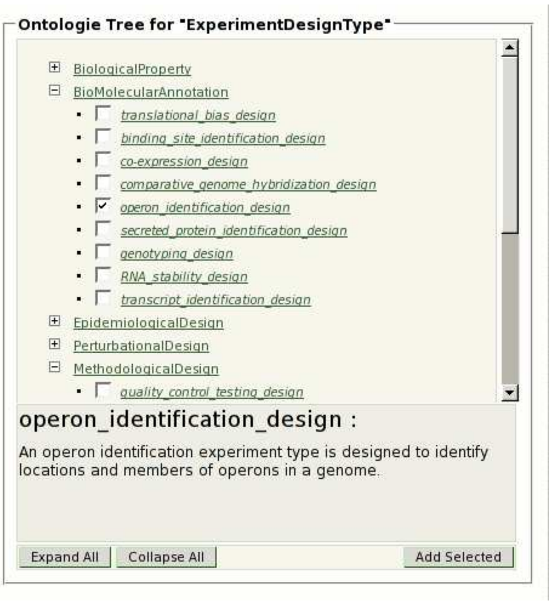
☑ Add the same privileges (ACL) to the marked sub-object. If the object has already an ACL for the same groups it will be changed to this settings. Objects where you don't have the right to change their privileges will be silently ignored.

Add ACL ☑ **Owner** barreiro **Identifier** PBA:Normal Heat_0 **Name** Cg4kPCR231s03 imported from ArrayLIMS

□	barreiro	Yes	Yes	Yes	Yes	Yes	Yes
□	diana	Yes	No	-	No	-	No
□	WebService	Yes	-	-	-	-	-

No group permissions

Figure 7.6: Access control in the EMMA2 web-interface. The individual ACLs of users are listed in the top box. The experiment has ACLs for two human users and is readable by the WebService user. In the group permissions box, a new group ACL for the group 'MTUB' is about to be assigned. A user can set all options for each object and dependent objects included in this experiment (bottom box).



Ontologie Tree for "ExperimentDesignType"

- ⊞ BiologicalProperty
- ⊟ BioMolecularAnnotation
  - ☐ translational_bias_design
  - ☐ binding_site_identification_design
  - ☐ co-expression_design
  - ☐ comparative_genome_hybridization_design
  - ☑ operon_identification_design
  - ☐ secreted_protein_identification_design
  - ☐ genotyping_design
  - ☐ RNA_stability_design
  - ☐ transcript_identification_design
- ⊞ EpidemiologicalDesign
- ⊞ PerturbationalDesign
- ⊟ MethodologicalDesign
  - ☐ quality_control_testing_design

operon_identification_design :

An operon identification experiment type is designed to identify locations and members of operons in a genome.

Figure 7.7: Screenshot of the OntoDB ontology browser. It displays all sub-classes of ExperimentDesignType which is used to annotate the type of experiment performed. A sub-tree is opened and the description for an item is displayed. Multiple entries can be marked to add the values into the corresponding fields in EMMA2.

7.2.4 Ontology Integration

The proper annotation of experimental data requires to use ontology terms from the MGED ontology. It is possible, though not trivial, to enter MAGE terms directly into the fields of forms displayed by EMMA2. The MGED ontology is fairly large system (see Section 4.1.3) and it provides only limited tools for browsing and searching it directly via the web.

To improve this, a more comfortable web-interface for use with ontologies is needed. Such an ontology browser is very useful within many functional genomics applications that involve human annotations; accordingly, the ontology browser has been developed and implemented as a separate application and is independently usable with all ontologies specified using the OWL format. Another advantage of such an ontology system is that it allows for locally customized ontologies.

The ontology browser is not part of EMMA2, but it can be linked into an existing application by providing a set of Perl-modules and HTML-templates (*OntoDB*) to include it into any application requiring an ontology. In contrast to the MGED ontology browser, OntoDB displays a tree representation of the ontology terms allowing to navigate through the nodes of the class hierarchy (see Figure 7.7 in comparison to Figure 4.2 on page 59, where the same hierarchy is depicted).

Another important feature is context-sensitive filtering. Searches can be restricted to certain ontology classes. For EMMA2 the filter display only those entries required by the current MAGE-OM object. As a result, the user is not distracted by a majority of irrelevant terms.

7.3 Data Integration and Interoperability

7.3.1 BRIDGE

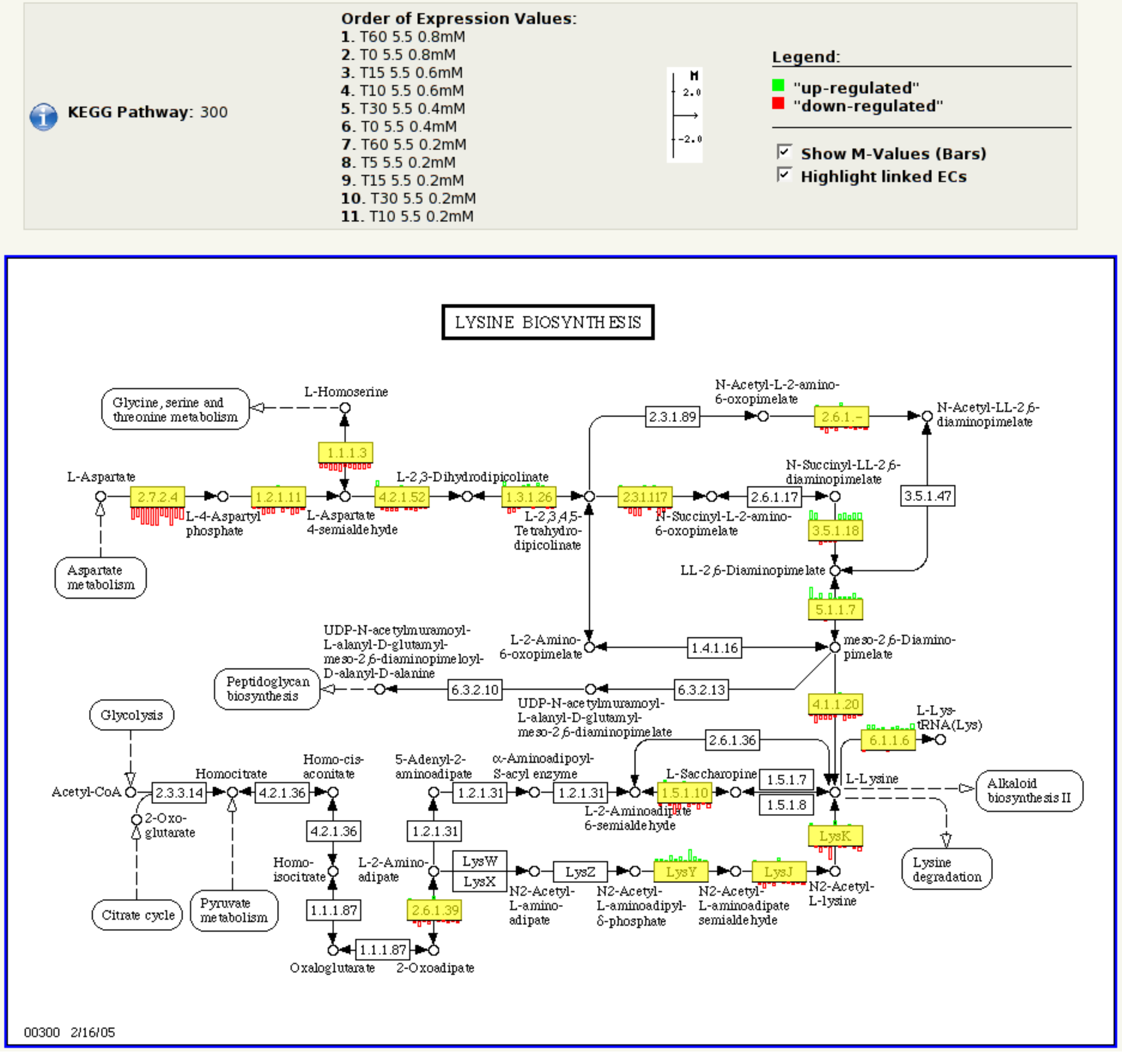
The BRIDGE-module is an application intended for integration of heterogeneous distributed datasources (Goesmann *et al.*, 2003, 2005). It is written in Perl and integrates seamlessly with the O2DBI abstraction layer. On the server-side BRIDGE makes all classes and methods of an O2DBI-API, that have been marked as external, accessible by remote clients. On the client-side BRIDGE provides transparent access to remote objects. The referred objects may be accessed in the same way as if they were local objects, including method invocation. The API of the application to which the referred objects belong needs to be installed on the client, which makes for a limiting requirement. Access control is fully provided for BRIDGE-references, making it possible to expose the full API of the application-layer to the client. To avoid overly complex access, however, only some sub-classes of *Identifiable* in MAGE-OM are referrable directly via BRIDGE.

The MAGE-OM class *BioSequence* has references to GenDB and ProDB *Region* classes, which constitute the most general super-class of a biological sequence representative in these applications. That way, sequence annotations and information about protein abundances can be retrieved directly from these applications. For non-genomic sequences like ESTs BioSequences representing ESTs in EMMA2 are linked to their counterparts in the SAMS system as SAMS is built on the same data-model as GenDB. All client access is performed by calling methods from the API of the referenced Region object. An example application is given in Figure 7.8 on the facing page.

BRIDGE uses Uniform Resource Identifiers (URIs) to reference external objects for each BioSequence object that it contains. Sequence annotations located on local or remote GenDB servers can be accessed by using the URIs to pass messages between the GenDB and EMMA2 servers over an internet connection. The URIs can be generated on the fly while importing ArrayLayouts and the corresponding sequence data into EMMA2.

7.3.2 Web-Services and BioMoby

Web-services can be defined as software applications, which interact with other remote software over an internet-connection using XML-based messages. As such, they provide public access to data or computation. The interface of a web-service can described by using XML-documents, allowing for XML-based searches for web-services. Often, HTTP (Hypertext Transfer Protocol) is used to transfer messages, while other transport-protocols can be used as well. Web-services are, unlike web-sites, not meant for direct human interaction, but for structured data-transmission between software. Nevertheless, the core functionality is often similar to a web-site. Many approaches to web-services originate from the intent to avoid error-prone parsing of HTML-pages (a good example for a straight-forward implementation is



KEGG Pathway: 300

Order of Expression Values:
1. T60 5.5 0.8mM
2. T0 5.5 0.8mM
3. T15 5.5 0.6mM
4. T10 5.5 0.6mM
5. T30 5.5 0.4mM
6. T0 5.5 0.4mM
7. T60 5.5 0.2mM
8. T5 5.5 0.2mM
9. T15 5.5 0.2mM
10. T30 5.5 0.2mM
11. T10 5.5 0.2mM

Legend:
🟩 **"up-regulated"**
🟥 **"down-regulated"**

☑ **Show M-Values (Bars)**
☑ **Highlight linked ECs**

Figure 7.8: Example for data-integration using BRIDGE. The gene-expression data in EMMA is projected on a KEGG-pathway map. The expression values for a time-course experiment are depicted as bars next to their enzymes. The task to create this graphic is to retrieve to which pathway a given BioSequence object in EMMA belongs. The BioSequence is linked to a CodingSequence (CDS) in GenDB, with each coding sequence having an Annotation. The Annotation can, if that information is provided, contain information about the pathways in which the EC-number of a gene occurs.

the web-service of the Google search-engine).

SOAP protocol is an open protocol[2] for "exchanging structured information in a decentralized, distributed environment" and relies on a lightweight XML-format for data-interchange. SOAP was originally derived from the XML-RPC project. Both protocols use HTTP but may also employ other transport-protocols. A SOAP-based web-service can be built using a standard web-server, like Apache. There exist language bindings for SOAP to many programming languages, including Perl and Java.

The Web Service Description Language (WSDL)[3] is an XML-application that allows for the description of web-services. The available methods, data-types and possible ways of access to the service can be described. It is often used in conjunction with SOAP.

Infrastructures for registering web-services have been set-up. The aim for a central registry is, to enable users to easily find services that process certain data-types or perform specific analyses.

The BioMoby-project provides a framework for the interoperability of distributed web-services related to functional genomics (Wilkinson and Links, 2002; Wilkinson *et al.*, 2004). It encompasses an ontology for the annotation of services and a central server, called BioMoby Central, to register and query for services (this branch is termed MOBY-S). Clients, service providers, and the BioMoby Central server communicate by exchanging so called MOBY Objects based on SOAP-messages. Services can be registered and described using WSDL documents. Given a data-type, this method allows client programs to automatically search for services of interest using a single-point of entry. The internal representation of the exchanged data is not defined by BioMoby, as its focus beeing on data-integration, not standardization.

One major advantage of web-services is, that they do not require the client to install a server API, thereby allowing for remote access by non-BRIDGE-aware software. The complete API does not have to be exposed to the client application. Compared to BRIDGE, the user can only retrieve data, but no methods of MAGE classes.

Web-services are public by default, though for SOAP, there is the possibility of authorization. Unrestricted access to the MAGE-API would expose the *create* and *delete* methods and other possibly dangerous operations. But even if authorization is implemented properly, the authorization is inconsistent with the open-access concept of BioMoby. This results in the need to register a large number of methods , that almost no one would be allowed to use.

Web-services implement a simplified remote interface as stated in the requirements. The MAGE-API comprises a very large number of classes, while the number of sensible queries is relatively low. Exposing all classes would cause a significant overhead. The web-service layer serves to provide queries which usually encompass

[2] `http://www.w3.org/TR/soap/`

[3] `http://www.w3.org/TR/wsdl`

a large number of classes from the MAGE-model.

7.3.3 EMMA2 as a Web-Service Provider

As a conclusion from the previous considerations, a SOAP web-services interface should not provide a direct interface to the MAGE-API. Instead, the webservices interface is built as an alternative presentation layer. The web-services layer can be seen as an alternative machine-readable presentation to the web-interface of the EMMA2 system.

The web-services implementation provides limited access to the query functionality of the adaptor modules of the facade layer. This layer simplifies and unifies the access to the application layer for the presentation modules. Only those objects that are set to be publicly accessible for web-services are passed.

The actual implementation of the web-services layer closely resembles the web-interface implementation. The web-services Perl script serves as a multiplexer for several loadable service modules. These modules use the `SOAP::Lite` Perl implementation to communicate with the clients via SOAP-messages. Unlike the web interface, no templates are needed. The web-service uses the same apache web-server used for the web-interface. No additional infrastructure is required.

There are methods allowing to retrieve the expression ratio measurements for reporters under a specific experimental condition. In addition, to facilitate use of this functionality by remote applications, a set of utility functions is required allowing for retrieval of reporter identifiers, experiments, and experimental factors. The following SOAP methods are provided by the web-service implementation:

getExpressionValues This SOAP method provides access to expression measurements (M-values) over all replicates of a Reporter or a list of Reporters. Lists of Experiment identifiers and Factor identifiers can be provided to further restrict the query.

getDataByType This method provides access to expression data for user specified data types and for a list of Reporter Identifiers.

getReporterIdsWithFilter allows to retrieve Reporters with expression measurements matching filter criteria for derived datasets.

fetchAllReporterIds This is an utility method which allows to retrieve a large list of all Reporters in the database.

SearchReporterIdsDescription This utility function provides functionality to perform a full-text search within the annotation data of the Reporters in the database.

SearchReporterIdsName provides a wildcard search for Reporters with a specific name.

fetchAllExperiments, fetchAllFactors, fetchAllDataTypes constitute utility functions allowing to retrieve a list of Identifiers and descriptions of corresponding classes. Normally, these functions have to be called prior to a second step of data retrieval to provide information about the existing experiments, experimental factors and data-types which can be queried.

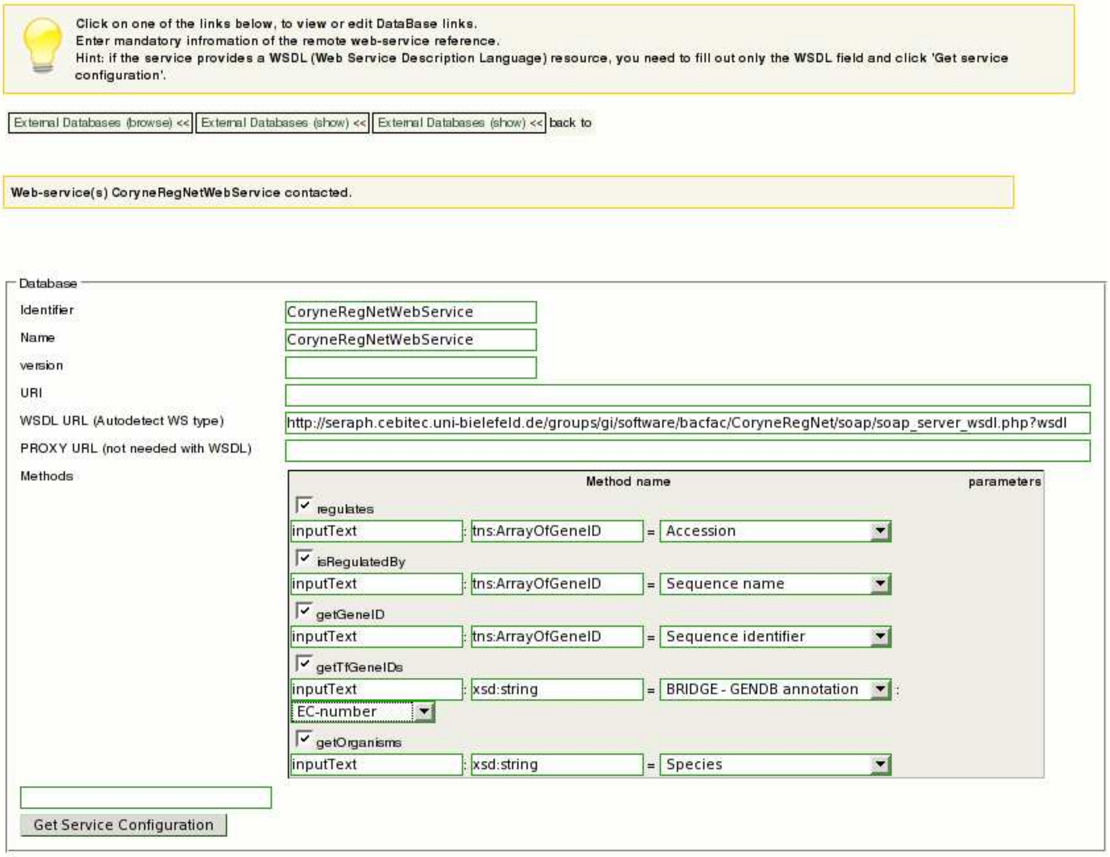
The web-services interface relies on the same facade-layer as the web-interface, but may only access a safe subset of its functionality. As web-services are usually invoked by anonymous clients, the web-service script uses a special user to authorize. To enforce access control, EMMA2 invokes the GPMS module with a specialized system-user named *WebService*. ACLs for this user can be granted or revoked for each object.

7.3.4 EMMA2 as a Web-Service Client

EMMA2 can also act as a web-service client by consuming the services provided by other SOAP enabled applications. All sequences objects in MAGE-OM can be linked with external databases and also web-services. While for web-based databases the web-interface of EMMA2 provides only a hyperlink to the external database, for web-services EMMA2 is able to query the service directly. Responses coming in as SOAP objects from the remote service can be interpreted and directly integrated into the local data-display. With support of the specification of SOAP data-types all responses from any web-service can be displayed automatically.

Most web-services, very much like EMMA2, provide multiple methods with several paramters. Therefore, a mechanism to set-up a web-service connection automatically is important as it allows project administrators to link sequence objects to new emerging web-services without further programming effort. If a WSDL specification of the service is available, which is the case for most web-services now, the process of setting up an EMMA-client for the service is simple. All available methods of the service are derived from the WSDL description together with their input parameters and their data-types. The web-service administrative user interface allows to assign MAGE-OM attributes such as sequence identifiers, sequence names, nucleotide sequence or species to the method parameters. Whenever a linked sequence is displayed, the SOAP methods will be called with the assigned parameters (see Figure 7.9 on the next page).

This mechanism even allows to link BRIDGE functionality with web-services. Annotation data retrieved via the BRIDGE interface can also be assigned as input parameter for a SAOP method. For example, EC numbers, COG classes, or promoter sequences of a gene can be retrieved from GenDB via BRIDGE after that the result can be transparently sent to a web-service to retrieve further information. Furthermore, web-service connections can be used in automated analysis pipelines, to provide labels for classification.



Click on one of the links below, to view or edit DataBase links.
Enter mandatory infromation of the remote web-service reference.
Hint: if the service provides a WSDL (Web Service Description Language) resource, you need to fill out only the WSDL field and click 'Get service configuration'.

External Databases (browse) <<
External Databases (show) <<
External Databases (show) <<
back to

Web-service(s) CoryneRegNetWebService contacted.

Database

Identifier: CoryneRegNetWebService
Name: CoryneRegNetWebService
version:
URI:
WSDL URL (Autodetect WS type): http://seraph.cebitec.uni-bielefeld.de/groups/gi/software/bacfac/CoryneRegNet/soap/soap_server_wsdl.php?wsdl
PROXY URL (not needed with WSDL):
Methods:

Method name parameters

☑ regulates
inputText : tns:ArrayOfGeneID = Accession

☑ isRegulatedBy
inputText : tns:ArrayOfGeneID = Sequence name

☑ getGeneID
inputText : tns:ArrayOfGeneID = Sequence identifier

☑ getTfGeneIDs
inputText : xsd:string = BRIDGE - GENDB annotation : EC-number

☑ getOrganisms
inputText : xsd:string = Species

Figure 7.9: Configuration interface to use EMMA2 as a web-service client. The configuration wizard allows to connect to a remote service providing a WSDL-service description. In this case, the CoryneRegNet Service is contacted, which is a database of regulatory networks (Baumbach *et al.*, 2006). All functions the service provides are listed and function call parameters can be assigned to MAGE-OM information and to BRIDGE features.

7.4 Implementation of Data Analysis

The implementation of the analysis pipeline system allows the definition of a pipeline consisting only of matching consecutive function types. Within an experiment there might exist many datasets, either measured or derived from other datasets by a transformation. To assist the user, the process of choosing a dataset is highly automatized. The whole interface relies heavily on guessing sensible defaults, but it provides access to all parameters for more experienced users.

7.4.1 Function Definition

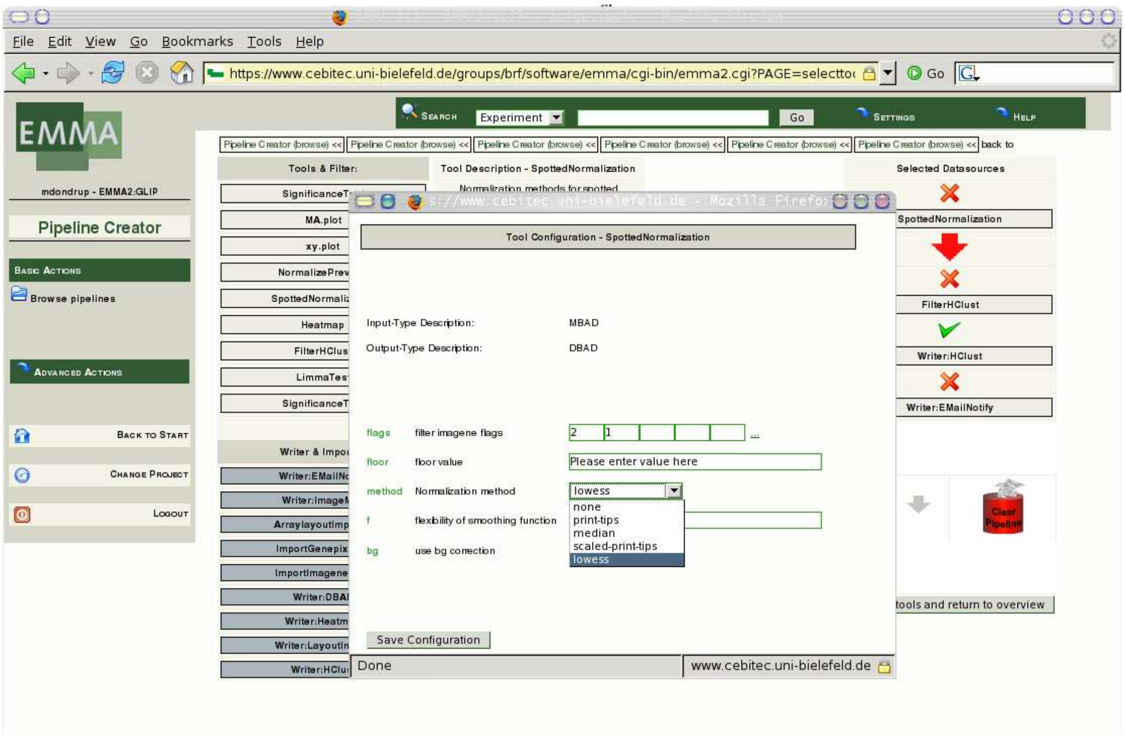
The definition of analysis pipelines from functions in the database is fully integrated into the web-interface. The 'Pipeline Creator' wizard allows to put a pipeline together from building block in a graphical programming style and to set default parameters for each of the building blocks (see Figure 7.10 on the facing page).

Defining new pipelines is restricted to administrative users (the 'Chief'); all users may use the predefined pipelines on their data. They may also adjust paramters for a single run, but cannot effect global defaults. Running an analysis method on an experiment is a three-step process. The user needs to choose the analysis method, the data to which to apply the method and to define paramters. In the simplest case, with default paramters, this can be accomplished by two clicks: by choosing the method and accepting the automated settings. By default, the pipeline system will assume, the method should be applied to all arrays in the experiment. It will further automatically select the required datasets of the appropriate type for the actual pipeline. Manual selection of datasets is still possible, but rarely recommended. Advanced users may also configure all function parameters (see Figure 7.11 on page 120).

After setting up a new project, it is devoid of analysis functions. They are by design regarded as optional code modules; the availability and combination of functions depends on the requirements of the projects. A set of functions can be expected to be required in almost any project seen so far. These functions are applicable for the four major fields of analysis of gene expression data identified in Chapter 3:

- Normalization and preprocessing of raw data
- Statistical inference to identify differential expression
- Unsupervised learning
- Supervised learning and classification

Such functions are provided in the base distribution of EMMA2, ensuring all fields of analysis can be covered from the start. All data analysis functions are implemented using the R programming environment. Most of the functions are either part of already existing code libraries or extensions thereof.


File Edit View Go Bookmarks Tools Help
https://www.cebitec.uni-bielefeld.de/groups/brf/software/emma/cgi-bin/emma2.cgi?PAGE=selecttoo
Go
EMMA
Search
Experiment
Go
Settings
Help
Pipeline Creator (browse) << Pipeline Creator (browse) << Pipeline Creator (browse) << Pipeline Creator (browse) << Pipeline Creator (browse) << Pipeline Creator (browse) << back to
mdondrup - EMMA2:GLIP
Pipeline Creator
Basic Actions
Browse pipelines
Advanced Actions
Back to Start
Change Project
Logout
Tools & Filter:
SignificanceT
MA.plot
xy.plot
NormalizePrev
SpottedNormaliz
Heatmap
FilterHClus
LimmaTes
SignificanceT
Writer & Impo
Writer:EMailNo
Writer:ImageN
ArraylayoutImp
ImportGenepix
ImportImagene
Writer:DBAI
Writer:Heatm
Writer:LayoutIn
Writer:HClu
Tool Description - SpottedNormalization
Normalization methods for spotted
Selected Datasources
SpottedNormalization
FilterHClust
Writer:HClust
Writer:EMailNotify
Clear Pipeline
tools and return to overview
s://www.cebitec.uni-bielefeld.de - Mozilla Firefox
Tool Configuration - SpottedNormalization
Save Configuration
Done
www.cebitec.uni-bielefeld.de

Input-Type Description: MBAD

Output-Type Description: DBAD

flags filter imagene flags 2 1 ...

floor floor value Please enter value here

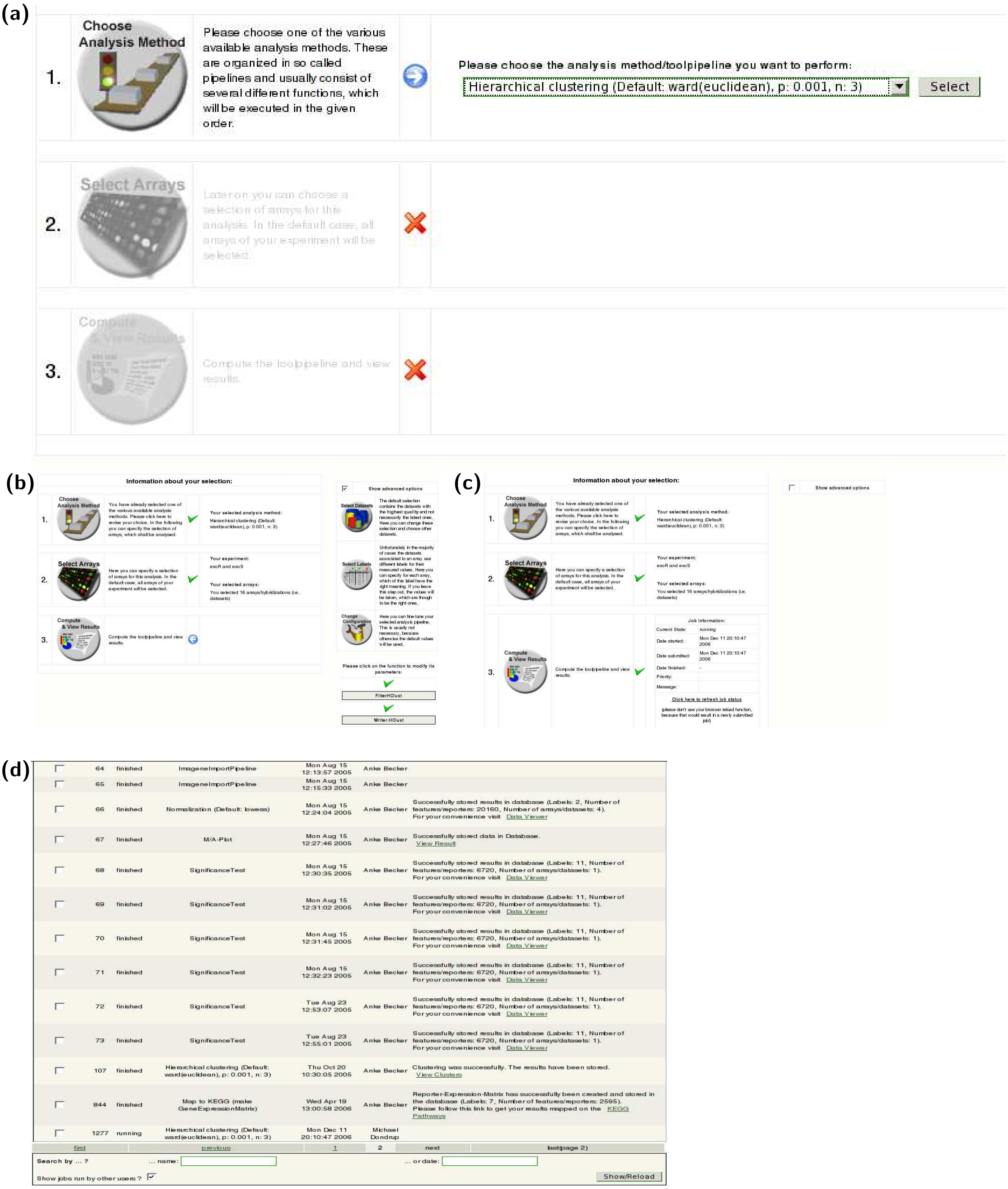
method Normalization method lowess

none
print-tips
median
scaled-print-tips
lowess

f flexbility of smoothing function

bg use bg correction

Figure 7.10: The 'Pipeline Creator' wizard. In the left column, all analysis methods, enabled in this project are depicted as blocks. The blocks at the top depict computational functions; dark blocks at the bottom depict writer and importer functions. On the right, the sequential analysis pipeline is depicted consisting of a list consecutive functions. In this case, the pipeline consist of normalization, filtering and clustering, a writer function to store the results in the database, and an email notify function. In the central window, the paramters of the normalization function are beeing adjusted.



1. Please choose one of the various available analysis methods. These are organized in so called pipelines and usually consist of several different functions, which will be executed in the given order.

Please choose the analysis method/toolpipeline you want to perform:

Hierarchical clustering (Default: ward(euclidean), p: 0.001, n: 3) ▾ Select

2. Later on you can choose a selection of arrays for the analysis. In the default case, all arrays of your experiment will be selected.

3. Compute the toolpipeline and view results.

Information about your selection:

1. You have already selected one of the various available analysis methods. Please click here to revise your choice. In the following you can specify the selection of arrays, which shall be analysed. ✓ Your selected analysis method: Hierarchical clustering (Default: ward(euclidean), p: 0.001, n: 3)

2. Here you can specify a selection of arrays for this analysis. In the default case, all arrays of your experiment will be selected. ✓ Your experiment: exoR and exoS. Your selected arrays: You selected 16 arrays/hybridizations (i.e. datasets)

3. Compute the toolpipeline and view results.

☑ Show advanced options

Select Datasets: The default selection contains the datasets with the highest quality and not necessarily the latest ones. Here you can change these selection and choose other datasets.

Select Labels: Unfortunately in the majority of cases the datasets associated to an array use different labels for their measured values. Here you can specify for each array, which of this label have the right meaning. If you leave this step out, the values will be taken, which are though to be the right ones.

Change Configuration: Here you can fine-tune your selected analysis pipeline. This is usually not necessary, because otherwise the default values will be used.

Please click on the function to modify its parameters:

✓ FilterHClust

✓ Writer:HClust

Information about your selection:

1. You have already selected one of the various available analysis methods. Please click here to revise your choice. In the following you can specify the selection of arrays, which shall be analysed. ✓ Your selected analysis method: Hierarchical clustering (Default: ward(euclidean), p: 0.001, n: 3)

2. Here you can specify a selection of arrays for this analysis. In the default case, all arrays of your experiment will be selected. ✓ Your experiment: exoR and exoS. Your selected arrays: You selected 16 arrays/hybridizations (i.e. datasets)

3. Compute the toolpipeline and view results. ✓

Job Information:	
Current State:	running
Date started:	Mon Dec 11 20:10:47 2006
Date submitted:	Mon Dec 11 20:10:47 2006
Date finished:	-
Priority:	
Message:	

Click here to refresh job status

(please don't use your browser reload function, because that would result in a newly submitted job!)

□ Show advanced options

□	64	finished	ImageneImportPipeline	Mon Aug 15 12:13:57 2005	Anke Becker	
□	65	finished	ImageneImportPipeline	Mon Aug 15 12:15:33 2005	Anke Becker	
□	66	finished	Normalization (Default: lowess)	Mon Aug 15 12:24:04 2005	Anke Becker	Successfully stored results in database (Labels: 2, Number of features/reporters: 20160, Number of arrays/datasets: 4). For your convenience visit Data Viewer
□	67	finished	M/A-Plot	Mon Aug 15 12:27:46 2005	Anke Becker	Successfully stored data in Database. View Result
□	68	finished	SignificanceTest	Mon Aug 15 12:30:35 2005	Anke Becker	Successfully stored results in database (Labels: 11, Number of features/reporters: 6720, Number of arrays/datasets: 1). For your convenience visit Data Viewer
□	69	finished	SignificanceTest	Mon Aug 15 12:31:02 2005	Anke Becker	Successfully stored results in database (Labels: 11, Number of features/reporters: 6720, Number of arrays/datasets: 1). For your convenience visit Data Viewer
□	70	finished	SignificanceTest	Mon Aug 15 12:31:45 2005	Anke Becker	Successfully stored results in database (Labels: 11, Number of features/reporters: 6720, Number of arrays/datasets: 1). For your convenience visit Data Viewer
□	71	finished	SignificanceTest	Mon Aug 15 12:32:23 2005	Anke Becker	Successfully stored results in database (Labels: 11, Number of features/reporters: 6720, Number of arrays/datasets: 1). For your convenience visit Data Viewer
□	72	finished	SignificanceTest	Tue Aug 23 12:53:07 2005	Anke Becker	Successfully stored results in database (Labels: 11, Number of features/reporters: 6720, Number of arrays/datasets: 1). For your convenience visit Data Viewer
□	73	finished	SignificanceTest	Tue Aug 23 12:55:01 2005	Anke Becker	Successfully stored results in database (Labels: 11, Number of features/reporters: 6720, Number of arrays/datasets: 1). For your convenience visit Data Viewer
□	107	finished	Hierarchical clustering (Default: ward(euclidean), p: 0.001, n: 3)	Thu Oct 20 10:30:05 2005	Anke Becker	Clustering was successfully. The results have been stored. View Clusters
□	844	finished	Map to KEGG (make GeneExpressionMatrix)	Wed Apr 19 13:00:58 2006	Anke Becker	Reporter-Expression-Matrix has successfully been created and stored in the database (Labels: 7, Number of features/reporters: 2595). Please follow this link to get your results mapped on the KEGG Pathways
□	1277	running	Hierarchical clustering (Default: ward(euclidean), p: 0.001, n: 3)	Mon Dec 11 20:10:47 2006	Michael Dondrup	

first | previous | 1 | 2 | next | last(page 2)

Search by ... ? ... name: ____ ... or date: ____

Show jobs run by other users ? ☑ Show/Reload

Figure 7.11: Automated data-analysis wizard. The first step allows to choose an analysis method **(a)**. The datasets that fit with the type of the pipeline are automatically selected. Buttons on the right allow to set further configuration options **(b)**. The method can directly be started by clicking the 'Compute' button **(c)**. The list of all jobs for an experiment, their results and messages are recorded in the job-list **(d)**.

To describe an analysis function for the PlugIn system, an XML application, called TOOLS-ML, has been defined, which is closely related to MAGE-ML. The descriptions of functions, their parameters and function types are imported directly into the database. A page to set the parameters of the function is automatically generated in the interface from the parameter descriptions in the XML-file. If a custom page for the parameters is required, this page can be added to the system and will override the automatic page.

For each data type, there can be additional utility functions, which serve to store the data in the database and present the results interactively. But the programmer of new PlugIns rarely needs to add such methods because most datatypes can be handled by the default components already included in EMMA2.

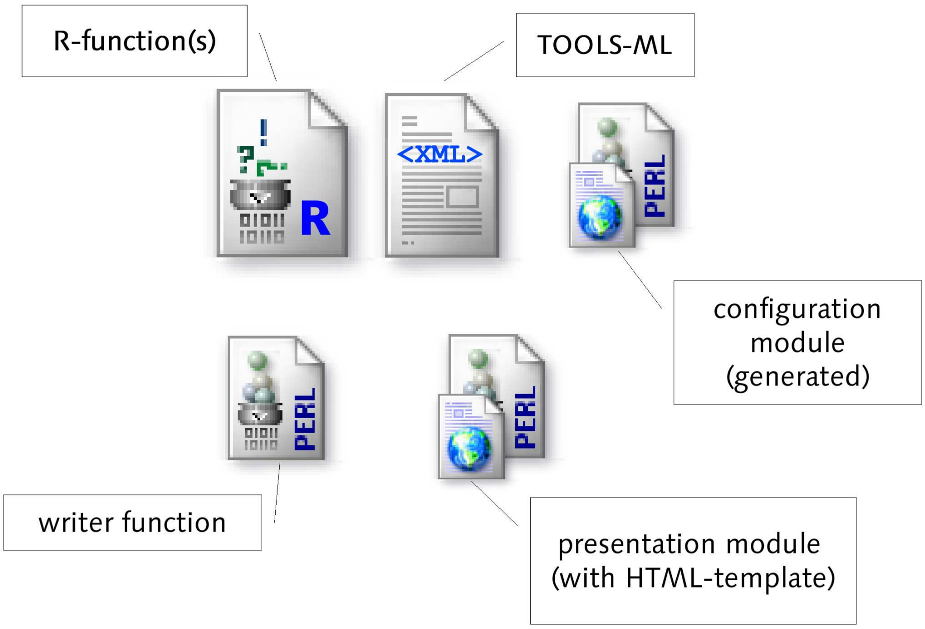

R-function(s)
TOOLS-ML
R
<XML>
PERL
configuration module (generated)
PERL
PERL
writer function
presentation module (with HTML-template)

Figure 7.12: Overview on the components necessary to add an new analysis function to EMMA. For most functions, only the function code itself and a Tools-ML document are required.

If an analysis method requires specialized visualization methods, the presentation module can be implemented and added the same way as other functional components are added to the web-interface. Figure 7.4.1 gives an overview of the mandatory and optional components used to specify an analysis PlugIn.

7.4.2 The Pipeline API

The EMMA2 pipeline system carries out all computational tasks of data analysis and also long running jobs like data import and export tasks. Therefore it makes use of the batch-queuing system Sun Grid Engine V6 (SGE). The batch system schedules compute jobs to a cluster of multiprocessor computers; by this mechanism it is poss to control parameters like how many simultaneous jobs can be executed and to which machines they are executed. The batch system is extensible to support other queuing systems in the future. The use of the batch system is completely transparent to a PlugIn programmer.

PlugIns can be programmed as either functions in the statistical environment R, as Perl-functions or as external scripts. For R and Perl functions there exists an API which the programmer can use inside the PlugIn function to access and return data from and to the EMMA2 core. For R, there exists an R-Perl interface based on the RSPerl R-package. The interface allows bidirectional transfer of complex data between an R funtion and the EMMA2 core; it is also used to call the appropriate R functions and to transfer the parameters of the functions.

All computational functions have access to raw or transformed array data by the use of so called BADObjects (for BioAssayData objects). BADObjects contain microarray data together with reporter annotations and experimental annotations to be able to group data sets. The required data-type, be it raw data, normalized data, or otherwise transformed data, is determined automatically by the type-system of the pipeline. The PlugIn programmer does not have to take care of how to get the required data from the database. At the beginning of the pipeline execution, data of the appropriate type is fetched automatically from the core API and put into the BADObject.

Most existing functions require to program a wrapper function to transform the data structures provided by EMMA2 into appropriate data structures (BADObjects) which can serve as input for the existing R functions. This step is needed because of the heterogeneous implementations of the available R functions. Importer and exporter utility functions, as functions dealing directly with the MAGE data model or with BRIDGE functionality were implemented as Perl subroutines.

The type of the first function in the pipeline determines the data type required for the analysis pipeline and is used to make an automatic selection of datasets. An automated selection of datasets by their type is not possible when there exist alternative datasets of the same type. Multiple datasets result from multiple applications of the same analysis functions.

To provide a solution for this, the system allows users to assign a quality rating to each dataset. The datasets with the highest quality rating are preferred. If the quality ratings are equal, the type system will choose the most recent dataset.

7.4.3 Normalization and Preprocessing

Normalization and preprocessing methods are dependent on the microarray platform . Spotted two-color microarrays yield two numeric intensities for spots with few per array replicates and also background intensity estimates. Affymetrix data contain one intensity measurement per cell further organized into related PM and MM probe-sets and also some global and local background estimates.

Preprocessing of spotted two-color microarrays can be divided into four steps (see also Section 3.3.2):

1. optional removal of flagged spots

2. optional background correction

3. calculation of a difference statistic

4. normalization of the difference statistic

.

This functionality is implemented in EMMA2 as the R analysis function `SpottedNormalization`, using the bioconductor R package `MAarrayNorm` (Dudoit and Yang, 2003). It allows for calculation of steps 2 to 4. Handling of flagged spots has been added to the function. In addition, floor values for channel intensities have been added. All spots showing intensities less than the threshold are set to the threshold value.

Five different methods are available for normalization of the logarithmic channel intensity ratios:

1. *No normalization* will perform only log-ratio computation.

2. *Median* will adjust the median of the data distribution of individual arrays to zero.

3. *Lowess* will adjust the data distribution by computing a Lowess function for each dataset.

4. *Location dependent* will compute a Lowess function for each grid on the array

5. *Scaled location* will do the same as *Location dependent* and additionally scale the data to have equal Median absolute deviation (MAD).

Variance stabilizing transformation is also applicable as an alternative method using the the Bioconductor package `vsn`. It computes a non-logarithmic difference statistic described in Section 3.3.2. Computation of threshold values is not implemented for vsn.

7.4.4 Statistical Tests

One-sample and two-sample t-tests and Wilcoxon's rank-sum test have been implemented using the R functions (`t.test` and `wilcox.test`). The CyberT method was added using the `bayesreg.R` library by Baldi and Long[4]. LIMMA, VarMixt and RankProd were added using the corresponding BioConductor packages[5]. The R-package `samr` from the Comprehensive R Archive Network (CRAN) provides the SAM method[6]. All methods have been complemented with an option to ignore per-array replicate measurement replacing them by their mean, in order not to overestimate the number of independent samples.

Adjusted p-values are computed by the R function `p.adjust`. The number of tests for the adjustment is by default computed from all real tests; that is, all test not having a sufficient number of replicates (at least two), are excluded from the adjustment list, because these tests should not be counted.

7.4.5 Cluster-Analysis

Hierarchical cluster analysis methods have been implemented using the R package `amap` from CRAN. This package provides a more memory efficient implementation of agglomerative algorithm also found in the `hclust` method. `amap` is required to efficiently cluster several thousand objects as found in microarray experiments. K-means clustering is implemented using the R method `kmeans`. Additional clustering algorithms by Kaufman and Rousseeuw are available from the `cluster` package. The neural-gas algorithm of Martinetz *et al.* is found in the R package `ccluster`; self-organizing maps are found in the `som` package; and model-based cluster analysis is taken from the `mclust` package.

7.4.6 Visualization Methods

All visualization methods are implemented using either graphics generated by R-functions, which form the R-plot device, or as Java-applets. All plots generated by the R-plot device can be exported as pixel-graphics or as postscript files. Computations and plotting are embedded as analysis pipelines which produce Observation objects. Observations are part of the complementary classes in the Tool-concept package. These classes provide containers for all results, which cannot be handled as BioAssayData in the MAGE-OM (see Section 6.2.3). The plots show up in the corresponding experiment. This serves for purposes of documentation of the analysis steps as well as to prevent unnecessary redundant execution of graphical analyses.

[4] downloaded from `http://visitor.ics.uci.edu/genex/cybert/`

[5] `http://www.bioconductor.org/`

[6] `http://cran.r-project.org/`

7.4.7 Scatterplots and MA-plots

Scatterplots and MA-plots are implemented based on plotting functions found in R. Both make use of a xy-scatterplot, where scatterplots assume linear scales, while MA-plots assume logarithmic ratios on the y-axis. In principle, it is possible to produce scatterplots of any data value from any dataset, but sensible defaults have been defined. Scatterplot pipelines will by default automatically select raw datasets, and plot the channel foreground intensities. MA-plot pipelines will automatically select normalized data and use M and A-values respectively.

Both functions offer a large number of user configureable options to add lines to the plots. Plot symbols, line-types and plot colors can be configured as flexible as in R itself. The MA-plot function offers the possibility to plot data arbitrary quantile ranges of the M-axis and to depict the quantile values in the legend.

Scatterplots of microarray data often show the problem of cluttering areas due to the high number of data points in a region. Many spots tend to have similar values. Therefore, a normal scatterplots show black areas of overplotted dots in the central region, from which no further information can be obtained. The density scatterplot is provided as an improvement. Instead of depicting each individual measurement in regions of high density, a density-score of the data distribution is calculated. The density value is mapped on a color scale. Density plots are now the default scatterplot method in EMMA2 (see Figure 3.6 on page 29 for a comparison of both plot types and Figure 8.2 on page 143 for a variant with topographic-map like colors), implemented using the `geneplotter` package from Bioconductor (Gentleman *et al.*, 2004).

All scatterplots are stored within the database; the user may view them by using the *Plot Viewer*, which is a regular component of the web-interface. The plot Viewer can display the resulting graphics as HTML image maps. Therefore, the scatterplot pipeline has to provide pixel coordinates which are stored within the `Observation::ImageMap` object and are read by the Plot Viewer. Each plot position is hyperlinked to the corresponding spot, or the gene sequence, and also to external BRIDGE resources, if these exist.

7.4.8 KEGG-Mapping

To be able to interpret gene-expression data in its functional context, the Pathway-mapping tool is provided. It is functional whenever there is an external annotation project in GenDB or SAMS for the studied organism and at least some of the sequences have a associated EC-number in their annotations.

The KEGG-mapping consists of an analysis pipeline and a visualization component in the web-interface. The KEGG-mapping pipeline will fetch the annotation data from BRIDGE, retrieve the assignments of EC-numbers and pathways by using the API of the foreign applications (mainly GenDB). In a second step, it computes a gene-expression matrix with joint replicate-measurements for each experimental condition. Individual experimental conditions of datasets are identified by a unique combination of their FactorValues in the EMMA2 database like in MAGE-OM. Additional data filters to remove genes without significant expression or expression change may also be included, but are unused by default.

The visualization component is divided into two steps to make the interpretation of the large number of pathways practicable. In the first step, a list of all available KEGG-maps is depicted. The user has the option, to search for a specific gene, EC-number or pathway. To aid the user to find an interesting pathway, the absolute and relative number of genes present in a pathway and in the data are depicted. A pathway might not be complete in the data, either because some genes of a pathway are not present in the organism, are not represented on the microarray or the data have been filtered out due to some criteria.

When the user has decided to investigate a specific pathway, as the second step, the pathway is depicted, using the pathway graphics downloaded from the KEGG-database. The expression values are plotted as bar-plots or heatmap plots on the region of the EC-number corresponding to their gene product. For multivariate experiments, one bar or heatmap box will be plotted for each condition.

All genes, which are present in the microarray layout are also marked, to separate them from missing genes. All genes are cross-linked with the respective annotation information via BRIDGE-links.

7.4.9 Categorial PCA-plots

Sometimes, there might be no prior hypotheses about interesting pathways. In this case, it is not trivial, to identify the pathways which justify further investigation by plotting KEGG-maps. The categorial PCA-plots are applicable whenever categorial annotation information is available, and provide a toolkit to aid the detection of interesting KEGG-pathways, COG-categories or subsystem annotations.

Visualization of multivariate data for a large number of classes can be counter-intuitive for the user. An approach should take into account the limitations of 2-dimensional visualizations and nevertheless allow for the visualization of the typical data-distribution of each category. Reduction of the data dimensionality is therefore required. Principle component analysis is used to reduce the data dimensionality by projecting the expression values on their first principle component. Then, a box-plot is made which puts all categories side by side, joining the expression values of all genes in the categories. A boxplots depicts information about the empirical location, standard deviation, and extreme points of the data distribution.

With single boxplots, it is possible to get global information about the overall distribution of the data. They have limitations, when the underlying distribution is in fact a mixture of distributions. A simple example of such a category is a KEGG-Map, containing several alternative reaction pathways, which are triggered by the experimental conditions. Some of the genes will react with down regulation and others with up-regulation. A simple box-plot will show only an increase of the variance, hiding the multimodal nature of the data.

Violin plots have obvious advantages over simple box-plots: they combine the simplicity of a box-plot with the detail of density estimation. In a violin plot, a mirrored plot of the empirical density estimate is added to a box-plot for the PCA-projection of the original data (see Figure 7.13 on the next page). This approach can easily reveal mixtures if the variation between the mixture-components is approximately parallel to the direction of the first principle component of all data. This is not necessarily the case; therefore PCA can also be computed individually for each category. This approach is guaranteed to find the direction of maximum variation, be there several groups along this direction or not.

Still, this approach can reveal information only about the density of the dimensionality reduced data. The original complexity of the data is hidden. Unfortunately, depicting the density of a multivariate distribution is not easily accomplished. This problem could possibly be addressed by model-based clustering of the data within each pathway separately.

The `Mclust` package allows for model-based cluster analysis and is used to compute an optimal cluster solution for each pathway. A box-plot for each cluster within a single pathway can be added to depict additional structure observed within the data.

7.4.10 Cluster Visualization

The cluster viewer is a Java-applet written for EMMA2 and completely embedded in the web-interface. The rationale for using a Java-applet instead of HTML-pages is the increased need for interactivity and efficiency when handling large datasets and trees. Other web-based software for microarray analysis depicts large trees and heatmaps as static web page with large images and hyperlinks. It is almost impossible to achieve free zooming and object manipulation (like for example rotation of trees and online adjustment of colors) with static web-pages.

The cluster viewer allows to inspect the results of cluster analysis pipeline which has been stored in the database. The result of a hierarchical cluster analysis is depicted as a rooted tree with a heatmap. Trees with lots of leaves are often not easy to navigate as the tree consists of thousands of objects. To solve this, the cluster viewer allows to zoom freely into the tree, to open subtrees in new windows, and to search for the location of genes within the tree. In addition, all branches can be made invisible and all sub-trees can be swapped.

Furthermore, the tree can be cut at any height, to form an acceptable number of clusters. The clusters can be further inspected with a k-cluster plot. This plot depicts the individual clusters, the contained expression profiles, and boxplots of the expression profiles for each experimental condition. Furthermore, one can manually select genes of interest and prepare bar-graphs, line-graphs, or so called web-plots of these profiles. All graphical visualizations created in the bluster viewer can exported as pixel graphics or postscript graphics, ready for inclusion in publications (see Figure 7.14 on page 129).

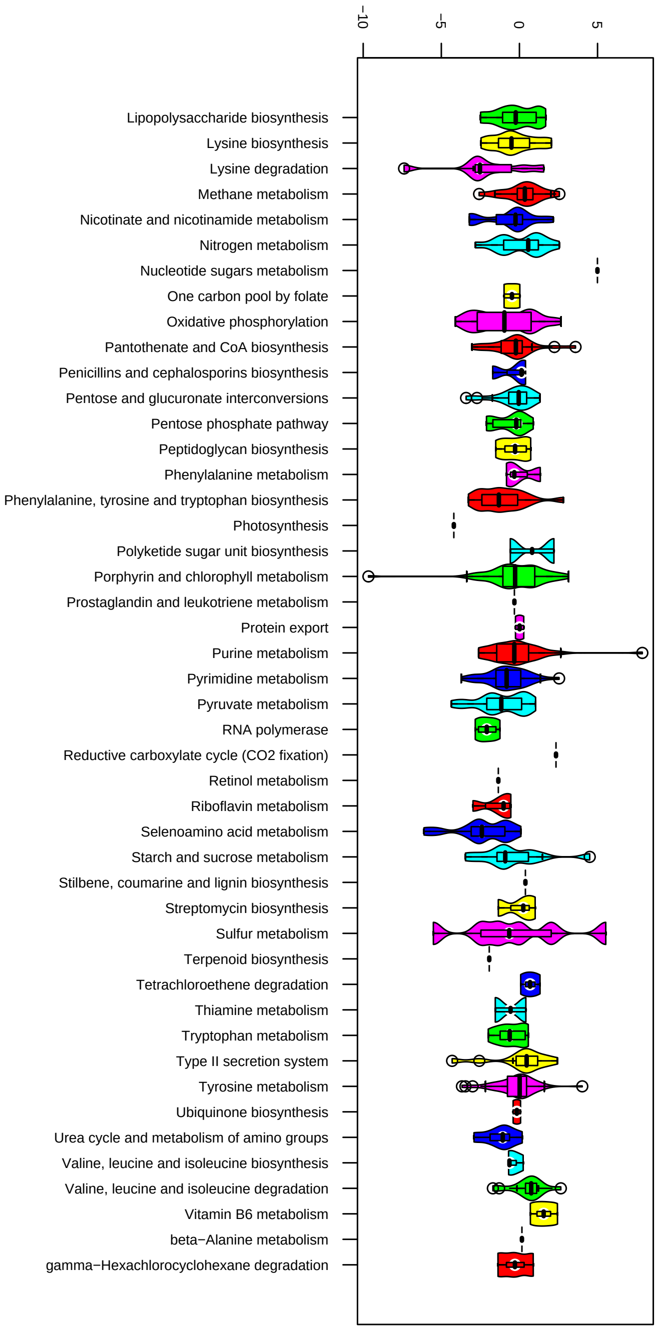
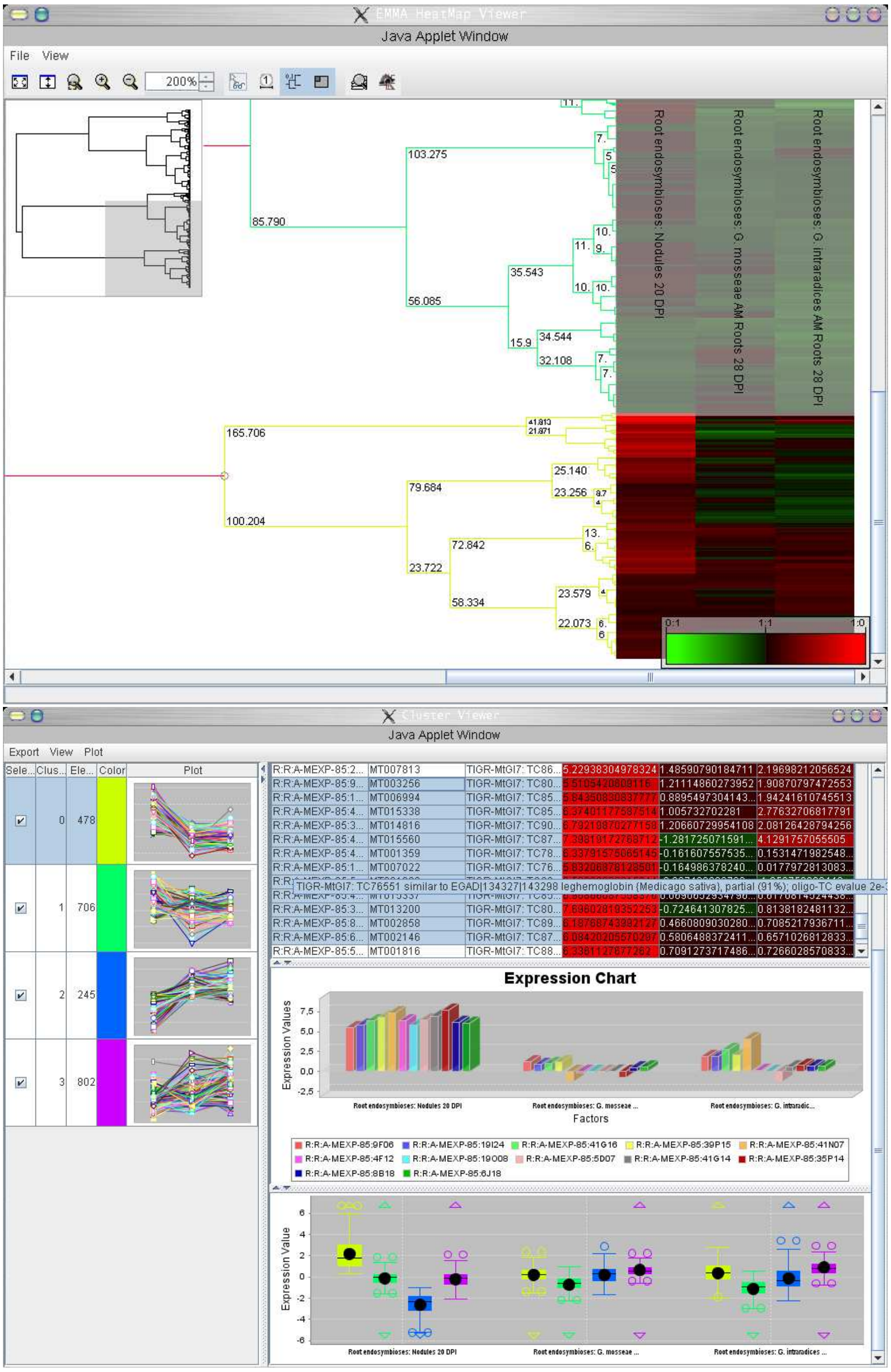


Figure 7.13: The 'KEGG chamber orchestra'. This plot is an example of the violin plots, generated by EMMA2, using the R function `advanced vioplot`. For each of the 91 KEGG-pathways (only the second half is depicted) found in *S. meliloti* in the example experiment, the data are projected on the first principle component and the density of the sample distribution is plotted in combination with a box plot. The width of the boxes of the boxplot depends on the number of members within the pathway. The bivariate nature of the distribution of the expression profiles in the group 'Oxidative Phosphoralation' is directly visible, while the median, spread, group size, and outliers are depicted by the boxplot.



R:R:A-MEXP-85:2...	MT007813	TIGR-MtGI7: TC86...	5.22938304978324	1.48590790184711	2.19698212056524
R:R:A-MEXP-85:9...	MT003256	TIGR-MtGI7: TC80...	5.5105420808116	1.21114860273952	1.90870797472553
R:R:A-MEXP-85:1...	MT006994	TIGR-MtGI7: TC85...	5.84350830837777	0.8895497304143...	1.94241610745513
R:R:A-MEXP-85:4...	MT015338	TIGR-MtGI7: TC85...	6.37401177587514	1.005732702281	2.77632706817791
R:R:A-MEXP-85:3...	MT014816	TIGR-MtGI7: TC90...	6.79219870277158	1.20660729954108	2.08126428794256
R:R:A-MEXP-85:4...	MT015560	TIGR-MtGI7: TC87...	7.39818172768712	-1.281725071591...	4.1291757055505
R:R:A-MEXP-85:4...	MT001359	TIGR-MtGI7: TC78...	6.33791575065145	-0.161607557535...	0.1531471982548...
R:R:A-MEXP-85:1...	MT007022	TIGR-MtGI7: TC76...	5.83206878128501	-0.164986378240...	0.0177972813083...
R:R[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
R:R:A-MEXP-85:4...	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
R:R:A-MEXP-85:3...	MT013200	TIGR-MtGI7: TC80...	7.69602819352253	-0.724641307825...	0.8138182481132...
R:R:A-MEXP-85:8...	MT002858	TIGR-MtGI7: TC89...	6.18768743992127	0.4660809030280...	0.7085217936711...
R:R:A-MEXP-85:6...	MT002146	TIGR-MtGI7: TC87...	6.08420205570297	0.5806488372411...	0.6571026812833...
R:R:A-MEXP-85:5...	MT001816	TIGR-MtGI7: TC88...	6.3381127877262	0.7091273717486...	0.7266028570833...

TIGR-MtGI7: TC76551 similar to EGAD|134327|143298 leghemoglobin (Medicago sativa), partial (91%); oligo-TC evalue 2e-

Figure 7.14: Two screenshots of the EMMA2 cluster viewer. The cluster viewer is a Java that allows easy and detailed navigation of the hierarchical clustering trees (top). The tree can be cut into at an individual hight, yielding individual clusters that can be further analyzed in the cluster panel (bottom). The ordered expression matrix is depicted as a heatmap with adjustable color coding (for example to provide a blue-yellow contrast instead of a standard red-green contrast.) The cluster panel can also be used for non hierarchical methods and provides multiple cluster plots, like bar-plots (bottom center), web-graphs (not-shown), line-graphs(left), and boxplots(at the bottom of the window).

7.4.11 3D-SOM Viewer

The 3D-SOM viewer is another Java applet written specifically for the visualization of the results of a Self-Organizing Map analysis. Unlike partition based clustering methods, self-organizing maps contain a topological ordering of the nodes. The high-dimensional expression data is projected on a low-dimensional (usually 2-dimensional) grid, while trying to preserve the topological releations within the data. When visualizing the 2D grid the third dimension of the visualization space can be used to convey more information about the organization of the grid nodes or representative features of the input data.

The SOM viewer contains four different visualization modes for the generated SOMs:

The static net depicts the network as a rectangular grid of connected balls representing the nodes of the SOM. The length of the interconnecting lines is constant, but the line width is inverse proportional to the Euclidean distance between nodes, depicting the connection strength. Shape and color parameters of all nodes can be controlled by user definable components of the node representative vectors. This results in a total of six components (three RGB-components and three axis components) that can be directly visualized by the appearance of the nodes. By default, the number of vectors attributed to a node controls the diameter of the ball.

The dynamic net has the same features as the Static Net mode. The only difference between the two is, that the distance between the node representative vectors is mapped on distance between the individual balls, so that nodes which are more similar to each other appear closer to each other.

The distance matrix mode depicts all nodes as part of a rectangular surface. The distance between nodes is mapped on the gray value of the surface. A maximum intensity value represents the minimal distance found between node representatives. The surface can either be flat, while the nodes are represented by 'pins' with variable lengths representing the number of vectors assigned to it; or the z-axis of the surface is taken to represent the number of vectors.

The Manhattan grid is a completely new way to visualize SOMs. It is based on the Static net, but with each node represented by a pin. All genes assigned to each node are individually accessible, as they appear as rings, making up the 'stem' of the pin. The position from top to bottom of the node members is assigned by their distance to the node representative vector. The distance is additionally mapped on a color code for each ring. This visualization method has the advantage, that all genes are directly accessible from their nodes and that it allows to judge the cluster quality by the color-code. A high number of nodes with many genes and with high distances to the node representatives can be an indication of a too small SOM grid. The inter-node distance is not visible in the Manhattan grid to keep the complexity of the plots low. As one can immediately switch between visualization modes, it is easy to use a more appropriate mode for inspecting the grid topology and consecutively identify groups of genes of interest.

CHAPTER 8

Applications and their Results

There are currently eight major national and international projects utilizing EMMA2. The projects to which the author has contributed by implementing and applying customized analysis functions are described in detail in the following sections. These projects cover a large variety of microarray applications from bacteria, plants, cancer research, and last but not least the study of marine eco-systems.

EMMA's analysis pipelines have also been used for evaluation of several statistical tests and methods for data-integration. As a secondary effect, this setting provides a framework to evaluate the whole software and its flexibility.

8.1 Overview of the Various EMMA2 Projects

The GenoMik project is currently the largest project with respect to the number of users, hybridizations, and array designs included. It is dedicated to bacteria relevant for agriculture, environment, and biotechnology.

The BACDIVERS project is focused on the *Rhizobia* family of bacteria, which offers high potential in agriculture, natural strain diversity and stress resistance. The projects MEDICAGO, MolMyk and Grainlegumes (GLIP) are all focused on the plant *Medicago truncatula*, which makes for an excellent model organism for symbiotic root interactions between plants and bacteria (e.g. *Rhizobia*) and plants and fungi.

The *Mamma Carcinoma* project is dedicated to human breast cancer research. It aims at improving clinical diagnostic methods and general medical treatment of patients.

The Marine Genomics project (MGE) offers a broad perspective for the application of transcriptomics software to study marine organisms. With respect to the number of project partners from all over Europe and also with respect to the studied organisms and array technologies, it is the most diverse project of all. It consists of project nodes dedicated to fish and shellfish, algae and marine bacteria. For many marine organisms, microarray studies are underway. They involve a large diversity of different array technologies and array layouts. Some laboratories use spotted microarrays, others Agilent and Affymetrix arrays. The application of tiling arrays is also planned. Currently, the project has produced only few hybridizations compared with the other projects, but this is going to change dramatically in thenear future.

In grand total as of January 2007, there are over 2700 hybridizations in more than 400 experiments in various EMMA2 projects. All corresponding raw-data and protocols were processed and uploaded using the ArrayLIMS.

Project	Organisms	Sequence type	Array Technology	# Arrays
MEDICAGO	*M. truncatula*	ESTs	cDNA macroarrays, cDNA microarrays	198
MolMyk	*M. truncatula*, *P. tremula*	ESTs	cDNA microarrays	144
GRAIN-LEGUMES	*M. truncatula*, *P. Sativum*	ESTs	oligo microarrays	343
BACDIVERS	*S. meliloti*	whole genome	cDNA & oligo microarrays	20
GenoMik	different prokaryotes	whole genome	cDNA & oligo microarrays	1510
PathoGenoMik	different prokaryotes	whole genome	cDNA & spotted & *in-situ* oligo arrays	155
Mamma carcinoma	*Homo sapiens*	whole genome	cancer oligo theme array	322
Marine Genomics (prospected)	marine prokaryotes and eukaryotes	ESTs & whole genome	spotted cDNA & oligo, *in-situ* oligo, tiling *in-situ* oligo arrays	48 (>1000)

Table 8.1: Overview of national and international projects which use EMMA2 as their central transcriptomics platform (figures as of January 2007).

8.2 The GenoMik Microarray Database for Bacteria

GenoMik is a project funded by the German Federal Ministry for Education and Research (BMBF) to foster genomic research in microorganisms. It is sub-divided into three competence networks which consist of several national contractors. The competence network dedicated to research on microorganisms relevant for agriculture, environment and industrial production is coordinated by the Department of Genetics (Bielefeld University). The PathoGenoMik network is dedicated to the study of the genomics of pathogenic bacteria which have an impact on public health. The projects comprise sequencing of the genomes of bacteria as well as proteomics and transcriptomics studies.

The Center for Biotechnology provides services for these networks, in particular for sequence analysis and transcriptomics tools including microarrays, hybridization facilities, and data analysis. The users of these tools are located at universities and research institutes throughout Germany.

A substantial number of bacterial genomes have been sequenced during the course of the project. With the availability of the complete genome sequences, some of them obtained at the CeBiTec, large series of microarrays could be produced. The list of organisms for which whole-genome microarrays have been constructed and analyzed with EMMA includes:

- *Sinorhizobium meliloti* (Galibert *et al.*, 2001) a symbiotic soil bacterium, colonizing the roots of legume plants like *Medicago truncatula.* This organism is relavant for agriculture as it is able to fixate atmospheric nitrogen and deliver it to the plant host. *Sinorhizobium meliloti* microarrays were also used to explore transposon mutants by DNA-hybridization.

- *Xanthomonas campestris pv. campestris* (Thieme *et al.*, 2005), a plant pathogenic bacterium that causes bacterial spot disease in pepper and tomato plants.

- *Xanthomonas campestris pv. vesicatoria* (da Silva *et al.*, 2002), the causative agent of black rot affecting crucifers (microarray unpublished).

- *Clavibacter michiganensis pv. michiganensis* (Lee *et al.*, 1997), another plant pathogen infecting tomato and other nightshade (*Solanaceae*) plants causing the bacterial wilt and canker disease (microarray unpublished).

- *Mycobacterium tuberculosis* (Cole *et al.*, 1998), the cause of human tuberculosis (microarray unpublished).

- *Streptomyces coelicolor* (Bentley *et al.*, 2002), (microarray unpublished)

- *Corynebacterium glutamicum* (Kalinowski *et al.*, 2003), a bacterium that is industrially used for the production of amino acids (Microarray: Hüser *et al.* (2003)).

- *Corynebacterium jeikeium* (Tauch *et al.*, 2005), a bacterium colonizing the human epidermis (microarray developed by Brune *et al.* (2006a))

- *Neisseria meningitides* (Tettelin *et al.*, 2000), (microarray unpublished)

- *Sorangium cellulosum* (Gerth *et al.*, 2003), So ce56, a model for myxobacteria. Myxobacteria are of scientific interest for their capability to produce low-molecular weight secondary metabolites some of which seem to have remarkable biomedical activity. With over 13 megabases, *Sorangium cellulosum* has the largest prokaryote genome discovered so far (genome and microarray unpublished).

In addition, strain specific *E. coli* microarrays have been manufactured at the University of Würzburg, department for infection biology.

8.2.1 Project Specific Requirements and Results

The GenoMik competence nodes consist of a large number of contractors, working on different bacteria resulting in a large number of array layouts and hybridizations. Many concerns regarding data-privacy have been raised from project partners. As it is not desireable to give rise to a overly large number of separate databases, it was decided that all contractors should share a single EMMA2 project as their common microarray repository. To secure the data, the access control mechanism was applied. Users were assigned into groups and access was restricted to layouts and datasets created by these groups. Each user is individually responsible for setting proper group access rights. No access violations were reported during the project.

As for all whole-genome microarrays generated during the project, genome sequences are available within the GenDB annotation system; direct BRIDGE-links from the reporter sequences to GenDB were established for data integration. The whole-genome microarrays for *Corynebacterium glutamicum* and *Sinorhizobium meliloti* were the first available microarrays within GenoMik. Hence, the largest number of publications making use of EMMA stems from research with these two organisms.

For *Corynebacterium glutamicum*, a spotted microarray (Cg4kPCR) was created from PCR-fragments representing 93% of all predicted coding sequences. To assess the validity of the results, we have performed a pioneering study of gene expresssion during bacterial growth with propionate as carbon source (Hüser *et al.*, 2003). The validation strategy is threefold: First, yellow-experiments were performed to measure the technical variation of the tool. Second, experimental conditions were adjusted such that a portion of genes with known function could be expected to show differential expression. Third, results from the microarray measurements were validated by using real-time RT-PCR measurements of the RNA material. As a result from the yellow experiments, the pure technical variation of the microarray could be assessed.

The correlation between channels was above 0.99 for most technical replicate arrays in the experiments. MA scatterplots were used to visualize the variation of the normalized and transformed data. For normalized data with joint replicate measurements (mean value), the 0.95 quantile range (the range containing 95% of the data) was found to be approximately $M \in [-0.6, 0.6]$. The authors decided to call a gene significantly differentially expressed, if its measurements achieve a significance level $\alpha = 0.05$ and their joint measurements satisfy $M \notin [-0.6, 0.6]$.

Real-time-RT-PCR measurements for a portion of approximately 10% of the differentially regulated genes identified by the microarray experiments were performed. All genes measured were also detected by the alternative method, while the peak expression values of some genes were much higher than in the microarray experiments. The dynamic range of RT-RT-PCR was thus found to be larger than that of the applied microarrays.

In the growth experiments performed in the same study, we could also detect a substantial number of genes with unknown function. This has led to a substantial number of follow-up experiments using EMMA2 and the produced microarrays to elucidate several gene-regulatory networks of *Corynebacterium glutamicum*.

Silberbach and coworkers investigated the response to nitrogen starvation (Silberbach *et al.*, 2005a) by microarray analysis. By a combined approach of transcription measurements, proteomics, chemostat measurements, and sequence analysis they further identified several regulatory genes of the nitrogen control network (Silberbach and Burkovski, 2006). Growth under ammonium limitation is another research target investigated by a combined transcriptomics–proteomics approach (Silberbach *et al.*, 2005b). Further experiments by Brune *et al.* (2006b) resulted in the characterization of a new iron uptake regulator. A combined approach with a remarkable amount of laboratory work led to the unravelling of the regulatory network of sulfur metabolism (Rey *et al.*, 2003, 2005; Koch *et al.*, 2005). Subsequent data-analysis steps with EMMA2 have marked a starting point for initial hypothesis followed by a vast amount of follow-up experiments, like quantitative-RT-PCR, proteomics approaches and DNA-binding experiments (e.g. Rückert *et al.*, 2005).

Hüser *et al.* (2005) analyzed a rationally designed *C. glutamicum* strain genetically engineered for an optimized pantothenate production. In a combined approach, the authors measured growth of the improved strain in comparison with the industrial production strain. Samples for microarray hybridizations were taken at six timepoints. The derived microarray data were normalized using global lowess normalization, and pre-filtered for the clustering step. The authors chose a p-value cutoff of 0.001 and, in addition, applied a filtering step on the standard-deviation. The EMMA2 pipeline excluded those genes, with standard deviations satisfying $1.2 < s/\overline{s}$, where $\overline{s}$ denotes the mean standard deviation of all M-values. Hierarchical cluster analysis of the lowess-normalized microarray data clearly revealed three major clusters, one of which contained the genes altered between the production and the engineered strain and other genes showing regulatory dependencies with them.

Rüberg *et al.* (2003) have constructed a microarray (Sm6kPCR), containing mainly PCR-fragments and few oligonucleotides representing all 6207 predicted protein-coding genes of *Sinorhizobium meliloti*. In addition to quality control experiments with samples grown under the same conditions (self-self hybridizations), the authors investigated the transcriptional response of the bacterium to an osmotic up-shift with the addition of NaCl. 137 genes were identified to be differentially expressed by filter-settings based on previous results. As the experiments showed equally promising results as for the Cg4kPCR microarrays, further follow-up experiments were conducted.

The symbiotic abilities of *Sinorhizobium meliloti* to fixate atmospheric nitrogen together with its host legume plant are the primary research goal within the competence node. Legume plants form specialized root organs – termed nodules – that are colonized by *S. meliloti* cells.

Based on the new Sm6kPCR array developed by the group, Becker *et al.* (2004) performed a global gene expression profiling study using the GenoMik EMMA2 microarray database. In numerous experiments they compared bacteria extracted from root nodules with bacteria grown under micro-oxic conditions. A large number of genes could be found to be differentially expressed between these conditions.

Based on the microarray data confirmed by quantitative-RT-PCR, further attempts were made to elucidate the regulatory networks involved in the nodulation process. Puskás *et al.* (2004) examined global changes in gene-expression under normal and micro-oxic conditions in a deletion mutant of the assumed nitrogen regulatory gene NtrR. Resulting from their data, the authors conclude that NtrR is not a nitrogen but a global regulator.

Within the GenoMik database, Hoang *et al.* (2004) studied the Sin quorum sensing system of *S. meliloti* by transcription profiling of eight different bacterial strains. Quorum sensing is a mechanism of bacterial communication, that depends on the population density and plays a crucial role in symbiosis between the symbiont and its plant host. Based on the resulting data they identified several novel regulatory dependencies.

A new *S. meliloti* microarray (Sm6kOligo) printed from 70mer oligonucleotides, also developed by the group of Becker and colleagues, contains 6223 representative reporters printed in triplicates. The technical variation of the oligonucleotide platform was reported to be even better than with the Sm6kPCR arrays (personal communication). All reporter sequences are mapped on the genomic sequence in the GenDB software and can directly be referred to in the EMMA2 web-interface and all analysis pipelines.

The new Sm6kOligo array was first described in a publication of Yao *et al.* (2004). The authors investigated the effect of two regulatory proteins on the nodulation specific phenotype of two mutants. By the use of microarrays and validation by RT-RT-PCR, the authors were able to gain new insights into the physiological changes of early symbiosis and the regulatory signal transduction pathways (see also Krol and Becker, 2004).

8.3 A Novel Evaluation Framework for Statistical Tests

The most basic question to pose to a replicated microarray experiment is, to identify a set of significantly regulated genes under the influence of known experimental conditions. This question is also relevant for further data-mining steps to restrict the number of genes to a manageable quantity of reliable candidates.

The intention of this experiment is to evaluate different statistical test methods for identifying differentially expressed genes with EMMA2. We also wanted to prove the effectiveness of the system for research in statistical methods. The central question is: Is the integration of new methods as Plug-Ins and their application to existing data as straight-forward for a programmer as required.

The optimal strategy to recommend for a standardized pipeline should be identified. As testing for significant expression is a task with an enormous impact on downstream analyses as no other class of methods, it is of primary importance to protect against application of methods which make assumptions about the data that do not hold, and inappropriate setting of parameters. From the experiences with the previously described projects, it can be concluded, that the researchers used slightly different strategies, all of which included statistical testing and fixed M-value cut-offs, determined empirically. It was, under all circumstances, necessary to evaluate other methods to be able to give clear advice on the preferred statistical inference method to use.

After studying the available literature on the problem of statistical test procedures for microarray data, it became clear that an experimental setup for a neutral and unbiased evaluation was required. Three central arguments for such a study can be given here:

1. Almost all authors comparing different test procedures also propose a new method. And in all cases, this method is favored in the evaluation of its authors.

2. The evaluation procedures differ very much from each other. While some authors use simulated data, others use real experimental examples, some new datasets and some already published. In the case of biological data, the authors try to show that their method produces results that are biologically more meaningful. The notion of *biologically meaningful* is a very subjective one and depends heavily on the nature of the data and the experimental questions.

3. With respect to a standardized procedure, all authors seem to resign, noting that such a gold-standard is hard to achieve. That does not mean it is generally impossible, but can be interpreted as a call such an attempt should be made to re-evaluate procedures.

Another, maybe less formal, argument is, that it seems worth to check if the nature of microarray data is so much different from other data, at all, that so many new methods are required to detect differences in means.

The author would also like to point to the advice given by Allison *et al.* (2006) in their review on methods for the analysis of microarray data: “In many areas (for example, cluster-analysis algorithms, normalization algorithms and false-discovery-rate estimation procedures), the need for thoroughly evaluating existing techniques currently seems to outweigh the need to develop new techniques.”

In principle, the identification of differential expression for one or two conditions boils down to the task to determine, whether the location parameters of unknown distributions deviate significantly from each other or from zero. The classical approach to this problem is Student's t-test if normality of the data under the null-hypothesis can be assumed, or Wilcox' rank-sum test as a non-parametric alternative. Other tests like the Cyber-T method, LIMMA, SAM, and VarMixt have been published recently. These methods have been developed with a focus on microarray-data (see Section 3.4) and in particular try to address the problem of small sample sizes due to limitations in the number of replicates.

The major hurdle for a validation experiment for test-procedures that rank genes according to differential expression is the lack of a gold-standard to compare the results with. There is currently no method known to acquire a 'true' ranking of genes. Neither is there a benchmark dataset against which the performance of test procedures can be evaluated, at least not for spotted arrays.

Qin *et al.* (2004) have investigated the effect of data transformation and ranking procedures on the data using cDNA microarrays with 10 spike-in genes at known concentrations.

Delmar *et al.* (2005) have described a detailed evaluation based on a set of 63 microarrays from a large study on human blood cancer cells. They have used the dataset as if it contained 63 biological replicates to evaluate an estimate of variance. In fact, each dataset in their evaluation has been hybridized with different samples grown under different conditions. As a consequence, the dataset is not a real replicate experiment. A spike-in dataset with Affymetrix microarrays has also been used by Delmar *et al.* to asses false positive and false negative rates. Furthermore, the evaluation is rather questionable for several reasons (see Subsection 3.4.8).

Here, a new approach to compare the relative merits of ranking methods will be considered, which is based on the application of a novel multi-replicate microarray.

The approach follows the assumption, that the common feature of all methods is the generation of p-values and that these p-values can be used to rank genes with respect to their probability of beeing truly differentially expressed. The actual p-value of a sample depends also on the sample size, such that it is impossible to compare the performance of the tools directly by comparison of p-values. On the other hand, the ranks of all gene samples should for all sample sizes remain relatively stable.

The central idea is to compare ranks of all genes for a small sample-size with the ranks of a very large sample size. A good method should resemble the ranking of the large sample also with a small sample.

Another argument for requiring rank stability for different sample sizes can be expressed as follows: Only in the case of stable ranking is it possible to select a set of significant genes for a given significance level for all sample sizes.

Gene	Regulator type	304	306	Expected regulation
SsuR	activator	–	–	no expression
McbR	repressor	–	++	↑ (306/304)
CysR	activator	++	–	↓ (306/304)

Table 8.2: Overview of the two mutations of *Corynebacterium glutamicum* ATCC 13032 strain used in this study. The exact names of the constructs are *Cg* Δ *SsuR* Δ *McbR* P_{neo} *CysR* (304) and *Cg* Δ *SsuR* Δ *CysR* P_{neo} *McbR* (306). – denotes a gene deletion of the HTH motif, to prevent binding to the DNA and thus loss of regulatory function of the transcription factor. ++ denotes a mutant having its native promoter sequence replaced by a strong constitutive promoter.

8.3.1 Materials and Methods

A multi-replicate microarray is a special type of microarray having a large number replicated spots compared to normal microarray designs. The original intent for making this microarray was to measure the optimal dilution of spotted oligonucleotide reporter molecules to achieve the overall best signal to noise ratio. This microarray was spotted using 70-mer oligonucleotides representing 92 unique coding sequences of *Corynebacterium glutamicum* and four external controls. The coding sequences mainly represent genes presumably involved in the sulfur metabolism of *Corynebacterium glutamicum* (Rückert *et al.*, 2005). Each oligo is represented in 80 replicates, comprising eight steps of dilution, resulting in a total of eight fully identical groups of 10 replicate spots per gene. 6 arrays of this design were available for this project yielding a total of 400 replicates per gene. The biological material generated for this experiment was taken from cultures of two genotypes of *Corynebacterium glutamicum* ATCC 13032 (G306 and G304). Both carry triple mutations in the same regulatory genes, genetically engineered by promoter constructs and gene deletion. The three regulators affected are *SsuR, CysR and McbR.* Knockout and overexpression constructs of regulators are a well established technique for many bacteria to cause a regulatory cascade and to study regulatory hierarchies. In this case, some of the affected genes are already known and published in prior studies. Rey *et al.* (2005) published a study on the *McbR* regulon and Koch *et al.* (2005) investigated the *SsuR* regulon. Table 8.2 gives an overview of the expected differential expression of the three genes and their effect on their regulatory network.

RNA from the two genotypes G306 and G304 was directly compared in six hybridizations. The multi-replicate microarray design was provided and hybridizations were carried out by Andrea Hüser. Both strains were grown under identical conditions on minimal sulfurless medium (MMS) with addition of 1mMol Cystein and 1mMol Sulfate. Cells were harvested and RNA extracted following standard protocols developed for *C. glutamicum.* Purified RNA was provided by Christian Rückert. Six microarrays of the multireplicate designs were available for the experiment. Density scatterplots of all data were produced (see Figure 8.1) that exhibit three distinct areas of up-regulated spots, downregulated spots and a smaller area of no regulation.

To compare an approximate false positive rate of the methods, a normal Cg3kOligo microarray design was used and hybridized with in a self-self experiment. RNA was extracted from *Corynebacterium glutamicum* cells in logarithmic growth phase, the RNA was split and labelled with Cy3 and Cy5 dyes (see Figures 8.2 on the next page). The correlation between background corrected channel intensities was generally better than 0.95 for the yellow experiments.

All methods described in Section 3.4 were included in the evaluation: the one-sample t-test (t-test), Wilcoxon's rank-sum statistics (Wilcox), Cyber-T, LIMMA, SAM, VarMixt, and RankProd. In addition, two methods were added for reasons of comparability:

Mean M-value ranker This method ranks all genes simply by computing the absolute mean of all replicates of a gene.

Random ranker This 'method' ranks genes by generating random permutations of all genes. It is added solely for the purpose of illustration. Any real method should perform significantly better than this method for any given number of replicates.

Pre-Processing and Normalization Images were scanned once per array and channel with equal settings and image analysis was performed with the ImaGene 6.0 software. Automated flagging of low-quality or low intensity spots was applied. No manual flagging of bad spots or defects was applied. Statistics on the raw data can be found in Table 8.3. The data illustrates that there are only slight differences in the global features of the two different experiments, hence they should be comparable. Large difference between the mean and median values are caused by a proportion of extreme measurements.

Spots having ImaGene flag *2* (which denotes low intensity) were excluded from the analysis. To minimize possible side effects of complex normalization procedures and to resemble the analysis of a normal experiment, it was decided to apply only background correction, computation of logarithmic ratios and global lowess normalization. According to this settings, M and A-values were computed with the standard EMMA2 pipeline. M vs. A scatterplots were also produced to inspect the data distributions. Spots having $A < 7$ were excluded from the analysis, leaving genes with various numbers of replicates.

Some tests require that the data are normally distributed. It is not necessary, and often not the case in the multi-replicate experiment, to have a single normal distribution for the whole microarray. Instead, it suffices to have gene-wise normal samples. The Shapiro-Wilk-test allows to test for normality. The test was performed separately for each step of dilution, yielding 752 reporters with up to 60 replicates.

From the 714 reporters with a sufficient number of replicates remaining after filtering for flags, only 256 show a significant deviation from the normal distribution

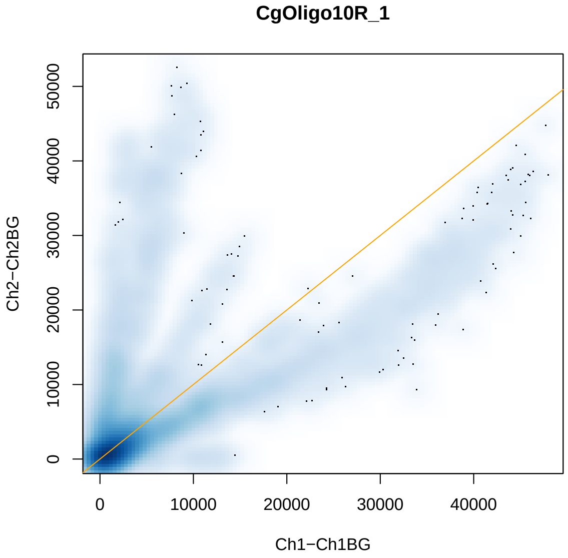


Figure 8.1: Density scatterplot of the raw and background subtracted intensities of one of the multireplicate microarrays. The plot exposed the tri-partide nature of the data distribution, which is due to the fact that the array represents over-expressed (upper partition), repressed (lower partition) and unchanged genes (smaller central partition). The yellow line represents the main diagonal.

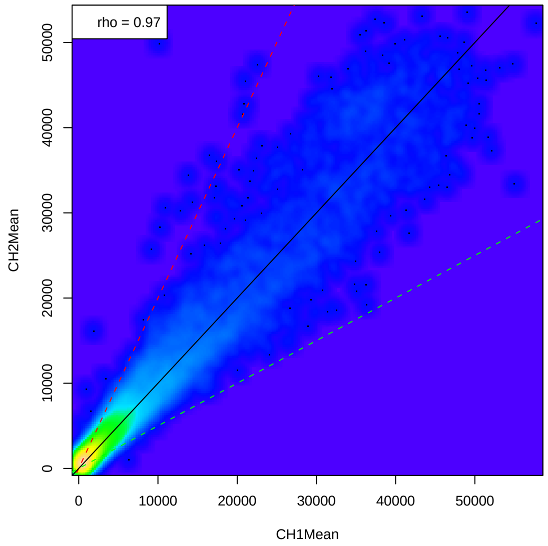
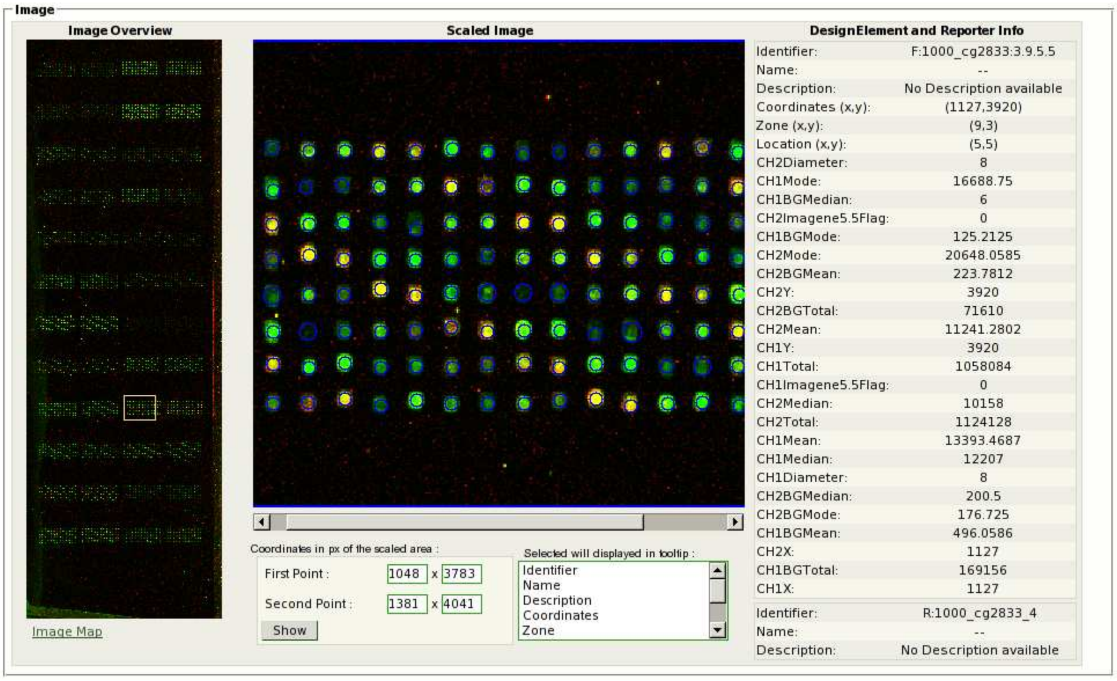


Figure 8.2: A ‘topographic’-density scatterplot of the channel raw intensities of the yellow experiment microarrays altogether. This plot is a density scatterplot using colors similar to a topographic map. Dark blue shades (the ‘deep-sea level’) denote data density close to zero, brighter shades to white denote high density. Most of the data-points clutter around channel intensities of 1000. Variation is very low as Pearson correlation coefficient of channel intensities is greater than 0.97.



Coordinates in px of the scaled area :

First Point : 1048 x 3783

Second Point : 1381 x 4041

Show

Selected will displayed in tooltip :

Identifier
Name
Description
Coordinates
Zone

DesignElement and Reporter Info

Identifier:	F:1000_cg2833:3.9.5.5
Name:	--
Description:	No Description available
Coordinates (x,y):	(1127,3920)
Zone (x,y):	(9,3)
Location (x,y):	(5,5)
CH2Diameter:	8
CH1Mode:	16688.75
CH1BGMedian:	6
CH2Imagene5.5Flag:	0
CH1BGMode:	125.2125
CH2Mode:	20648.0585
CH2BGMean:	223.7812
CH2Y:	3920
CH2BGTotal:	71610
CH2Mean:	11241.2802
CH1Y:	3920
CH1Total:	1058084
CH1Imagene5.5Flag:	0
CH2Median:	10158
CH2Total:	1124128
CH1Mean:	13393.4687
CH1Median:	12207
CH1Diameter:	8
CH2BGMedian:	200.5
CH2BGMode:	176.725
CH1BGMean:	496.0586
CH2X:	1127
CH1BGTotal:	169156
CH1X:	1127
Identifier:	R:1000_cg2833_4
Name:	--
Description:	No Description available

Figure 8.3: False color image of both channels of a multireplicate array stored in EMMA2. Diagonal bands of identical replicates are clearly visible.

with $\alpha = 0.05$. From this result it can be concluded, that evidence against normality of the microarray data used is not strong enough to vote against parametric t-test methods. In any case, some expression values are significantly not normal, which is advantageous for this experiment, because the methods can prove robustness against such deviations.

Generation of a Standard-of-Truth The essential thought of generating a standard-of-truth (SoT) can be explicated as: Obtain a ranking of genes based on a very large sample, in order to compare results from smaller samples with it. Unfortunately, it cannot be assumed that all methods agree completely on the ranks of all genes for the large sample set. Even worse VarMixt is known to yield results that might vary between multiple runs of the method. It is possible in principle, that there is a high level of disagreement between any two methods. In such a case, the generation of the SoT would be impossible. On the other hand, one could expect that at least all methods which rely on simple or modified t-statistics should agree on the ranking for large sample sizes.

First, a measure of agreement for the p-values needs to be defined. As it was stated before, the actual p-value depends on the sample size for all methods and might also vary between methods. Also, we cannot introduce the notion of a 'true' p-value even for given sample size, as the underlying distribution is unknown.

Therefore, Spearman's rank-correlation coefficient is used to assess p-value agreement in the following way: let x_i be the actual p-value result for one out of N genes

	MR	Yellow
Mean CH1	4273	4795
Median CH1	1811	2101
Mean CH2	5561	6022
Median CH2	2159	2378
Mean CH1BG	497	558
Median CH1BG	462	502
Mean CH2BG	337	360
Median CH2BG	303	301
Sd. CH1	6841	8574
Sd.CH2	8192	7461
% flagged	15.5%	17%

Table 8.3: Statistical figures (rounded) on the raw data of the validation datasets. MR denotes the six multi-replicate arrays. Yellow represents a self-self hybridization experiment using four Cg3kOligo microarrays containing all 3000 coding sequences.

$i \in [1, \ldots, N]$ of an application of a test method and y_i the p-value of the same gene for another method or another application. Then, we can assign ranks to each observation x_i, y_i, where R_i denotes the rank of x_i and R'_i denotes the rank of y_i, given all p-values for both tests $(\mathbf{x}, \mathbf{y})$, which may also include ties. Now, Spearman's rank correlation can be calculated as defined:

$$r'(R, R') = \frac{\sum (R_i - \bar{R})(R'_i - \bar{R}')}{\sqrt{\sum (R_i - \bar{R})^2 \sum (R'_i - \bar{R}')^2}} \ , \tag{8.1}$$

where $(\bar{R}, \bar{R}')$ denote the arithmetic mean of all ranks of $(\mathbf{x}, \mathbf{y})$.

In case of a repeated experiment, R_i and R'_i in Equation 8.1 have to be replaced by $\hat{R}_i$ and $\hat{R}'_i$ which denote the arithmetic mean of ranks over all repetitions for gene i.

Two experiment settings were formed by different grouping of reporters within the array design:

Joint replicates (JMR) For the first experiment, all steps of dilution for the same gene were jointly assigned to one reporter sequence, yielding 80 replicates per array and a total of 500 replicates for the MR and MR-bad datasets.

Distinct replicates (DMR) For the second experiment all steps of dilution were treated as individual reporters, yielding 10 replicates per array and 60 replicates per data-set.

In a first step, self-consistency of VarMixt was evaluated on the JMR dataset. The method was applied to the complete pre-processed multi-replicate datasets for 10 times. All repetitions were cross-compared using rank correlation. Astonishingly, self consistency was generally not better than $r' \leq 0.8$. To verify this result, VarMixt was applied to the DMR-experiment and self-self experiment as well. The results are slightly better for the DMR data and much better ($r' \approx 0.9$) for the self-self data; it seems that the stability depends on the number of genes available. This might be caused by the variance of the EM-estimation used for finding the variance parameters, in that a lack of supporting data-points might lead to unstable model estimation. Also, manual setting the variance classes size to 3 or 4 did not improve the result. As a result, VarMixt was excluded from the generation of the SoT-dataset.

To provide enough replicates, genes having less than 20 quality spots were excluded from the analysis. Default parameters were used for the parameter *conf* as the influence of the *conf* parameter seems marginal. Different settings were compared on the JMR experiment and rank correlations $r' \geq 0.99$ were achieved.

Algorithm 1 Compute SoT

```
for all datasets $\mathbf{d} \in \mathfrak{D} = \{$JMR, JMR-bad, DMR, DMR-bad $\}$ do
  for all methods $m \in \mathfrak{M} = \{$t-test, Wilcox, CyberT, LIMMA $\}$ do
    for all genes $g_i, i \in [1, \ldots, n]$ do
      $p_{i,m,d} \leftarrow m(\overrightarrow{d_i})$
    end for
    $\mathbf{R}_{d,m} \leftarrow \text{rank}(\mathbf{p})$ /* compute ranks */
    $\bar{\mathbf{R}}_d \leftarrow \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} R_{d,m}$ /* compute mean rank */
  end for
end for
```

Algorithm Compute SoT describes the generation of the SoT dataset. The idea is rather straight forward: compute the ranks for all representative methods ($\mathcal{M}$) for all genes on all four datasets. After that, let the mean rank of a gene be its 'true' rank for the dataset in the evaluation.

Evaluation Procedure Evaluation of rank stability was carried out by applying each evaluation method to each of the two multi-replicate datasets (JMR and DMR). Sample size was artificially reduced by randomly drawing samples between 2 and 20 replicates in size from the large sample. The samples were drawn irrespective of the array on which they are located within the dataset. To compensate for effects of randomness, this process was repeated 10 times for each sample size without reusing the already drawn samples. The rank for each of 10 repetitions was compared against the SoT data using Spearman's rank correlation coefficient (Equation 8.1).

To be fair to all methods, the following conditions were met:

- All random samples were equal for all methods within each repetition.
- Only the random sample data were visible for each method.

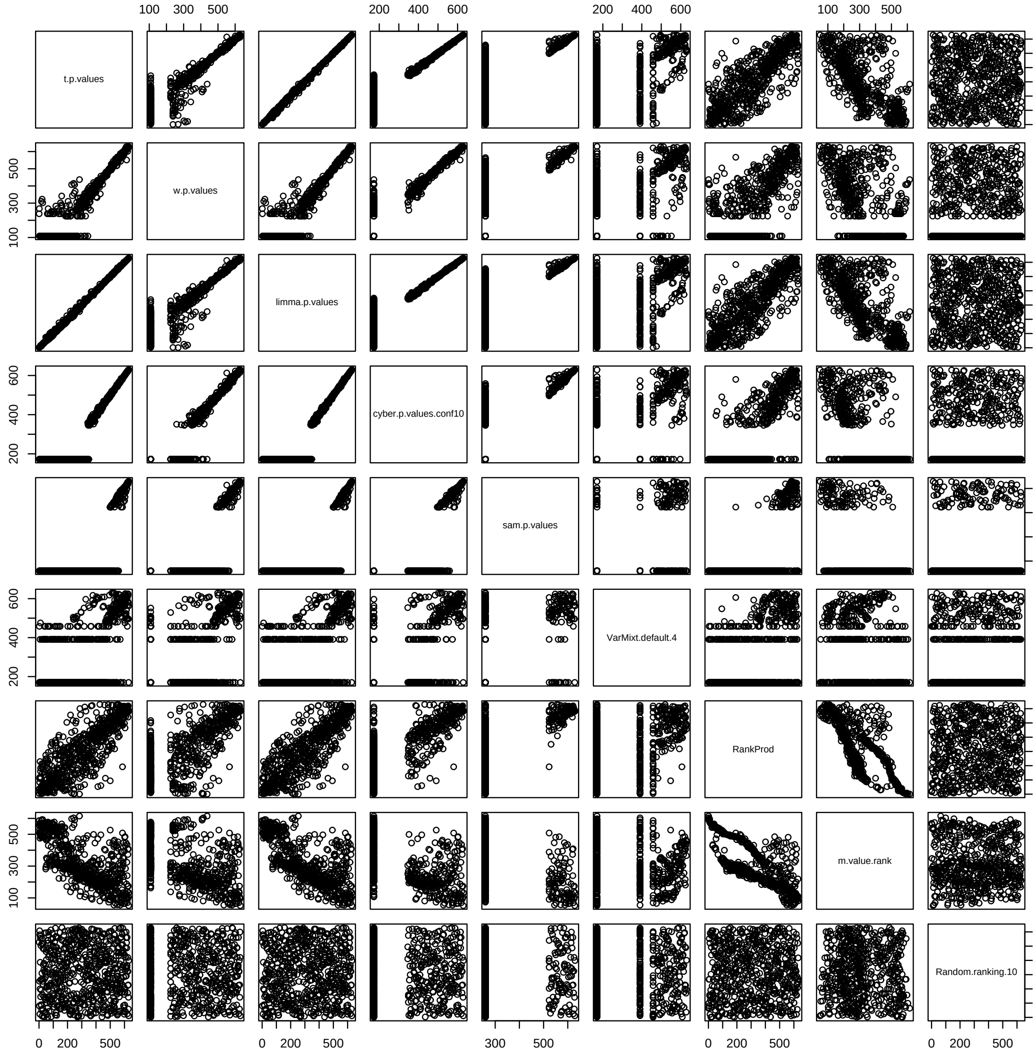


Figure 8.4: Rank comparison plot of all methods included in the generation of the SoT dataset for the DMR data. Each point depicts the rank of a p-value for gene computed by the test method. It is clearly visible, that the methods t-test, Limma, Wilcoxon, and CyberT produce rather similar rankings. Some methods like Wilcoxon and CyberT produce a large number of equal p-values, visible as straight lines. SAM, VarMixt and RankProd conform much less with the other test procedures.

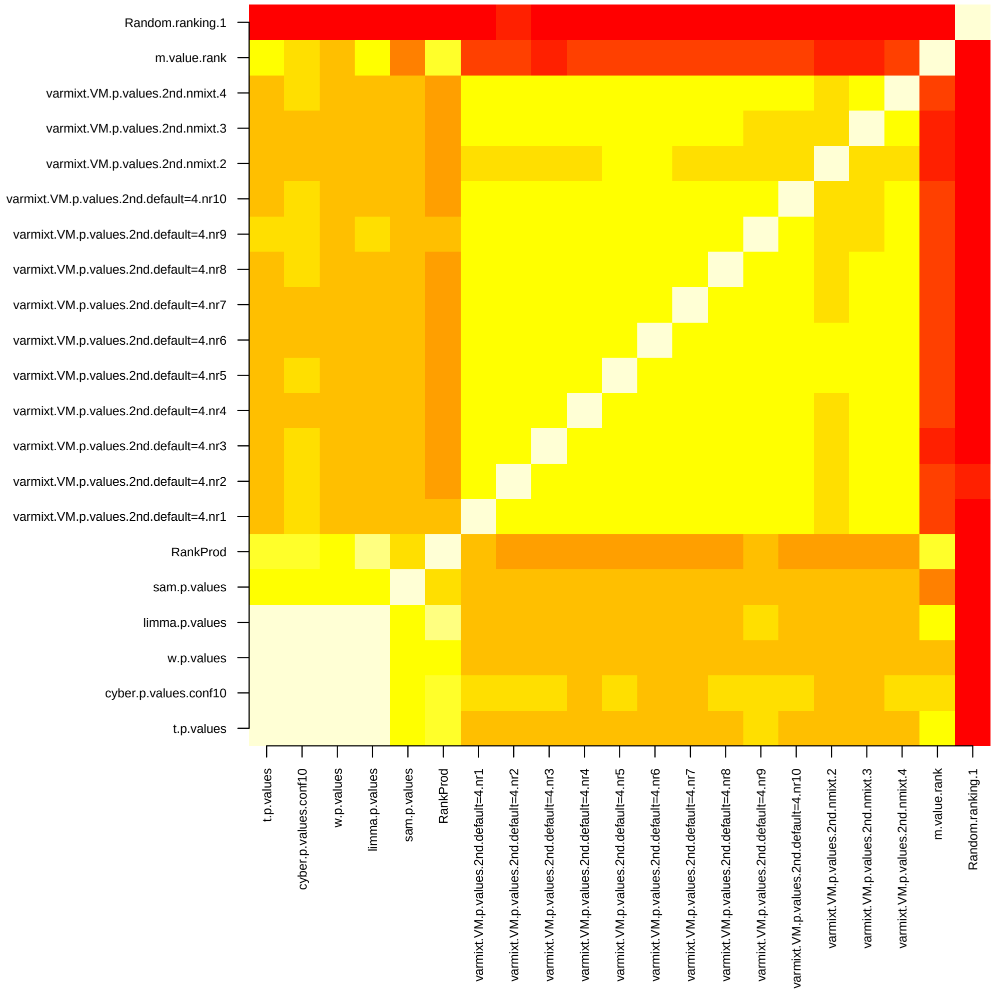


Figure 8.5: Correlation plot as a comparison of the rank concordance between methods in the DMR datasets. White depicts high correlation (≈ 1) and darker shades depict lower rank correlation. Very high correlation can be observed between all t-statistic methods (t-test, CyberT, Limma, partially SAM) and also Wilcoxon's test. Self-consistency for VarMixt has been tested by repeating it ten times. The correlation between individual runs is not much better than between VarMixt and the t-test.

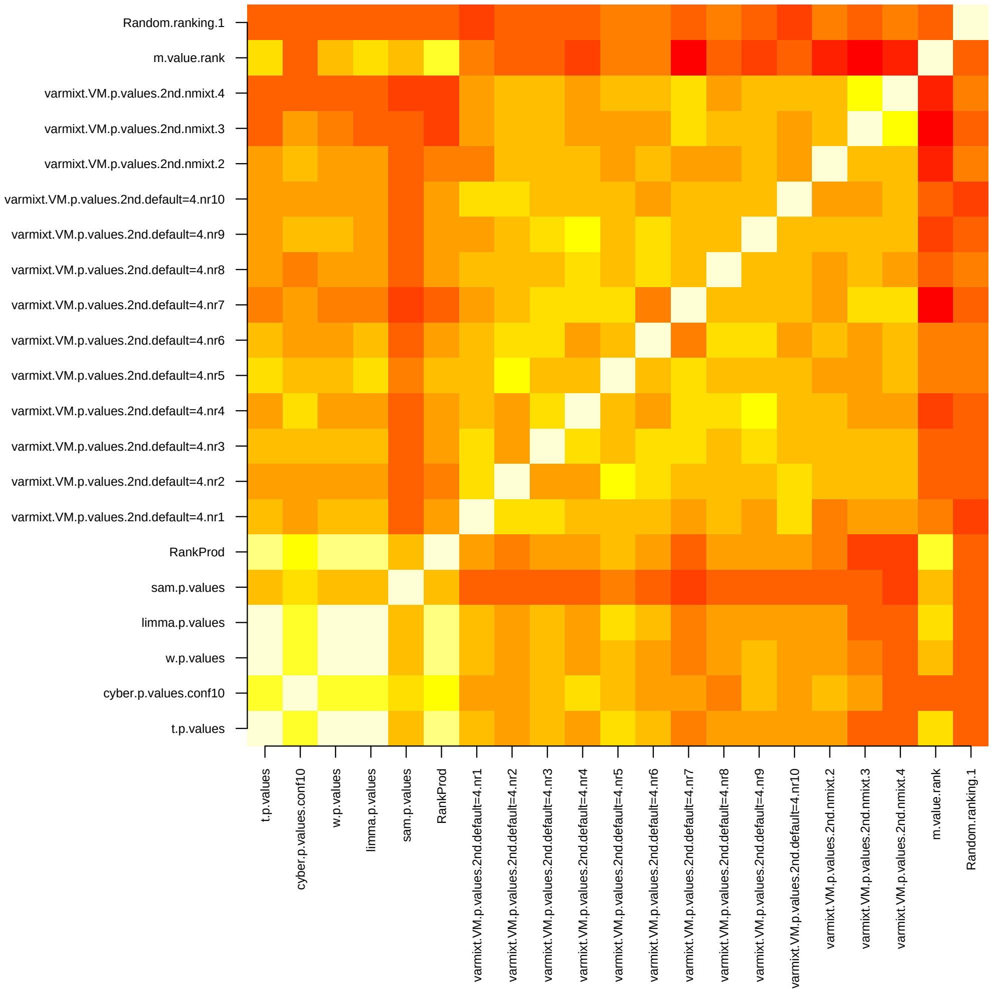


Figure 8.6: Correlation plot to compare rank concordance of all methods on the JMR dataset, consisting of 87 genes with a maximum of 400 replicates.

- For methods (like CyberT) requiring additional measurements and statistics, these were computed, but based only on the random sample.
- Parameters for CyberT, VarMixt and LIMMA were set to the default heuristic
- LIMMA was provided with information about technical and biological replicates, but as results were worse with the additional information, was set to treat all replicates equally.

Algorithm 2 Evaluation procedure (For the sake of simplicity, no differentiation between datasets ($d \in \mathfrak{D}$) is made.)

```
for all sample sizes $l \in \{2, 3, \ldots, L\}$ do
  for all repetitions $j \in \{1, \ldots, R\}$ do
    for all genes $g_i, i \in \{1, \ldots, N\}$ do
      $\overrightarrow{d}_{l,i,j} \leftarrow \text{sample}_l(\overrightarrow{d}_{g_i})$ /* draw gene specific sample of size $l$ */
    end for
    for all methods $m \in \mathfrak{M}$ do
      for all genes $g_i, i \in \{1, \ldots, N\}$ do
        $p_{i,m,j} \leftarrow m(\overrightarrow{d}_i, j)$ /* Compute all p-values */
      end for
      $\mathbf{R}_{m,j,l} \leftarrow \text{rank}(p_{\bullet,m,j})$ /* and rank them. */
      $r'_{m,j,l} \leftarrow r'(\mathbf{R}_{m,j,l}, \bar{\mathbf{R}})$ /* Compute rank correlation with SoT
      according to 8.1 */
    end for

  end for
end for
```

8.3.2 Results

Estimation of Empirical Type I Error Rate To estimate an empirical type I error rate, the Yellow dataset was used and all methods were applied to it. As it was a self-self experiment, under optimal conditions no gene should be called differentially expressed by any method. In this experiment, we provide the tests with a situation, where we know *a-priori*, all null-hypothesis must hold. Still, there can well be the probability of seeing a constant differential expression in the normalized data. Anyway, theoretically and under the assumption that all null-hypothesis are true we can expect a proportion of false rejections as big as our significance level α controlling the type I error rate (see Section 3.4.1).

The empirical type I error rates achieved for each method giving p-values, were measured for $\alpha \in \{0.05, 0.01\}$. As there are not enough replicates, Wilcox' method yields no significant hit at all, hence it also has zero power to detect anything at

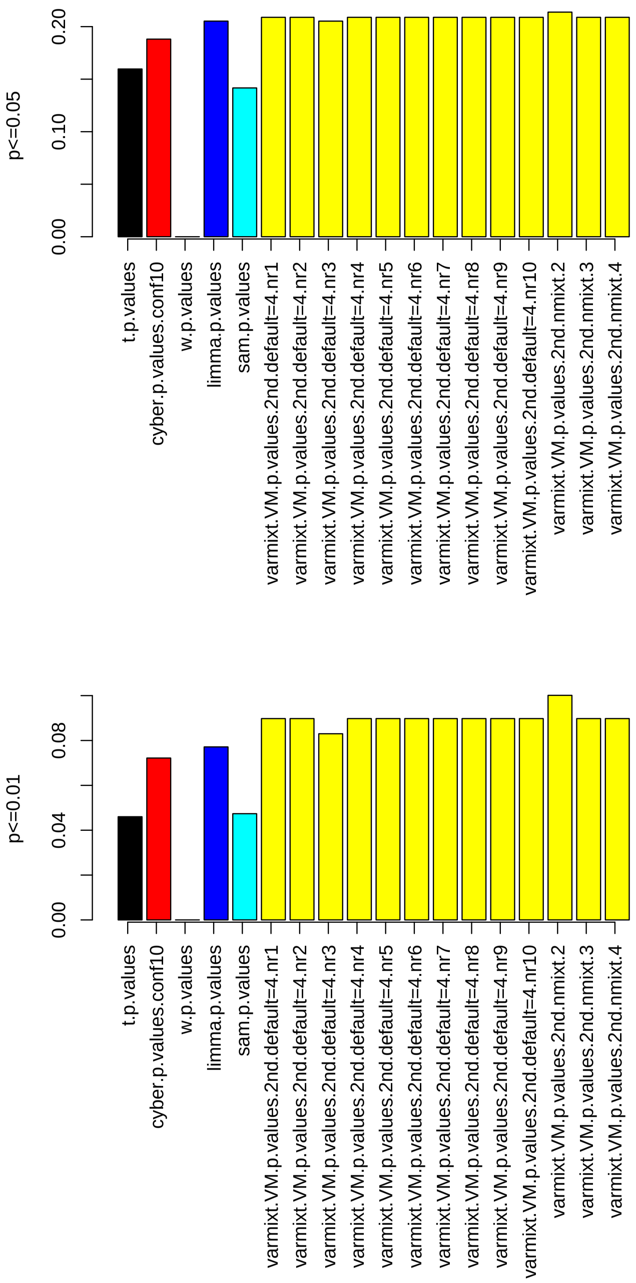


Figure 8.7: Barplots of an estimated empirical type I error rate(four replicates). The proportion of genes with p-values below 0.05 (upper) and 0.01 (lower) is depicted as a bar for each method. 10 repetitions of the VarMixt method and three additional passes with different settings for the number of variance classes are depicted in yellow. With respect to the type I error, t-test and SAM perform preferable, while sample size is too small for Wilcoxon's test.

this threshold. For all other methods the empirical type I error vastly exceeds the nominal one. A likely reason for this is, that all methods underestimate the empirical variance in the data, thereby yielding too large test-statistics.

The best result for the $\alpha = 0.05$ threshold is achieved by SAM (0.142), followed by the t-test (0.160). All other methods perform worse (see Figure 8.7). For the $\alpha = 0.01$ threshold, the t-test (0.046) is nearly equal to SAM (0.047). At each threshold, CyberT, LIMMA, and VarMixt are outperformed. Interestingly enough, CyberT cannot profit from the regularization of the variance estimate, even with a large *conf* parameter, assigning high confidence to the overall variance.

This result is also in accordance with some of the results of Delmar *et al.* although the authors used adjusted p-values. From their results, it can be seen that all methods yield an empirical higher error rate than the nominal FDR value of 0.01 the authors used as a cutoff. While the t-test in their publication achieved the best control over the FDR (with a adjusted p-value of 0.03) compared to the other methods, this was in fact not mentioned by the authors.

Evaluation of Empirical Rank Concordance The rank concordance with the SoT is depicted in the two line-graphs. Most methods (t-test, CyberT, LIMMA, Wilcox) appear to converge to a value of 1 for larger numbers of replicates. Figure 8.8 shows line plots of the rank correlation for both datasets. For all methods small samples provide limited capacity to reproduce the rank ordering of the SoT when compared to larger samples. This holds in particular for the t-test, which is outperformed for two replicates by any other method, except Wilcox' test.

On the other hand, CyberT provides the best rank-concordance, followed by LIMMA and SAM. This finding contributes evidence to support the application of methods using regularized t-statistics. It is remarkable that the SAM test resembles the SoT so well for low sample sizes, although it did not contribute to it; SAM resembles the behavior of a t-test for small sample sizes well on this data set.

In combination with its low empirical false positive rate, it can be suggested as the method of choice for very small samples. CyberT can also be used to achieve an optimum in rank-concordance, while accepting a higher probability of false discoveries. These recommendations are certainly to be treated with some care, as the data might not be completely representative for all microarray studies.

8.4 Stress Response Analysis of Sinorhizobium meliloti Within the BACDIVERS Project

The BACDIVERS project is an international project financed by the European Commission. The project focuses on the detection and elucidation of bacterial diversity of legume symbiotic agrobacteria with respect to environmental stress conditions. These bacteria (mainly Rhizobia) fixate atmospheric nitrogen and make it available for their plant host.

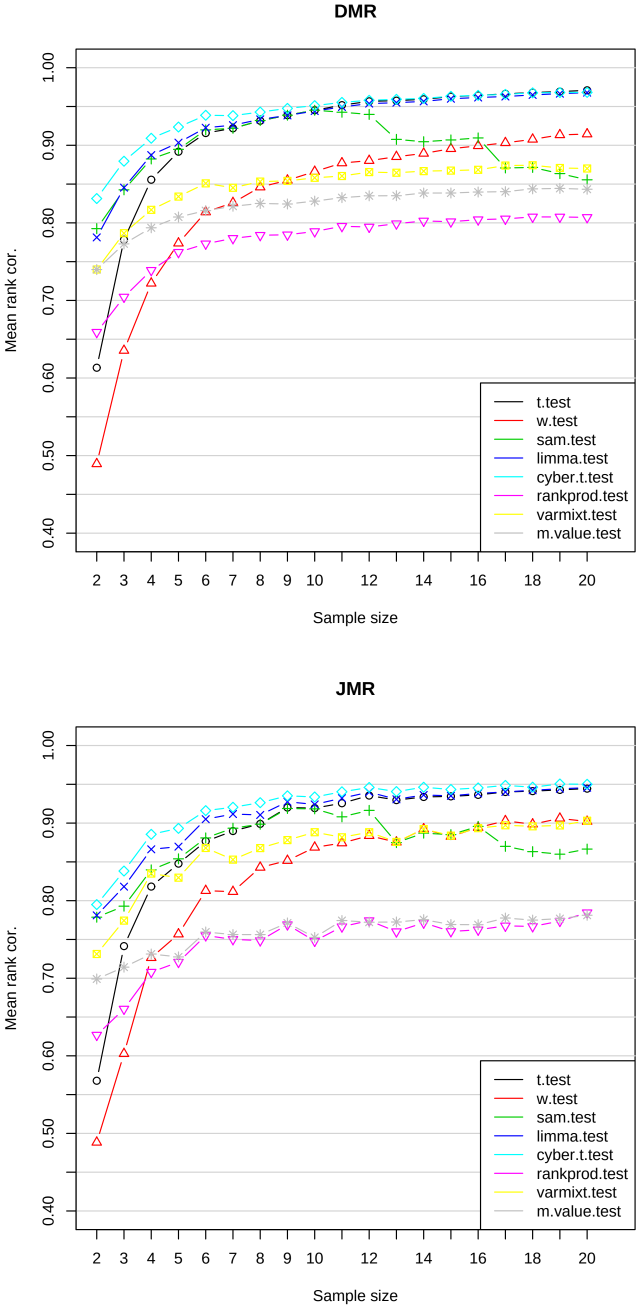


Figure 8.8: Plot of the empirical rank-concordance on the DMR and JMR datasets over the number of replicates. The t-test like methods exhibit a similar behavior of converging to 1. It is clearly visible, that on small sample sizes most methods perform better than the t-test, whereas for large sample sizes all t-test like methods perform comparable.

The principle question behind this project is to elucidate genes of the symbiont involved in the response to suboptimal environmental conditions. Other efforts aim at the identification of strains which can easily adapt to high salt concentrations, low pH, or starvation. This could in principle help to optimize plant growth and reduce the amount of fertilizers needed in legume agriculture.

In this projects many microarray experiments were carried out. The EMMA2 software was used to build a central microarray repository for the whole project and to analyse the datasets from the hybridizations. The largest portion of microarray experiments was carried out for expression profiling, while other arrays were used for strain detection.

In the following section, the analysis of two microarray experiments to measure the influence of pH-Stress and osmotic stress using the EMMA2 software is depicted. The analysis of the microarray datasets was focused on the automated incorporation of external knowledge like biochemical pathways and COG categories. These annotation data are interpreted as categorial variables, that provide class labels for a subset of the analyzed genes. The incorporation of external knowledge is accomplished using a BRIDGE-based connection to the genome annotation system GenDB. The incorporation of the sequence annotation can be done automatically within an analysis pipeline. Without such a feature, it would be required, to retrieve annotations from external sources by BLAST runs against KEGG or COG and this step would certainly involve manual curation effort for each gene of interest. Even then, incorporation of the most up-to date annotations would not be an option, as the most recent annotation of the *Sinorhizobium meliloti* genome is held in the GenDB database. In contrast to this manual approach, the aim of using EMMA2 in an integrative approach with GenDB is to profit from the already existing curated and reliable annotation within an automated data-analysis step. The following sections will highlight the practical benefit from such an integrative approach.

8.4.1 Project Setup

The project was set up and initialized using the standardized procedures. The database was initialized using the the auto-generated SQL-files and applying the general optimization file. The function repository was populated by importing analysis function definitions from the TOOLS-ML file. Additional functions for the data-integration tasks like KEGG-boxplots, which are not mandatory for other stand-alone projects, were added. The project administrators used the imported analysis functions to set-up pipelines for use within the project.

The array layout for the Sm6Oligo microarrays produced at the Center for Biotechnology was imported by the project administrators. Therefore, an Array Description Format (ADF) file was created from the spotter files containing for each spot the reporter information and the oligonucleotide sequences. The sequence objects in the EMMA2 database were then linked to the corresponding region objects in the GenDB *S. meliloti* project using the BRIDGE component.

8.4.2 Experiment Design

The approach of this survey was to search for hidden regularities within the gene-expression patterns of *S. meliloti* under stress conditions. Therefore, a microarray experiments consisting of 18 spotted oligonucleotide microarrays (Krol and Becker, 2004) was set up. A culture of *S. meliloti* was grown under low-pH conditions and compared to a culture grown in normal medium. Samples from the culture under the treatment condition were taken at 6 consecutive timepoints. A sample from the normal medium was taken as common reference. The labeled extracts from all timepoints were competitively hybridized against the reference sample, giving a total of three replicates per timepoint, one with the labeling reversed. The experiment was conducted and the experimental data were collected by Christoph Hellweg at the CeBiTec (unpublished).

For the osmotic stress experiment, cell-cultures of the S. meliloti wild-type were grown under two different concentrations of sodium chloride, namely 300 mM and 400 mM. Three biological replicates of the cell cultures were grown and sampled individually. Samples from the cultures grown at 400 mM were harvested at five non-equidistant timepoints (0, 30 s, 15 min, 30, 60 and 240 minutes), samples grown at 300 mM were harvested at only 3 timepoints (0, 30 and 60 minutes). Thus, the experiment consists of two experimental factors ('NaCl concentration' and 'Time') with 2 and 6 levels and 9 unique combinations within the experiment. The microarrays were hybridized with the sample extracts against a common reference pooled from all timepoints. Each biological replicate was assigned to a single array, resulting in a total of 27 arrays.

8.4.3 Management of Experimental Protocols and Data

Experimental raw-data were processed using the ArrayLIMS web-interface. The ArrayLIMS project was initialized with the protocols for culture growth, RNA-extraction, labeling and hybridization. For each of the hybridized microarrays, the forms in the ArrayLIMS workflow were filled by the lab-technicians. Images and raw-data quantification files from the Imagene software were also uploaded to the ArrayLIMS system, for import to EMMA2.

After uploading the data into the respective EMMA2 project, preprocessing and global lowess normalization was applied as described in Section 3.3.2.

8.4.4 Identification of Significant Functional Categories

The data from the microarray experiment were analyzed by using multivariate analysis of variance (MANOVA), principle component analysis (PCA), and subsequent cluster-analysis to identify genes of particular interest. Prior information about the from the genome annotations stored in the GenDB project were available for a fraction of genes. No strong hypothesis about probable correspondence of gene expression profiles could be stated before the analysis, nevertheless functional classification of the genes was present within the database and should be used. For *S. meliloti* the functional classification of COG, enzymatic reaction pathways (KEGG) and functional subsystems of Overbeek and coworkers were available within the annotation system.

A sequence of EMMA2 pipeline functions was applied to preprocessed datasets. First, it was necessary to find out if the available datasets were at all containing enough information to distinguish between different classes for each of the annotation formalisms. For a multivariate microarray experiment, the classification is the response variable and the predictor variable is a vector of length 6.

Thus, a MANOVA pipeline was applied to compare multivariate inter-class with intra-class variances. Therefore, only genes having a sensible annotation for any of the classification schemes were retained. After applying the filtering step, 2737 genes were identified within one of 21 COG categorie, 532 genes could be identified to belong to a KEGG enzymatic reaction; a rather small subset of 99 sequences were annotated to belong to one out of 8 functional subsystem. The functional classification was automatically extracted from the GenDB annotations via the BRIDGE communication layer during pipeline execution.

For the identified genes, replicate measurements where joined by their arithmetic mean. Each joint gene-expression vector was assigned its class label and tested against the null hypothesis, that there was no difference in expression profiles between classes. The null hypothesis could be rejected for COG, KEGG, and subsystems. This can be interpreted as there is at least one group, category or pathway for which the expression data differs significantly from the mean, or in other words some aspect of the prior annotation knowledge is reflected in the data. In contrast, if no external knowledge was available, the only sensible analysis step would have been to identify genes which show differential expression by applying a genewise statistical test.

The next question to investigate is, which classes do in fact react to the experimental condition. The ANOVA approach only provides information about whether there is at least one class with significant reaction, but not which or how many. Box and whisker plots are suitable means to address this problem by visualization, if there not to many classes to compare. In addition to the information about the mean expression profiles for all genes in a group, they provide a visual measure of the within group variation. For each functional subsystem in GenDB, a boxplot visualization was plotted. Four types of reaction patterns could be identified by inspecting the plots: 1. subsystems that show no response, 2. subsystems that show a directed response over time, they tend to be either downregulated or up-regulated with a clear trend, without increased variance,. 3. subsystems that show a trend and also an increase in variance, 4. subsystems that show no clear trend, but react with increased variance.

Most subsystems showed an increase in variance with the treatment. The biosynthesis of the amino-acid tryptophan , however, could be shown to exhibit no response to the stress condition. This is a behavior contrasts other amino-acid synthesis like methionine that show down-regulation and an increased variance. At this step, the hypothesis, that regulation of tryptophan synthesis is markedly different from regulatory mechanisms of other amino-acids, could be readily derived from the automated pipeline.

The Stable-Pathway Approach

In order to apply the boxplot method to all 87 KEGG-pathways, it was necessary to apply data reduction for an appropriate visualization. Boxplots of complete expression profiles would be confusing, but this problem can be solved by dimensionality reduction. Reducing the dimensionality to the principle direction of variance provides a means to plot only one bar per pathway.

Therefore, principle component analysis was applied to find the direction of maximum variance within the data. The main focus of this analyis step was, to detect pathways, which are rather stable in their expression profiles, while eventually having a common trend in their expression. The original expression data were projected on the first PCA component, which described more than 80 percent of the experimental variance, for this experiment. By analyzing the factor loadings of the first principle component in a plot, it turned out, that the first component is a linear combination of all original axes with almost equal weights.

The boxplot of the first component were then made and visually inspected for interesting groups (see Figure 8.9 on the following page). It appeared that the vast majority of pathways exhibits an increased variance, like for the subsystems plots, but a small subset remains relatively stable. To automatically detect these pathways, the variance of each pathway s_{P_k} was compared to the overall data variance s_D by a F-test. The F-test computes the ratio

$$F = \frac{s_{P_k}}{s_D}$$

and compares it to a given value. The alternative hypothesis was, that $F < 1$, so to find groups having variance significantly smaller than the overall variance.

The stable pathway pipeline was added to the function repository of EMMA2. It returns a list of p-values and adjusted p-values using Bonferroni's method for each pathway. As a result, a small number of stable pathways could be identified as significant with $p \leq 0.05$. The top-scoring pathway with respect to its p-value is the *Glycolysis/Glyconeogenesis* pathway; it is also the only significant finding after bonferroni correction.

As a conclusion from the previous findings, it seems reasonable to further investigate the Glycolysis/Glyconeogenesis pathway. In the next step, a lines-plot of the expression profiles of this pathway was produced. The plot exhibited an increase in the M-values of the last timepoint for the majority of the genes in this pathway. When each of the genes is tested individually by a t-test, however, no significant change in expression was detected for any of them. This fact assures the hypothesis, that there are relevant findings for the analysis of per-pathway expression profiles, which cannot be made on a per-gene basis and can at least to some extent be automatically detected by an analysis pipeline.

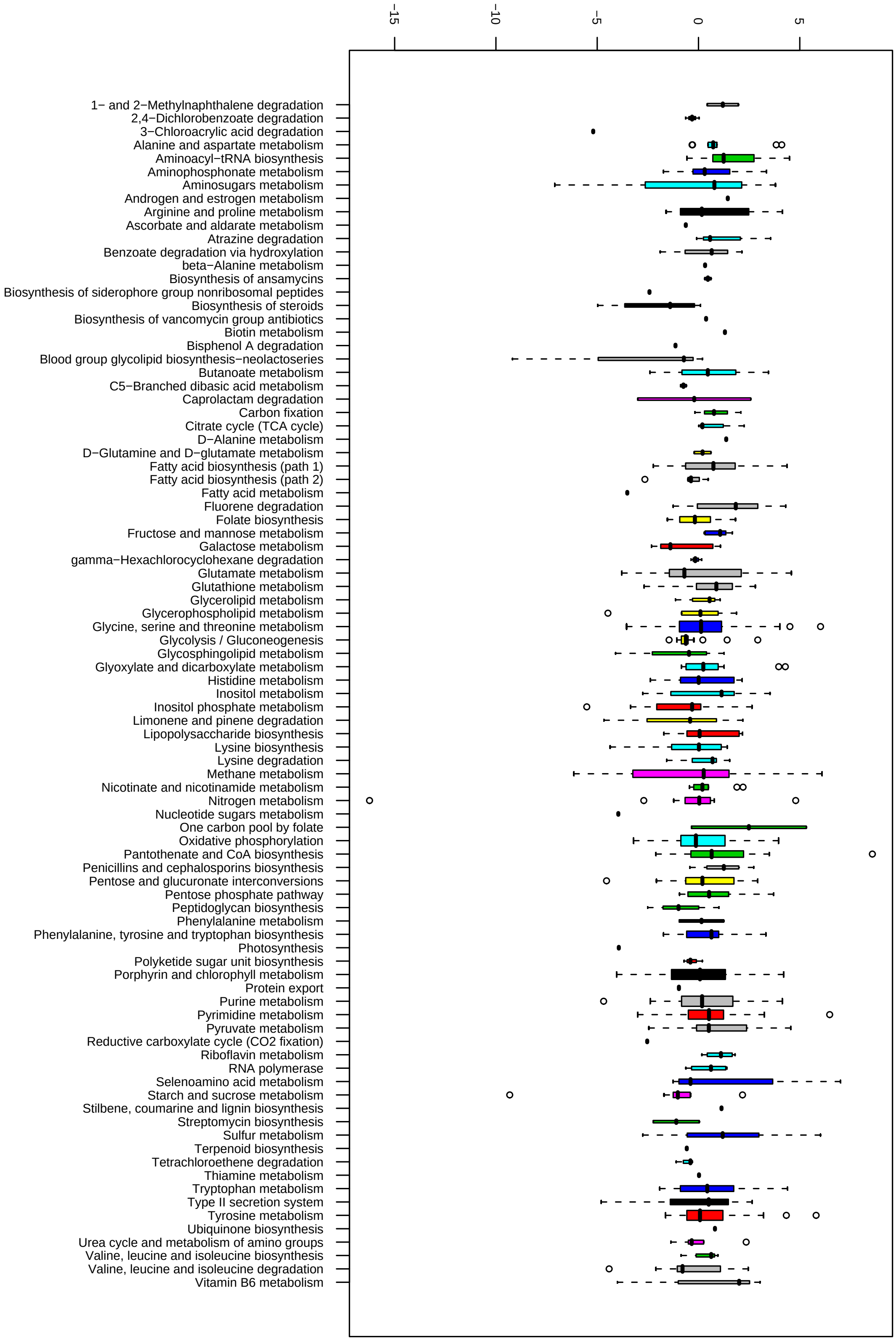


Figure 8.9: Boxplot of the data projected on first principle component for each KEGG-pathway.

To illustrate the response to stress for the Glycolysis/Glycogenesis pathway, the expression values of the genes were projected on the KEGG-map using the KEGG-mapping tool in EMMA2 (the complete map is depicted in Figure). It is easy to detect the corresponding genes and their metabolites which are relevant for further measurements.

The last step in the analysis process was, to try to correlate the expression profiles of the single stable pathway to other measured data from functional genomics experiments. Therefore, metabolic profiling and proteomics experiments were conducted. With the help of these, the level of confidence in the significance of the Glycolysis pathway was increased, because measurements of protein and metabolite concentrations showed good correlation and exhibited an increase at the last measured time-point.

8.5 The Plant Microarray Databases

Three projects which use EMMA2 deal with functional genomics of plants. As they are closely related, they will all be portrayed together. All three projects have in common the use of *Medicago truncatula* ('barrel medic'), a legume plant, as their primary model organism. Like the vast majority of terrestrial plants, *Medicago truncatula* enters beneficial symbiotic plant-microbe interactions. Symbiosis with soil fungi (mycorrhiza) is seen as the by far most widespread form of this sort of beneficial co-existence. Furthermore, legume plants have the unique capacity to establish a nitrogen-fixing root nodule symbiosis with endosymbiotic bacteria, collectively termed rhizobia; one of which is *Sinorhizobium meliloti* (see Sections 8.4 and 8.2). The perspective to study a host-symbiont system provides the primary rationale to foster research in legume plants.

The project “MolMyk: Molecular Basics of Mycorrhizal Symbiosis”, funded by the DFG is focused on the study of mycorrhiza. To study the transcription profiles of these legume plants without having the complete genomic sequences of the organisms, Expressed Sequence Tags (ESTs) were generated (Journet *et al.*, 2002). They were stored and annotated using the SAMS software. A complete review on the bioinformatics tools available in MolMyk, including SAMS and EMMA2 is given by Küster *et al.* (2006).

A set of PCR primers were designed from the EST sequences to obtain a set of PCR-fragments of the Root Interactions Transcriptome (RIT). Mt6k-RIT microarrays and macroarrays were constructed with PCR fragments, based on a collection of 6359 EST sequences in triplicates on glass slides and nylon membranes.

Based on two random cDNA libraries derived of developing flowers and pods, 1776 *M. truncatula* cDNAs representing non-redundant EST-clusters were PCR-amplified and used to upgrade the Mt6k-RIT collection to an Mt8k cDNA set representing approximately 6300 different *M. truncatula* genes (Firnhaber *et al.*, 2005). Mt6k-RIT and Mt8k micro were shared between the participants of the project Molmyk as well as the European Union genome project “MEDICAGO:

Integrated structural, functional and comparative genomics of the model legume *Medicago truncatula*".

The Grainlegumes Integrated Project (GLIP) is an international project co-funded by the European Commission (http://www.eugrainlegumes.org); it is based on the activities of the MEDICAGO project. An expressed goal of the project is to improve the utilization of legume plants like peas, lupins, and beans as crops in European agriculture for protein nutrition. A problem for European farmers with growing grain legumes is in particular yield inconsistency caused by a variety of biotic and abiotic stresses. This project currently comprises 53 contractors from 18 countries. Microarray data-analysis is a substantial part of legume research within GLIP. Currently workpackage 5.3 'transcription-profiling' is concerned with microarrays and workpage 6.1 'bioinformatics' includes data-analysis and the developement of databases to share the resulting data between institutions that are part of GLIP. Another deliverable in this workpage is the integration of the heterogeneous datasources that emerge during the project with each other and also with plant-related databases from other projects.

Within the GLIP project *Medicago truncatula* and *Pisum sativum* (pea) are used as model organisms. For both plants, the complete genomic sequence has not been finished yet. For *Medicago truncatula* the sequencing project is ongoing, but the sequence and annotations is not stable, yet. Therefore, tentative consensus sequences (TCs) of ESTs have been used to generate 70-mer oligonucleotides for the microarray designs. The use of ESTs and the existence of a only partially sequenced genome mark a fundamental difference between the GLIP project and microbial related projects, where fully sequenced genomes are available. The clustering of ESTs in TCs is subject to frequent changes; direct mapping of reporter oligomer sequences is in principle feasible but not final.

Within the GLIP project, 3 different microarray designs are used. The Mt16kOli1 design (Hohnjec *et al.*, 2005) contains 16,086 unique reporters spotted in duplicates, representing all TCs of the TIGR *Medicago truncatula* gene index[1]. This array design has 33696 features layed out in 4 horizontal and 12 vertical grids.

The Mt16kOli1Plus layout (Thompson *et al.*, 2005) is based on the Mt16kOli1 oligo-set, enriched by 384 reporters representing mainly transcription factors and other known regulatory elements. With 34944 spots, the Mt16kOli1Plus microarray is one of the largest cDNA microarray designs currently available. An overview on EMMA2 and its application in GLIP is found in Küster and Dondrup (2006). A general overview on all transcriptomics and bioinformatics tools provided by the CeBiTec for legume plants is found in the The *Medicago truncatula* handbook (Bekel *et al.*, 2006).

[1] www.tigr.org

Species	**Array design**	**Array type**	**ArrayExpress Identifier**	**Sequences on the array**
M. truncatula	Mt6kRIT	cDNA macroarray	A-MEXP-80	Probes representing 5648 EST-clusters from *M. truncatula* root nodules, AM roots, and uninfected roots (Küster *et al.*, 2004)
M. truncatula	Mt6kRIT	cDNA microarray	A-MEXP-81	Probes representing 5648 EST-clusters from *M. truncatula* root nodules, AM roots, and uninfected roots (Küster *et al.*, 2004)
M. truncatula	Mt8k	cDNA microarray	A-MEXP-84	Mt6kRIT probe set plus 1776 probes representing EST-clusters from *M. truncatula* flowers and pods (Firnhaber *et al.*, 2005)
M. truncatula	Mt16kOLI1	70mer oligonucleotide microarray	A-MEXP-85	Probes representing 16.086 tentative consensus sequences of the TIGR *M. truncatula* Gene Index version 5 (Hohnjec *et al.*, 2005)
M. truncatula	Mt16kOLI1Plus	70mer oligonucleotide microarray	A-MEXP-138	Mt16kOLI1 probe set plus 384 probes primarily representing transcription factors (Thompson *et al.*, 2005)
P. tremula + *A. muscaria*	Pt2.4kOLI1	70mer oligonucleotide microarray	A-MEXP-202	Probes representing 2350 EST-clusters from poplar ECM Uwe Nehls, Universität Tübingen

Table 8.4: Expression profiling tools generated in the MolMyk, MEDICAGO, and GLIP projects.

8.5.1 Project Specific Requirements

One goal of all Medicago related projects is to deliver integrated databases, that store the microarray-based expression data obtained in the course of the project, together with relevant information on the experimental conditions profiled and the protocols used to obtain transcriptome profiles.

Currently the sequencing of the *Medicago truncatula* genome is an ongoing project. Consequently, the reporter sequences of the microarrays were designed against ESTs. From the point of data representation, there are no direct implications. The PCR-primer pairs and oligonucleotide sequences can be directly entered into EMMA2 sequence objects. The EST sequences are organized into tentative consensus sequences by clustering the ESTs. While new ESTs are sequenced or existing ESTs resequenced the assignment of ESTs to TCs might change and hence the annotation of the TCs. The process of re-annotation of ESTs is frequently carried out, creating the need for dynamic updates of the annotations.

The EST annotation should be kept up to date by linking the internal sequence annotations against an external component using the BRIDGE integration component. At first, there was no BRIDGE-aware component suitable for linking the representations against. As the internal data representation of EMMA2 allows for storage of freetext annotations of sequences, this was used as a fall-back. The ESTs were then linked against the TIGR-medicago gene index to facilitate mapping the ESTs on their TCs. The sequence descriptions in EMMA2 were regularly updated by a script. At a later stage, the sequence data was linked against the BRIDGE-aware SAMS application (see below).

Another specific requirement, that emerged early in the GLIP project was a datamining component for sequence annotations. The users should be able to query for expression data from the whole project, not only from specific experiments. The query should be made on known unique sequence identifiers or with a boolean fulltext search within the annotations. This search strategy should resemble search mechanism employed by search-engines for web-pages.

8.5.2 Project Setup

The GLIP-microarray database was set-up using the standardized procedures. Databases for the ArrayLIMS and EMMA2 systems where created using the standard table-definitions. No additional optimizations were required in the first place. The standard role definitions were also found to be sufficient for the GLIP database. The initial project administrator (termed 'Chief') was the only initially registered user. All other users were registered by the administrator on their request.

Concerns of data privacy raised by the user community resulted in the creation of several user-groups making collaboration within an institution possible, while disclosing the data to other users for a limited period of time.

The requested integration of the novel datamining tool created the necessity to modify the database and EMMA2 modules. Due to the flexible modular design, this step required only limited efforts. To make specific data fields searchable, a single full-text index was added to the Description-object of the GLIP-database. A display module and a backend module were added to the standard modules to provide the requested functionality (see Figure 8.10 on the next page). After this implementation step and an automated database-update, the datamining functionality was automatically available within all other projects. Data integration with the ESTs stored in the SAMS system was established by adding BRIDGE URIs to the array layouts in EMMA2.

8.5.3 Results

After the set-up phase, the users were able to upload experimental data to the ArrayLIMS and to use the analysis pipelines autonomously. Currently, the plant databases contain a total of over 300 hybridizations.

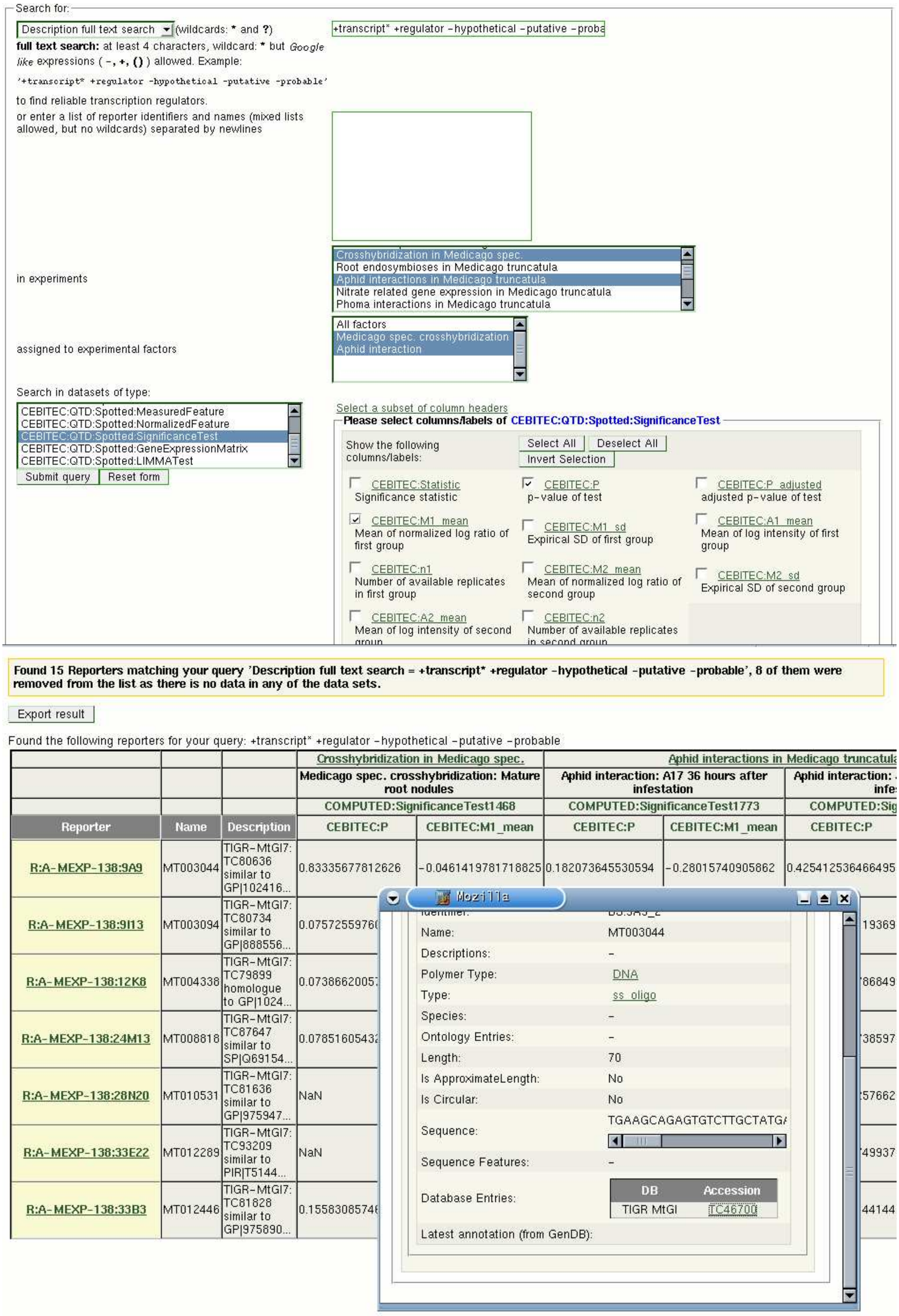
Several relevant findings could be obtained by using the microarray data in conjunction with the EMMA2 analysis pipelines. Within a study of Yahyaoui *et al.* (2004), over 750 genes, including a large proportion of transcription factors, were found to be differentially expressed during root nodulation by using the Mt6k-RIT macroarrays and microarrays. The authors applied a pipeline of normalization and t-test with a combined filtering strategy in combination with hierarchical cluster analysis. By visual inspection of the cluster results, the authors end up with five independent clusters and conclude that there exists a clear switch between a general root-specific and nodule-specific gene expression program.

Based on Mt6kRIT microarray hybridizations, several comparative transcription profiling studies of root nodules and root tissues during AM formation (Küster *et al.*, 2004; Manthey *et al.*, 2004) now allow for a more global comparison of expression profiles during nodulation and formation of mycorrhizza. It was found that the two endosymbioses, although they were known to share common mechanisms, have only limited overlap of their genetic programs, with 75 genes being co-induced in the two interactions.

The article of Firnhaber and colleagues provides insights into the developmental expression regulation during the development of *M. truncatula* flowers and pods. The authors describe the extension of the of the Mt6RIT towards the Mt8k microarrays and their subsequent application to identify more than 700 genes with developmental expression regulation (Firnhaber *et al.*, 2005).

In a recent study, the more comprehensive Mt16kOli1 70mer oligonucleotide microarrays were applied to specify the overlapping genetic program activated by two commonly studied microsymbionts, *Glomus mosseae* and *Glomus intraradices.* In total, 201 plant genes were significantly co-induced at least 2-fold in either interaction (Hohnjec *et al.*, 2005), using normalization functions and statistical analysis pipelines implemented in EMMA2. A set of well-known marker genes were found to be co-activated, thus validating the transcriptomics data (Hohnjec *et al.*, 2006).

As EMMA2 is used throughout all three plant-related functional genomics projects, a cross project integration of the obtained expression data is feasible.



Search for:

Description full text search (wildcards: * and ?)

+transcript* +regulator –hypothetical –putative –proba

full text search: at least 4 characters, wildcard: * but *Google like* expressions (–, +, ()) allowed. Example:

`'+transcript* +regulator -hypothetical -putative -probable'`

to find reliable transcription regulators.

or enter a list of reporter identifiers and names (mixed lists allowed, but no wildcards) separated by newlines

in experiments

- Crosshybridization in Medicago spec.
- Root endosymbioses in Medicago truncatula
- Aphid interactions in Medicago truncatula
- Nitrate related gene expression in Medicago truncatula
- Phoma interactions in Medicago truncatula

assigned to experimental factors

- All factors
- Medicago spec. crosshybridization
- Aphid interaction

Search in datasets of type:

- CEBITEC:QTD:Spotted:MeasuredFeature
- CEBITEC:QTD:Spotted:NormalizedFeature
- CEBITEC:QTD:Spotted:SignificanceTest
- CEBITEC:QTD:Spotted:GeneExpressionMatrix
- CEBITEC:QTD:Spotted:LIMMATest

Select a subset of column headers

Please select columns/labels of CEBITEC:QTD:Spotted:SignificanceTest

Show the following columns/labels:

□ CEBITEC:Statistic Significance statistic	☑ CEBITEC:P p–value of test	□ CEBITEC:P_adjusted adjusted p–value of test
☑ CEBITEC:M1_mean Mean of normalized log ratio of first group	□ CEBITEC:M1_sd Expirical SD of first group	□ CEBITEC:A1_mean Mean of log intensity of first group
□ CEBITEC:n1 Number of available replicates in first group	□ CEBITEC:M2_mean Mean of normalized log ratio of second group	□ CEBITEC:M2_sd Expirical SD of second group
□ CEBITEC:A2_mean Mean of log intensity of second group	□ CEBITEC:n2 Number of available replicates in second group	

Found 15 Reporters matching your query 'Description full text search = +transcript* +regulator –hypothetical –putative –probable', 8 of them were removed from the list as there is no data in any of the data sets.

Found the following reporters for your query: +transcript* +regulator –hypothetical –putative –probable

			Crosshybridization in Medicago spec.		Aphid interactions in Medicago truncatula		
			Medicago spec. crosshybridization: Mature root nodules		Aphid interaction: A17 36 hours after infestation		Aphid interaction: infe
			COMPUTED:SignificanceTest1468		COMPUTED:SignificanceTest1773		COMPUTED:Sig
Reporter	Name	Description	CEBITEC:P	CEBITEC:M1_mean	CEBITEC:P	CEBITEC:M1_mean	CEBITEC:P
R:A-MEXP-138:9A9	MT003044	TIGR-MtGI7: TC80636 similar to GP|102416...	0.83335677812626	–0.0461419781718825	0.182073645530594	–0.28015740905862	0.425412536466495
R:A-MEXP-138:9I13	MT003094	TIGR-MtGI7: TC80734 similar to GP|888556...	0.0757255976				19369
R:A-MEXP-138:12K8	MT004338	TIGR-MtGI7: TC79899 homologue to GP|1024...	0.0738662005				86849
R:A-MEXP-138:24M13	MT008818	TIGR-MtGI7: TC87647 similar to SP|Q69154...	0.0785160543				38597
R:A-MEXP-138:28N20	MT010531	TIGR-MtGI7: TC81636 similar to GP|975947...	NaN				57662
R:A-MEXP-138:33E22	MT012289	TIGR-MtGI7: TC93209 similar to PIR|T5144...	NaN				49937
R:A-MEXP-138:33B3	MT012446	TIGR-MtGI7: TC81828 similar to GP|975890...	0.1558308574				44144

Name:	MT003044
Descriptions:	–
Polymer Type:	DNA
Type:	ss_oligo
Species:	–
Ontology Entries:	–
Length:	70
Is ApproximateLength:	No
Is Circular:	No
Sequence:	TGAAGCAGAGTGTCTTGCTATG
Sequence Features:	–
Database Entries:	DB: TIGR MtGI Accession: TC46700
Latest annotation (from GenDB):	

Figure 8.10: The Datamining Wizard of EMMA2. It allows to search for expression data by a boolean full-text search. The search can be restricted to experiments and conditions. Below the search mask a table containing the results of the search is depicted.

This unique feature of EMMA2 provides the opportunity to gain new insights into the transcriptional program of legume-symbiont interactions by cross-comparison of all datasets.

8.6 The Mamma Carcinoma Microarray Database

Breast cancer is the most frequently diagnosed cancer-type of women. The Mamma Carcinoma project is a joint effort of the Bielefeld Municipal Hospital and the Center for Biotechnology to investigate the molecular mechanisms of breast cancer. The collaborators envision in the long term the ability to predict the outcome of chemotherapeutic treatment of different tumor-types, thereby aiming at the optimization of primary and adjuvant therapeutic regimes. Therefore, tumor tissue samples from patients with breast cancer are collected prior to and after chemotherapeutic treatment, including samples from patients with rezidivs.

For transcription profiling of tumor samples, a custom microarray representing 108 human tumor marker genes is employed. The array layout contains 108 tumor specific marker genes, 48 of which have been identified in a study of van ’t Veer *et al.* (2002). As this layout is rather small, multiple logical arrays can be printed on one physical array. Every logical array can be individually hybridized and accessed with different material. High-throughput custom glass slides from several vendors were tested within the project.

8.6.1 Project Specific Requirements and Results

The small number of genes, represented on the microarray, poses requirements specific to special interest microarrays. For this project, it seems reasonable to assume that in most cases the assumptions underlying a global regression normalization approach are violated. The proportion of differentially expressed tumor marker genes in tumor samples cannot be regarded as small and unbiased.

To ensure the ability to normalsize the data, a large proportion of internal and external controls was included in the array. A proportion of the included controls was annotated as positive controls and expressed in all tissue samples. In theory, these controls should provide a suitable reference for normalization. Therefore, a flexible normalization pipeline to use this control set was required. In the course of the project, the control set had to be re-annotated, because the first-line annotations did not deliver reliable results. Therefore, the project administrator imported a second re-annotated layout. It would have been possible to update the original layout, but the users requested to be able to compare the results obtained from both layouts.

A control-based normalization pipeline was set up for this purpose. This pipeline allows to select different control sets based on their control type. Positive controls do not allow intensity dependent normalization. The available functions, were restricted to the mean, trimmed mean, and median of the control data distribution.

The trimmed mean allows to remove a configurable proportion of extreme values, making the method robust against outliers.

To validate the results, control-based normalization was applied to Sm6kOligo whole-genome microarrays. There were only minor differences in the obtained M-values, and very high correlation was detected. No gene was significantly different between both methods at significance level of $\alpha = 0.05$.

Further analysis steps include classification approaches and prediction of treatment outcome. All further classification, regression, and prediction approaches will be carried out based on the normalized and pre-filtered data in the EMMA2 database.

CHAPTER 9

Discussion

Das Geheimnis zu langweilen besteht darin, alles zu sagen.

Voltaire

In the previous chapters, the design and implementation of the EMMA2 software to support microarray analysis has been described in detail. The development of microarray technology was driven by the need for functional analysis of the high volume of sequence information resulting from several genome sequencing projects including the human genome project. The presence of many genes with unknown function led to the development of functional genomics, including the new – so called 'Omics – approaches trancriptomics, proteomics and metabolomics.

Microarrays can be described as a constanly evolving technology; many different flavours and applications have been developed during the last decade. The measurement of mRNA expression is still the most frequent application of array technology. Gene-expression microarray measurements have been published for bacterial organisms, plants, and animals. The range of possible applications has been extended to DNA hybridization analysis and protein microarrays of very divergent kinds. Microarrays thus have their applications in several 'Omics approaches.

The data sets resulting from microarray analysis are large in terms of data-volume, even when compared to proteomics and metabolomics approaches. In addition, the data exhibit a fundamental influence of variation, both experimental and biological in nature. Both features, data-volume and variability, lead to the development of new approaches, while for some fields of analsis existing methods from statistical inference, data-mining, and machine-learning can be, and were successfully applied in the literature. Despite the rapid development of new analysis techniques, as was often pointed out, the need to carefully evaluate existing methods is rising.

A substantial number of academic and commercial systems, which aim at storage and analysis of microarray data have been developed. Taking this into account, the need to develop another such software for the purpose of application in distributed projects is neglectable. Despite the sheer availability of software packages, other relevant aspects of proper design and implementation of software have to be taken into account.

Commercial closed source applications exhibit remarkable problems for the academic community, as has been shown. It is not only their high financial requirements for obtaining licenses, support, and maintenance. Even more, it is the closed source distribution of the software, which renders the application of commercial products considerably problematic.

The central issue with closed source software is the implementation of algorithms. It is hard or even impossible to figure out exact algorithms and implementation details in closed source software. When it comes to publication of inferences based on statistical analysis or data-mining, it is of primary importance to be able to know the exact algorithm which allows to reproduce the obtained results.

An interesting example of hard to reproduce analysis steps can be found in the analysis algorithm found in the closed-source MAS 5.0 data-analysis from Affymetrix. MAS 5.0 is used for calculation of aggregated expression values per gene from the probe-level signal intensities found on Affymetrix GeneChips. There is also an open-source alternative in the Bioconductor project which attempts to use the same algorithm as described in the the MAS 5.0 documentation. Still, the resulting data might exhibit differences between both implementations.

For the current open source implementations, differences between these implementations and EMMA2 lie in the completeness of the implementation of open standards, and in general, in the compliance with principles of software design, such as use of design patterns and the implementation of a fully documented API. While most of the current open-source microarray software lacks a carefully designed API, all systems lack full support of the MAGE standard, with some providing partial export functionality. None provides complete support for import and export of MAGE-ML files.

A flexible and customizable data analysis mechanism which makes use of graphical programming is also unique to EMMA2. Large distributed projects like the projects which use EMMA2 include laboratories of very different size and amount of bioinformatics staff. Few laboratories in these projects would be able in principle to set up their own analysis software and establish an analysis pipeline for their users. Other laboratories would neither have the resources to effectively establish and use such pipelines. Therefore, it was considered as an essential part of the project to provide centralized services also for data-analysis, in addition to providing storage. That way, it was feasible to provide a rather standardized way of performing at least the basic analysis steps like normalization and inference.

The use of customizable analysis-pipelines within the project showed substantial benefits for a standardized conduct of experimental analysis. The plethora of methods for data-analysis (see Chapter 3) leaves the researcher with the need to choose the optimal set of methods and to decide on good parameters. The need for knowledge, at least on a basic level, about the applicable methods cannot be compensated by a software.

Taking into account the rapid evolution of the array technology observed thus far concerns, whether an analysis software can satisfy all the different emerging technological requirements are justified. A software solution such as EMMA2 has to be almost complete with respect to the data-sources and microarray technologies supported and also flexible as far as the implementation of analysis methods is concerned.

Completeness primarily affects the database-model of the application. If a concept representing a relevant part of a biological experiment cannot be expressed, the model is incomplete. It either needs to be extended, or the lack of functionality has to be accepted. It seems an unreasonably hard task, to build a complete database model, for such a quickly evolving domain as microarray technology. Instead, considerations of completeness imply to use a standardized representation of the domain: the MAGE object model.

With respect to completeness, it seems well justified to use MAGE-OM, though it still cannot guarantee complete coverage of all possible future developments. Instead, it provides a complete implementation of all concepts which can be represented in the only existing standardized language for microarray data: MAGE-ML.

Diverse concerns have been raised about the use of MAGE-OM as a database model, most of them well founded. Primarily, the model is not designed for data-storage but data-transfer; in addition it has some disadvantages such as highly complex class hierarchies, few redundancies and also the lack of efficiency in some parts. On the other hand, only the most trivial of domains can be described in a single optimal model. Instead, any design process involves subjective considerations, which would lead to a suboptimal representation of the real world; the most relevant of such flaws would be to derive a model incompatible with the approved standard.

For the development process of this software, another approach was taken. The MAGE object model was used and automatically transformed in a object-oriented database representation. Code auto-generation was used wherever possible throughout the project. This attempt significantly could reduce manual coding efforts.

Using an extended three-tier architecture, an efficient database system has been implemented, that provides interfaces with high usability. The system has been successfully applied within national and international projects. It is capable of handling high-volume data coming from microarray hybridizations and provides use analysis methods.

Although the efficiency of the design and implementation of EMMA2 has been demonstrated within several large-scale applications, there are still features which are worth being improved. An example can be given regarding the amount of newly developed methods for data analysis. Even using EMMA's PlugIn system, it is impossible to keep up with the recent rapid development of new methods. The availability of source-code of new methods is a major hurdle for rapid integration. A lot of novel methods are published as open-source R packages. Still, it seems almost impossible to obtain all existing algorithms and to evaluate their merits. Also, the introduction of a new function into EMMA still requires a small programming effort, which is due to the different data-structures the algorithms require. But what is most important for providing a standardized toolset and requires a lot of resources: it is essential to understand an algorithm, evaluate it, and develop guidelines on how to set parameters.

The description of a new evaluation framework for statistical tests provides a good impression about the complexity of well grounded and fair method evaluation. By evaluating all test methods it was clearly found that there is no prove for any advantage for any of them. Although it was claimed by all authors of all methods, not all of them seem to provide a significant advantage over a simple t-test. In contrary, bad reproducibility by lack of self-consistency was discovered for the VarMixt test. From the multi-replicate analysis, it can be inferred that the SAM test and the CyberT method perform better than the t-test with very few replicates. However, for reasonable sized experiments, all methods perform comparable. As a result, there can be one simple conclusion: to be very careful about untestified advantages of any new analysis method.

Furthermore the user interface of the software has to be extended in the future owing to the introduction of new array technologies. In some places, the interface needs to be technology dependent; for example the red-green false color images for two-color microarrays do not make sense for single-channel technologies.

With respect to the integration of data from different sources, EMMA2 has made a vast step forward. It is now possible to automatize integrated data-analysis tasks by the use of BRIDGE in combination with web-services. No other application can provide this level of automated data integration. The merits of data-integration could be demonstrated by the use of EMMA2 pipelines on expression-data for which comprehensive genome annotations are available within the GenDB system.

It should be emphasized that the approach of data-integration of different 'Omics data-sources is also feasible without any specialized software. Annotation data can be down-loaded from public resources as well as expression data. The added value of a software framework consisting of several linked applications is the maintenance of consistent references between biological sequence entries and other relevant objects within the framework. But most important, it is the automation of an otherwise tedious task, which makes BRIDGE-based data-integration of EMMA2 so attractive.

For example, the results of an integrated pathway analysis have shown that these methods primarily add to the formulation of novel hypotheses. High expectations to immediately uncover hidden regulatory mechanisms by the application of data-mining algorithms on microarray data alone, are possibly over optimistic.

The same optimism is expressed about the expectations for the emerging field of systems biology. In the future, EMMA2 can function as a concise data repository for model-development and model-evaluation. More care must be taken regarding the annotation of experimental variables to be able to determine distinct gene-expression values for specific outcomes of model simulations of such experimental conditions.

The benefits EMMA2 can provide to systems-biology could still be improved. The difficulty, in this case, is not the lack of a retrieval interface; EMMA2 provides BRIDGE and SAOP interfaces. But still, there is no systems-biology software in operation that makes use of the large amount of transcriptomics data collected in EMMA2 projects.

A possible hinderance can be described as a dilemma: EMMA2 cannot provide one single and unique measurement for each gene and experimental conditions. There can be possibly many experiments studying a single condition or a comparison of conditions; there can be possibly many analysis paths which can lead to a condensed measurement of all replicated datasets. It does not seem feasible to restrict the user to a single experiment per factor with a single normalization and aggregation method; this would be a restriction of exploratory power the users would simply not accept.

One could argue, this question is raised by the complexity of transcriptomics data. As was said already in the introduction, there are infinitely many states for the expressome of an organism. Following this argument, expecting a single condensed value could mean just to have an over-simplified or even naive understanding of the data. Still, it also seems too easy to deny this issue. To provide reduced-complexity views on the data is a legitimate requirement for future software environments, that go a step further into the direction of algorithmic exploration (e.g. simulation) of the cell.

A potential solution for the described dilemma could be to rely on manual curation of the data-sets by project administrators or designated curators. For each completed analysis, a representative condensed data-set can be determined and made availability for cross-experiment comparisons. This is already feasible with EMMA2.

A data-warehouse (IgetDB), currently under development at the CeBiTec, provides a further level of data integration and will support faster queries. Based on the BRIDGE framework, IgetDB consequently follows the data-integration approach. By the use of ontologies, it could provide a condensed and simplified view for all 'Omics. Transcriptomics, proteomics, and metabolomics data will be submitted to it after curation. Further optimized for high-speed queries, IgetDB, BRIDGE and EMMA2 have the potential to become an excellent resource for future modeling systems and even for the new field of constructive biology.

An even better impression of the future of 'Omics and bioinformatics in Bielefeld can be envisaged by having a look at the large set of complex software-tools developed at the CeBiTec (see Figure 9.1). There are established systems for sequence annotation (GenDB and SAMS) and also for proteomics (ProDB). These, rely on invisible tools for developers like O2DBI, GPMS and BRIDGE. More complex tools

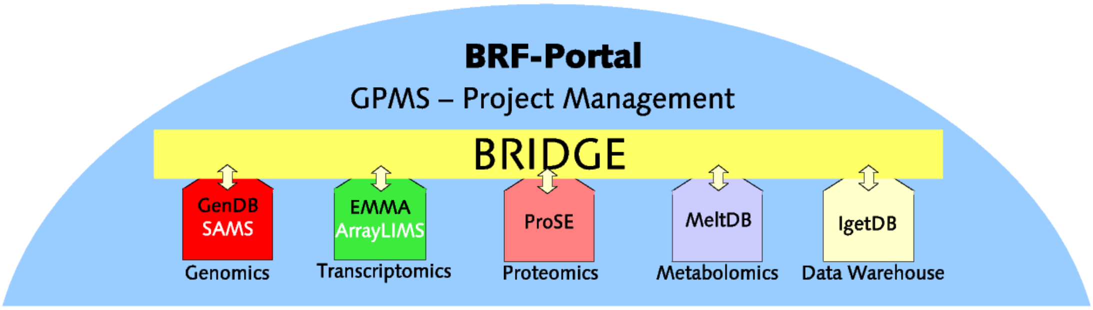


Figure 9.1: The CeBiTec 'Omics software family based on BRIDGE. Unified access to all software is provided by the BRF portal; user management is provided by GPMS.

are developed, complementary to or as an extension to the already existing software: IgetDB, MeltDB for metabolomics, ProSE as the second generation of proteomics software. Among these complex tools, EMMA2 plays a prominent role; it is applied successfully in many projects and integrates seamlessly with other tools and into the CeBiTec infrastructure in general.

But the impact of EMMA2 is not restricted to the CeBiTec. The projects hosted by EMMA2 are mostly collaborative projects with other institutions, and large quantities of arrays have been hybridized by them. Many project partners do not have sufficient resources to maintain secure storage and efficient analysis of microarray data.

In contrast to ordinary repositories that provide only the data, EMMA2 provides a unified system comprising data and methods. Providing a centralized system that provides unification of storage, data analysis, and open standards for communication a unique tool to share both, expertize on data-analysis and data, between the participating institutions. Having large datasets at hands allows to compare the results achieved to those of the other laboratories. Furthermore, methods for data-analysis can be evaluated directly within the analysis environment based on a solid foundation of real-world data. Consequently, new methods can also be made available to all participants immediately.

By providing a LIMS system, exact protocols can be easily disseminated to the involved project partners for standardization. This is extremely important to reduce cross-laboratory variation and assure all experiments remain comparable, thereby enabling the core-facility to give optimized recommendations for handling of the microarrays.

How useful EMMA2 has become as an integrative platform can be best inferred by comparing the existing and actively use projects with a project that is still in its infancy with respect microarrays. Let us take the Marine Genomics Europe (MGE) network as an example which is dedicated to the study of marine organisms and for which EMMA2 provides the central transcriptomics platform. Currently there are only few arrays but the prospected number of experiments within the next year is greater than 1000.

It has been shown within the GenoMik project that the sheer number of hybridizations is not a problem at all, provided a sufficient amount of storage space is provided for the raw data. In contrast to MGE which is a very large European project with a very large number of laboratories, GenoMik is a national project with fewer partners. The Medicago and GLIP projects, on the other hand, are projects which are comparable to MGE in the degree of distribution over different countries. The need for fine-grained access control has been raised within this project, too. It seems highly probable that access control is going to be required in every future European project.

The oceans provide the most diverse eco-systems on the earth. This is reflected in the MGE project, where we find a very large diversity of organisms, ranging from cyanobacteria to higher eukaryotes. While EMMA2 does not depend on a specific class of organisms, there a still things to consider. In the first place, the diversity of organisms is reflected in the number of different array designs which carry sequences of different origin. It has been proven that the quantity of array designs is manageable and does not pose specific requirements. Very different array platforms from diverse manufacturers have been employed.

Data integration has been used successfully in almost all projects using the BRIDGE framework. That was feasible because the databases to link to are located at the CeBiTec. However, for such a diverse project as MGE we cannot rely on this. It is likely that there will be many resources of functional genomics annotations for the involved organisms. This is already the case for some of the existing sequence data. To be able to link to these resources and to be able to integrate such sequence annotations into EMMA's computational pipelines another mechanism is provided by the use of web-services. Remote queries can be used by EMMA2 to retrieve data and other applications may include expression data in the future. This degree of web-service integration make for an unparalleled feature among systems used as centralized repositories such as ArrayExpress of GEO.

The preceeding examples certify that the system is capable of performing all tasks encountered thus far, to do so is true only if the mandatory process of extending the software is taken into account. It does not seem this is going to change for even larger projects in the future. We have to assume that there will be more extensions required than could be envisioned at the moment. Hence, extensibility has been a key feature of EMMA2 already by design. Several extension could already be made, such as a data-mining component, delivering another unique feature of this system compared to other analysis systems, for the GLIP project and several customized analysis PlugIns. By the use of MAGE-OM, EMMA2 offers a vast potential for further extensions for novel array based methods.

In conclusion, EMMA2 provides a large set of methods for microarray data analysis, and what is more, it provides unique features for management, retrieval and exchange of the contained data. With its extensibility and integrative capabilities of other functional genomics resources, a useful combination of instruments has been established, that allows to push functional genomics research forward.

Bibliography

Aach J., Rindone W., Church G.: Systematic management and analysis of yeast gene expression data. *Genome Res*, 10:431–445, (2000).

Alagic S.: *Object oriented database programming*. Texts and monographs in computer science, Springer, New York [u.a.] (1989).

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

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

Attoor S., Dougherty E. R., Chen Y., Bittner M. L., Trent J. M.: Which is better for cDNA-microarray-based classification: ratios or direct intensities. *Bioinformatics*, 20(16):2513–2520, (2004).

Baitaluk M., Qian X., Godbole S., Raval A., Ray A., Gupta A.: PathSys: integrating molecular interaction graphs for systems biology. *BMC Bioinformatics*, 7:55, (2006).

Baker M. D., Wolanin P. M., Stock J. B.: Systems biology of bacterial chemotaxis. *Curr Opin Microbiol*, 9(2):187–192, (2006).

Baldi P., Hatfield G.: *DNA Microarrays and Gene Expression*. Cambridge University Press (2002).

Baldi P., Long A. D.: A Bayesian framework for the analysis of microarray expression data: regularized t -test and statistical inferences of gene changes. *Bioinformatics*, 17(6):509–519, (2001).

Ball C., Awad I., Demeter J., Gollub J., Hebert J., Hernandez-Boussard T., Jin H., Matese J., Nitzberg M., Wymore F., Zachariah Z., Brown P., Sherlock G.: The Stanford Microarray Database accommodates additional microarray platforms and data formats. *Nucleic Acids Res*, 33(Database issue):580–582, (2005).

Ballman K. V., Grill D. E., Oberg A. L., Therneau T. M.: Faster cyclic loess: normalizing RNA arrays via linear models. *Bioinformatics*, 20(16):2778–2786, (2004).

Bammler T., Beyer R. P., Bhattacharya S., Boorman G. A., Boyles A., Bradford B. U., Bumgarner R. E., Bushel P. R., Chaturvedi K., Choi D., Cunningham M. L., Deng S., Dressman H. K., Fannin R. D., Farin F. M., Freedman J. H., Fry R. C., Harper A., Humble M. C., Hurban P., Kavanagh T. J., Kaufmann W. K., Kerr K. F., Jing L., Lapidus J. A., Lasarev M. R., Li J., Li Y.-J., Lobenhofer E. K., Lu X., Malek R. L., Milton S., Nagalla S. R., O'malley J. P., Palmer V. S., Pattee P., Paules R. S., Perou C. M., Phillips K., Qin L.-X., Qiu Y., Quigley S. D., Rodland M., Rusyn I., Samson L. D., Schwartz D. A., Shi Y., Shin J.-L., Sieber S. O., Slifer S., Speer M. C., Spencer P. S., Sproles D. I., Swenberg J. A., Suk W. A., Sullivan R. C., Tian R., Tennant R. W., Todd S. A., Tucker C. J., Houten B. V., Weis B. K., Xuan S., Zarbl H., of the Toxicogenomics Research Consortium M.: Standardizing global gene expression analysis between laboratories and across platforms. *Nat Methods*, 2(5):351–356, (2005).

Ban N., Nissen P., Hansen J., Moore P. B., Steitz T. A.: The complete atomic structure of the large ribosomal subunit at 2.4 A resolution. *Science*, 289(5481):905–920, (2000).

Barrett T., Suzek T. O., Troup D. B., Wilhite S. E., Ngau W.-C., Ledoux P., Rudnev D., Lash A. E., Fujibuchi W., Edgar R.: NCBI GEO: mining millions of expression profiles–database and tools. *Nucleic Acids Res*, 33(Database issue):D562–D566, (2005).

Baumbach J., Brinkrolf K., Czaja L., Rahmann S., Tauch A.: CoryneRegNet: An ontology-based data warehouse of corynebacterial transcription factors and regulatory networks. *BMC Genomics*, 7(1):24, (2006).

Becker A., Bergès H., *et al.*: Global changes gene expression of s. meliloti under microoxic and symbiotic conditions. *Mol Plant Micr*.

Bekel T., Henckel K., Dondrup M., Goesmann A., Küster H.: *The Medicago truncatula handbook*, chap. EST Analysis and Expression Profiling Software from Bielefeld University. Noble Foundation (2006).

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

Bentley S. D., Chater K. F., Cerdeo-Trraga A.-M., Challis G. L., Thomson N. R., James K. D., Harris D. E., Quail M. A., Kieser H., Harper D., Bateman A., Brown S., Chandra G., Chen C. W., Collins M., Cronin A., Fraser A., Goble A., Hidalgo J., Hornsby T., Howarth S., Huang C.-H., Kieser T., Larke L., Murphy L., Oliver K., O'Neil S., Rabbinowitsch E., Rajandream M.-A., Rutherford K., Rutter S., Seeger K., Saunders D., Sharp S., Squares R., Squares S., Taylor K., Warren T., Wietzorrek A., Woodward J., Barrell B. G., Parkhill J., Hopwood D. A.: Complete genome sequence of the model actinomycete Streptomyces coelicolor A3(2). *Nature*, 417(6885):141–147, (2002).

Bertone P., M S.: Advances in functional protein microarray technology. *FEBS J.*, 272(21):5400 – 5411, (2005).

Bertone P., Stolc V., Royce T. E., Rozowsky J. S., Urban A. E., Zhu X., Rinn J. L., Tongprasit W., Samanta M., Weissman S., Gerstein M., Snyder M.: Global identification of human transcribed sequences with genome tiling arrays. *Science*, 306(5705):2242–2246, (2004).

Black M. A., Doerge R. W.: Calculation of the minimum number of replicate spots required for detection of significant gene expression fold change in microarray experiments. *Bioinformatics*, 18(12):1609–1616, (2002).

Blaha M., Premerlani W.: *Object oriented modeling and design for database applications*. Prentice Hall, Upper Saddle River, NJ (1998).

Blake W. J., KAErn M., Cantor C. R., Collins J. J.: Noise in eukaryotic gene expression. *Nature*, 422(6932):633–637, (2003).

Bolstad B. M., Irizarry R. A., Astrand M., Speed T. P.: A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, (2003).

Braasch D. A., Corey D. R.: Locked nucleic acid (LNA): fine-tuning the recognition of DNA and RNA. *Chem Biol*, 8(1):1–7, (2001).

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

Breitling R., Armengaud P., Amtmann A., Herzyk P.: Rank products: a simple, yet powerful, new method to detect differentially regulated genes in replicated microarray experiments. *FEBS Lett*, 573(1-3):83–92, (2004).

Brune I., Becker A., Paarmann D., Albersmeier A., Kalinowski J., Pühler A., Tauch A.: Under the influence of the active deodorant ingredient 4-hydroxy-3-methoxybenzyl alcohol, the skin bacterium Corynebacterium jeikeium moderately responds with differential gene expression. *J Biotechnol*, 127(1):21–33, (2006a).

Brune I., Werner H., Huser A., Kalinowski J., Puhler A., Tauch A.: The DtxR protein acting as dual transcriptional regulator directs a global regulatory network involved in iron metabolism of Corynebacterium glutamicum. *BMC Genomics*, 7(1):21, (2006b).

Butte A. J., Tamayo P., Slonim D., Golub T. R., Kohane I. S.: Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. *Proc Natl Acad Sci U S A*, 97(22):12182–12186, (2000).

Canales R. D., Luo Y., Willey J. C., Austermiller B., Barbacioru C. C., Boysen C., Hunkapiller K., Jensen R. V., Knight C. R., Lee K. Y., Ma Y., Maqsodi B., Papallo A., Peters E. H., Poulter K., Ruppel P. L., Samaha R. R., Shi L., Yang W., Zhang L., Goodsaid F. M.: Evaluation of DNA microarray results with quantitative gene expression platforms. *Nat Biotechnol*, 24(9):1115–1122, (2006).

Chen Y., Dougherty E., Bittner M.: Ratio-based decisions and the quantitative analysis of cDNA microarray images. *Journal of Biomedical Optics*, 2:364–374, (1997).

Churchill G. A.: Fundamentals of experimental design for cDNA microarrays. *Nat Genet*, 32 Suppl:490–495, (2002).

Churchill G. A.: Using ANOVA to analyze microarray data. *Biotechniques*, 37(2):173–5, 177, (2004).

Cleveland W., Devlin S.: Locally weighted regression: An approach to regression analysis by local fitting. *J Am Stat Assoc*, 83:596–610, (1988).

Cole S. T., Brosch R., Parkhill J., Garnier T., Churcher C., Harris D., Gordon S. V., Eiglmeier K., Gas S., Barry C. E., Tekaia F., Badcock K., Basham D., Brown D., Chillingworth T., Connor R., Davies R., Devlin K., Feltwell T., Gentles S., Hamlin N., Holroyd S., Hornsby T., Jagels K., Krogh A., McLean J., Moule S., Murphy L., Oliver K., Osborne J., Quail M. A., Rajandream M. A., Rogers J., Rutter S., Seeger K., Skelton J., Squares R., Squares S., Sulston J. E., Taylor K., Whitehead S., Barrell B. G.: Deciphering the biology of Mycobacterium tuberculosis from the complete genome sequence. *Nature*, 393(6685):537–544, (1998).

Consortium M. A. Q. C., Shi L., Reid L. H., Jones W. D., Shippy R., Warrington J. A., Baker S. C., Collins P. J., de Longueville F., Kawasaki E. S., Lee K. Y., Luo Y., Sun Y. A., Willey J. C., Setterquist R. A., Fischer G. M., Tong W., Dragan Y. P., Dix D. J., Frueh F. W., Goodsaid F. M., Herman D., Jensen R. V., Johnson C. D., Lobenhofer E. K., Puri R. K., Schrf U., Thierry-Mieg J., Wang C., Wilson M., Wolber P. K., Zhang L., Amur S., Bao W., Barbacioru C. C., Lucas A. B., Bertholet V., Boysen C., Bromley B., Brown D., Brunner A., Canales R., Cao X. M., Cebula T. A., Chen J. J., Cheng J., Chu T.-M., Chudin E., Corson J., Corton J. C., Croner L. J., Davies C., Davison T. S., Delenstarr G., Deng X., Dorris D., Eklund A. C., hui Fan X., Fang H., Fulmer-Smentek S., Fuscoe J. C., Gallagher K., Ge W., Guo L., Guo X., Hager J., Haje P. K., Han J., Han T., Harbottle H. C., Harris S. C., Hatchwell E., Hauser C. A., Hester S., Hong H., Hurban P., Jackson S. A., Ji H., Knight C. R., Kuo W. P., LeClerc J. E., Levy S., Li Q.-Z., Liu C., Liu Y., Lombardi M. J., Ma Y., Magnuson S. R., Maqsodi B., McDaniel T., Mei N., Myklebost O., Ning B., Novoradovskaya N., Orr M. S., Osborn T. W., Papallo A., Patterson T. A., Perkins R. G., Peters E. H., Peterson R., Philips K. L., Pine P. S., Pusztai L., Qian F., Ren H., Rosen M., Rosenzweig B. A., Samaha R. R., Schena M., Schroth G. P., Shchegrova S., Smith D. D., Staedtler F., Su Z., Sun H., Szallasi Z., Tezak Z., Thierry-Mieg D., Thompson K. L., Tikhonova I., Turpaz Y., Vallanat B., Van C., Walker S. J., Wang S. J., Wang Y., Wolfinger R., Wong A., Wu J., Xiao C., Xie Q., Xu J., Yang W., Zhang L., Zhong S., Zong Y., Slikker W.: The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements. *Nat Biotechnol*, 24(9):1151–1161, (2006).

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

da Silva A. C. R., Ferro J. A., Reinach F. C., Farah C. S., Furlan L. R., Quaggio R. B., Monteiro-Vitorello C. B., Sluys M. A. V., Almeida N. F., Alves L. M. C., do Amaral A. M., Bertolini M. C., Camargo L. E. A., Camarotte G., Cannavan F., Cardozo J., Chambergo F., Ciapina L. P., Cicarelli R. M. B., Coutinho L. L., Cursino-Santos J. R., El-Dorry H., Faria J. B., Ferreira A. J. S., Ferreira R. C. C., Ferro M. I. T., Formighieri E. F., Franco M. C., Greggio C. C., Gruber A., Katsuyama A. M., Kishi L. T., Leite R. P., Lemos E. G. M., Lemos M. V. F., Locali E. C., Machado M. A., Madeira A. M. B. N., Martinez-Rossi N. M., Martins E. C., Meidanis J., Menck C. F. M., Miyaki C. Y., Moon D. H., Moreira L. M., Novo M. T. M., Okura V. K., Oliveira M. C., Oliveira V. R., Pereira H. A., Rossi A., Sena J. A. D., Silva C., de Souza R. F., Spinola L. A. F., Takita M. A., Tamura R. E., Teixeira E. C., Tezza R. I. D., dos Santos M. T., Truffi D., Tsai S. M., White F. F., Setubal J. C., Kitajima J. P.: Comparison of the genomes of two Xanthomonas pathogens with differing host specificities. *Nature*, 417(6887):459–463, (2002).

Delmar P., Robin S., Daudin J. J.: VarMixt: efficient variance modelling for the differential analysis of replicated gene expression data. *Bioinformatics*, 21(4):502–508, (2005).

Doi N., Takashima H., Kinjo M., Sakata K., Kawahashi Y., Oishi Y., Oyama R., Miyamoto-Sato E., Sawasaki T., Endo Y., Yanagawa H.: Novel fluorescence labeling and high-throughput assay technologies for in vitro analysis of protein interactions. *Genome Res*, 12(3):487–492, (2002).

Downer J., Sevinsky J. R., Ahn N. G., Resing K. A., Betterton M. D.: Incorporating expression data in metabolic modeling: A case study of lactate dehydrogenase. *J Theor Biol*, 240(3):464–474, (2005).

Drew H. R., Wing R. M., Takano T., Broka C., Tanaka S., Itakura K., Dickerson R. E.: Structure of a B-DNA dodecamer: conformation and dynamics. *Proc Natl Acad Sci U S A*, 78(4):2179–2183, (1981).

Dudoit S., Yang Y. H.: *The Analysis of Gene Expression Data*, chap. Bioconductor R packages for exploratory analysis and normalization of cDNA microarray data, pages 73–101. Springer (2003).

Dudoit S., Yang Y. H., Callow M. J., Speed T. P.: Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistia Sinica*, 12:111–139, (2002).

Dysvik B., Jonassen I.: J-Express: exploring gene expression data using Java. *Bioinformatics*, 17(4):369–370, (2001).

Edgar R., Domrachev M., Lash A.: Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res*, 30(1):207–210, (2002).

Eickhoff B., Korn B., Schick M., Poustka A., van der Bosch J.: Normalization of array hybridization experiments in differential gene expression analysis. *Nucleic Acids Res*, 27(22):e33, (1999).

Eisen M., Spellman P., Brown P., Botstein D.: Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A*, 95(25):14863–14868, (1998).

Ermolaeva O., Rastogi M., Pruitt K., Schuler G., Bittner M., Chen Y., Simon R., Meltzer P., Trent J., Boguski M.: Data management and analysis for gene expression arrays. *Nat Genet*, 20(1):19–23, (1998).

Finkelstein D., Ewing R., Gollub J., Sterky F., Cherry J. M., Somerville S.: Microarray data quality analysis: lessons from the AFGC project. Arabidopsis Functional Genomics Consortium. *Plant Mol Biol*, 48(1-2):119–131, (2002).

Firnhaber C., Pühler A., Küster H.: EST sequencing and time course microarray hybridizations identify more than 700 Medicago truncatula genes with developmental expression regulation in flowers and pods. *Planta*, 222(2):269–283, (2005).

Fisher R. A.: The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, (1936).

Florek K., Lukasewicz, Perkal J., Steinhaus H., Zubrzycki S.: Sur la liaison et la division des points d'un ensemble fini. *Colloquium Mathematicum*, 2:282–285, (1951).

Fodor S. P., Rava R. P., Huang X. C., Pease A. C., Holmes C. P., Adams C. L.: Multiplexed biochemical assays with biological chips. *Nature*, 364(6437):555–556, (1993).

Fraunholz M. J.: Systems biology in malaria research. *Trends Parasitol*, 21(9):393–395, (2005).

Friboulet A., Thomas D.: Systems Biology-an interdisciplinary approach. *Biosens Bioelectron*, 20(12):2404–2407, (2005).

Fu J., Jansen R. C.: Optimal design and analysis of genetic studies on gene expression. *Genetics*, 172(3):1993–1999, (2006).

Galibert F., Finan T. M., Long S. R., Puhler A., Abola P., Ampe F., Barloy-Hubler F., Barnett M. J., Becker A., Boistard P., Bothe G., Boutry M., Bowser L., Buhrmester J., Cadieu E., Capela D., Chain P., Cowie A., Davis R. W., Dreano S., Federspiel N. A., Fisher R. F., Gloux S., Godrie T., Goffeau A., Golding B., Gouzy J., Gurjal M., Hernandez-Lucas I., Hong A., Huizar L., Hyman R. W., Jones T., Kahn D., Kahn M. L., Kalman S., Keating D. H., Kiss E., Komp C., Lelaure V., Masuy D., Palm C., Peck M. C., Pohl T. M., Portetelle D., Purnelle B., Ramsperger U., Surzycki R., Thebault P., Vandenbol M., Vorholter F. J., Weidner S., Wells D. H., Wong K., Yeh K. C., Batut J.: The composite genome of the legume symbiont Sinorhizobium meliloti. *Science*, 293(5530):668–672, (2001).

Gamma E.: *Design patterns*. Addison-Wesley professional computing series, Addison-Wesley, Boston, 28. print. Edn. (2004).

Gardiner-Garden M., Littlejohn T. G.: A comparison of microarray databases. *Brief Bioinform*, 2(2):143–158, (2001).

Garwood K., McLaughlin T., Garwood C., Joens S., Morrison N., Taylor C. F., Carroll K., Evans C., Whetton A. D., Hart S., Stead D., Yin Z., Brown A. J. P., Hesketh A., Chater K., Hansson L., Mewissen M., Ghazal P., Howard J., Lilley K. S., Gaskell S. J., Brass A., Hubbard S. J., Oliver S. G., Paton N. W.: PEDRo: a database for storing, searching and disseminating experimental proteomics data. *BMC Genomics*, 5(1):68, (2004).

Genesereth M., Nilsson L.: *On logic in AI*, chap. Logical Foundation of Artificial Intelligence. Morgan Kaufmann (1987).

Gentleman R. C., Carey V. J., Bates D. M., Bolstad B., Dettling M., Dudoit S., Ellis B., Gautier L., Ge Y., Gentry J., Hornik K., Hothorn T., Huber W., Iacus S., Irizarry R., Li F. L. C., Maechler M., Rossini A. J., Sawitzki G., Smith C., Smyth G., Tierney L., Yang J. Y. H., Zhang J.: Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biol*, 5:R80, (2004).

Gentry T. J., Wickham G. S., Schadt C. W., He Z., Zhou J.: Microarray applications in microbial ecology research. *Microb Ecol*, 52(2):159–175, (2006).

Gerth K., Pradella S., Perlova O., Beyer S., Müller R.: Myxobacteria: proficient producers of novel natural products with various biological activities–past and future biotechnological aspects with the focus on the genus Sorangium. *J Biotechnol*, 106(2-3):233–253, (2003).

Gnatt A. L., Cramer P., Fu J., Bushnell D. A., Kornberg R. D.: Structural basis of transcription: an RNA polymerase II elongation complex at 3.3 A resolution. *Science*, 292(5523):1876–1882, (2001).

Goesmann A., Linke B., Bartels D., Dondrup M., Krause L., Neuweger H., Oehm S., Paczian T., Wilke A., Meyer F.: BRIGEP - The BRIDGE-based Genome-Transcriptome-Proteome Browser. *Nucleic Acids Res*, 33:W710–W160, (2005).

Goesmann A., Linke B., Rupp O., Krause L., Bartels D., Dondrup M., McHardy A., Wilke A., Pühler A., Meyer F.: Building a BRIDGE for the integration of heterogeneous data from functional genomics into a platform for systems biology. *J Biotechnol*, 106(2–3):157–167, (2003).

Gollub J., Ball C., Binkley G., Demeter J., Finkelstein D., Hebert J., Hernandez-Boussard T., Jin H., Kaloper M., Matese J., Schroeder M., Brown P., Botstein D., Sherlock G.: The Stanford Microarray Database: data access and quality assessment tools. *Nucleic Acids Res*, 31(1):94–96, (2003).

Golub T. R., Slonim D. K., Tamayo P., Huard C., Gaasenbeek M., Mesirov J. P., Coller H., Loh M. L., Downing J. R., Caligiuri M. A., Bloomfield C. D., Lander E. S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, (1999).

Goryachev A. B., Toh D. J., Lee T.: Systems analysis of a quorum sensing network: design constraints imposed by the functional requirements, network topology and kinetic constants. *Biosystems*, 83(2-3):178–187, (2006).

Grabe N., Neuber K.: A multicellular systems biology model predicts epidermal morphology, kinetics and Ca2+ flow. *Bioinformatics*, 21(17):3541–3547, (2005).

Gruber T.: A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, (1993).

Gutiérrez R. A., Shasha D. E., Coruzzi G. M.: Systems biology for the virtual plant. *Plant Physiol*, 138(2):550–554, (2005).

Hancock D., Wilson M., Velarde G., Morrison N., Hayes A., Hulme H., Wood A. J., Nashar K., Kell D. B., Brass A.: maxdLoad2 and maxdBrowse: standards-compliant tools for microarray experimental annotation, data management and dissemination. *BMC Bioinformatics*, 6:264, (2005).

Handelsman J.: Metagenomics: application of genomics to uncultured microorganisms. *Microbiol Mol Biol Rev*, 68(4):669–685, (2004).

Hayes K. R., Baggs J. E., Hogenesch J. B.: Circadian clocks are seeing the systems biology light. *Genome Biol*, 6(5):219, (2005).

Ho S.-M., Lau K.-M.: DNA microarrays in prostate cancer. *Curr Urol Rep*, 3(1):53–60, (2002).

Hoang H. H., Becker A., Gonzlez J. E.: The LuxR homolog ExpR, in combination with the Sin quorum sensing system, plays a central role in Sinorhizobium meliloti gene expression. *J Bacteriol*, 186(16):5460–5472, (2004).

Hochberg Y.: A sharper bonferroni procedure for multiple tests of significance. *Biometrika*, 75:800–803, (1988).

Hohnjec N., Henckel K., Bekel T., Gouzy J., Michael Dondrup A. G., Küster H.: Transcriptional snapshots provide insights into the molecular basis of arbuscular mycorrhiza in the model legume medicago truncatula. *Functional Plant Biology*, 33(8):737–748, (2006).

Hohnjec N., Vieweg M. F., Pühler A., Becker A., Küster H.: Overlaps in the transcriptional profiles of Medicago truncatula roots inoculated with two different Glomus fungi provide insights into the genetic program activated during arbuscular mycorrhiza. *Plant Physiol*, 137(4):1283–1301, (2005).

Holm S.: A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6:65–70, (1979).

Hommel G.: A stagewise rejective multiple test procedure based on a modified Bonferroni test. *Biometrika*, 75:383–386, (1988).

Huber M., Wei T.-F., Müller U. R., Lefebvre P. A., Marla S. S., Bao Y. P.: Gold nanoparticle probe-based gene expression analysis with unamplified total human RNA. *Nucleic Acids Res*, 32(18):e137, (2004).

Huber W., von Heydebreck A., Sültmann H., Poustka A., Vin gron M.: Variance stabilization applied to microarray data calibration and to the quantification of differential expression. *Bioinformatics*, 18(Suppl. 1):96 – 104, (2002).

Hüser A., Becker A., Brune I., Dondrup M., Kalinowski J., Plassmeier J., Pühler A., Wiegräbe I., Tauch A.: Development of a Corynebacterium glutamicum DNA microarray and validation by genome-wide expression profiling during growth with propionate as carbon source. *J Biotechnol*, 106(2–3):135–146, (2003).

Hüser A. T., Chassagnole C., Lindley N. D., Merkamm M., Guyonvarch A., Elisáková V., Pátek M., Kalinowski J., Brune I., Pühler A., Tauch A.: Rational design of a Corynebacterium glutamicum pantothenate production strain and its characterization by metabolic flux analysis and genome-wide transcriptional profiling. *Appl Environ Microbiol*, 71(6):3255–3268, (2005).

Ikeo K., Ishi-i J., Tamura T., Gojobori T., Tateno Y.: CIBEX: center for information biology gene expression database. *C R Biologies*, 326(10-11):1079–1082, (2003).

Ioannidis J. P. A.: Microarrays and molecular research: noise discovery? *Lancet*, 365(9458):454–455, (2005).

Jones A., Hunt E., Wastling J. M., Pizarro A., Stoeckert C. J.: An object model and database for functional genomics. *Bioinformatics*, 20(10):1583–1590, (2004).

Journet E.-P., van Tuinen D., Gouzy J., Crespeau H., Carreau V., Farmer M.-J., Niebel A., Schiex T., Jaillon O., Chatagnier O., Godiard L., Micheli F., Kahn D., Gianinazzi-Pearson V., Gamas P.: Exploring root symbiotic programs in the model legume Medicago truncatula using EST analysis. *Nucleic Acids Res*, 30(24):5579–5592, (2002).

Kaern M., Elston T. C., Blake W. J., Collins J. J.: Stochasticity in gene expression: from theories to phenotypes. *Nat Rev Genet*, 6(6):451–464, (2005).

Kalinowski J., Bathe B., Bartels D., Bischoff N., Bott M., Burkovski A., Dusch N., Eggeling L., Eikmanns B. J., Gaigalat L., Goesmann A., Hartmann M., Huthmacher K., Krämer R., Linke B., McHardy A. C., Meyer F., Möckel B., Pfefferle W., Pühler A., Rey D. A., Rückert C., Rupp O., Sahm H., Wendisch V. F., Wiegräbe I., Tauch A.: The complete Corynebacterium glutamicum ATCC 13032 genome sequence and its impact on the production of L-aspartate-derived amino acids and vitamins. *J Biotechnol*, 104(1-3):5–25, (2003).

Kaufman R., Rousseeuw P. J.: *Finding Groups in Data. An Introduction to Cluster Analysis*. Wiley Series in Probability and Mathematical Statistics, Wiley (1990).

Kerr M. K.: Design considerations for efficient and effective microarray studies. *Biometrics*, 59(4):822–828, (2003).

Kerr M. K., Churchill G. A.: Experimental design for gene expression microarrays. *Biostatistics*, 2(2):183–201, (2001).

Kidgell C., Winzeler E.: Elucidating genetic diversity with oligonucleotide arrays. *Chromosome Research*, 13(3):225 – 235, (2005).

Killion P. J., Sherlock G., Iyer V. R.: The Longhorn Array Database (LAD): an open-source, MIAME compliant implementation of the Stanford Microarray Database (SMD). *BMC Bioinformatics*, 4:32, (2003).

Kim Y., Geiger J. H., Hahn S., Sigler P. B.: Crystal structure of a yeast TBP/TATA-box complex. *Nature*, 365(6446):512–520, (1993).

Kitano H.: Computational systems biology. *Nature*, 420(6912):206–210, (2002a).

Kitano H.: Systems biology: a brief overview. *Science*, 295(5560):1662–1664, (2002b).

Koch D. J., Rückert C., Albersmeier A., Hüser A. T., Tauch A., Pühler A., Kalinowski J.: The transcriptional regulator SsuR activates expression of the Corynebacterium glutamicum sulphonate utilization genes in the absence of sulphate. *Mol Microbiol*, 58(2):480–494, (2005).

Kohane I. S., Kho A. T., Butte A.: *Microarrays for an Integrative Genomics*. MIT Press, Cambridge (2003).

Kohonen T.: *Self-organizing maps*. Springer (1995).

Krol E., Becker A.: Global transcriptional analysis of the phosphate starvation response in Sinorhizobium meliloti strains 1021 and 2011. *Mol Genet Genomics*, 272(1):1–17, (2004).

Kroll T. C., Wölfl S.: Ranking: a closer look on globalisation methods for normalisation of gene expression arrays. *Nucleic Acids Res*, 30(11):e50, (2002).

Küster H., Becker A., Firnhaber C., Hohnjec N., Manthey K., Perlick A. M., Bekel T., Dondrup M., Henckel K., Goesmann A., Meyer F., Wipf D., Requena N., Hildebrandt U., Hampp R., Nehls U., Krajinski F., Franken P., Pühler A.: Development of bioinformatic tools to support EST-sequencing, in silico- and microarray-based transcriptome profiling in mycorrhizal symbioses. *Phytochemistry*, 68(1):19–32, (2006).

Küster H., Dondrup M.: Integration of gene expression profiling in European projects on legumes. *Grain Legumes*, 46:14–15, (2006).

Küster H., Hohnjec N., Krajinski F., El Y. F., Manthey K., Gouzy J., Dondrup M., Meyer F., Kalinowski J., Brechenmacher L., van Tuinen D., Gianinazzi-Pearson V., Pühler A., Gamas P., Becker A.: Construction and validation of cDNA-based Mt6k-RIT macro- and microarrays to explore root endosymbioses in the model legume Medicago truncatula. *J Biotechnol*, 108(2):95–113, (2004).

Laub M. T., McAdams H. H., Feldblyum T., Fraser C. M., Shapiro L.: Global analysis of the genetic network controlling a bacterial cell cycle. *Science*, 290(5499):2144–2148, (2000).

Lee I. M., Bartoszyk I. M., Gundersen-Rindal D. E., Davis R. E.: Phylogeny and classification of bacteria in the genera Clavibacter and Rathayibacter on the basis of 16s rRNA gene sequence analyses. *Appl Environ Microbiol*, 63(7):2631–2636, (1997).

Lee P. D., Sladek R., Greenwood C. M. T., Hudson T. J.: Control genes and variability: absence of ubiquitous reference transcripts in diverse mammalian expression studies. *Genome Res*, 12(2):292–297, (2002).

Lewin B.: *Genes VIII*. Pearson Prentice Hall, Upper Saddle River, NJ (2004).

Li C., Wong W. H.: Model-based analysis of oligonucleotide arrays: expression index computation and outlier detection. *Proc Natl Acad Sci U S A*, 98(1):31–36, (2001).

Li D., Li J., Ouyang S., Wang J., Wu S., Wan P., Zhu Y., Xu X., He F.: Protein interaction networks of Saccharomyces cerevisiae, Caenorhabditis elegans and Drosophila melanogaster: large-scale organization and robustness. *Proteomics*, 6(2):456–461, (2006).

Li S. S., Bigler J., Lampe J. W., Potter J. D., Feng Z.: FDR-controlling testing procedures and sample size determination for microarrays. *Stat Med*, 24(15):2267–2280, (2005).

Liao J. G., Lin Y., Selvanayagam Z. E., Shih W. J.: A mixture model for estimating the local false discovery rate in DNA microarray analysis. *Bioinformatics*, 20(16):2694–2701, (2004).

Lipshutz R. J., Fodor S. P., Gingeras T. R., Lockhart D. J.: High density synthetic oligonucleotide arrays. *Nat Genet*, 21(1 Suppl):20–24, (1999).

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

Liu J. J., Cutler G., Li W., Pan Z., Peng S., Hoey T., Chen L., Ling X. B.: Multiclass cancer classification and biomarker discovery using GA-based algorithms. *Bioinformatics*, 21(11):2691–2697, (2005).

Liu J.-P., Guerasimova A., Schwartz R., Lange M., Lehrach H., Nyrsik L., Janitz M.: LNA-modified oligodeoxynucleotide hybridization with DNA microarrays printed on nanoporous membrane slides. *Comb Chem High Throughput Screen*, 9(8):591–597, (2006).

Long W.-H., Xiao H.-S., Gu X.-M., Zhang Q.-H., Yang H.-J., Zhao G.-P., Liu J.-H.: A universal microarray for detection of SARS coronavirus. *J Virol Methods*, 121(1):57–63, (2004).

Lu T., Costello C. M., Croucher P. J. P., Häsler R., Deuschl G., Schreiber S.: Can Zipf's law be adapted to normalize microarrays? *BMC Bioinformatics*, 6:37, (2005).

Luhmann N.: *Soziale Systeme. Grundriss einer allgemeinen Theorie*. Suhrkamp, Frankfurt (1994).

MacBeath G., Schreiber S. L.: Printing proteins as microarrays for high-throughput function determination. *Science*, 289(5485):1760–1763, (2000).

MacQueen J.: Some methods for classification and analysis of multivariate observations. In: L. M. Le Cam, J. Neyman, eds., *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley: University of California Press (1967).

Mainguy J., Macdonnell G., Bund S., Wild D.: Kmd : An open-source port of the arrayexpress microarray database. *ApplIED Bioinformatics*, 3(4):257–260, (2004).

Manthey K., Krajinski F., Hohnjec N., Firnhaber C., Pühler A., Perlick A. M., Küster H.: Transcriptome profiling in root nodules and arbuscular mycorrhiza identifies a collection of novel genes induced during Medicago truncatula root endosymbioses. *Mol Plant Microbe Interact*, 17(10):1063–1077, (2004).

Martinetz T. M., Berkovich S., Schulten K. J.: Neural-gas network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4(4):558–569, (1993).

Masson L., Maynard C., Brousseau R., SH G., Hemmingsen S., Hill J., Paccagnella A., Oda R., Kimura N.: Identification of pathogenic Helicobacter species by chaperonin-60 differentiation on plastic DNA arrays. *Genomics*, 87(1):104 – 112, (2006).

Mattoon D., Michaud G., Merkel J., Schweitzer B.: Biomarker discovery using protein microarray technology platforms: antibody-antigen complex profiling. *Expert Rev Proteomics*, 2(6):879–889, (2005).

Matys V., Kel-Margoulis O. V., Fricke E., Liebich I., Land S., Barre-Dirrie A., Reuter I., Chekmenev D., Krull M., Hornischer K., Voss N., Stegmaier P., Lewicki-Potapov B., Saxel H., Kel A. E., Wingender E.: TRANSFAC and its module TRANSCompel: transcriptional gene regulation in eukaryotes. *Nucleic Acids Res*, 34(Database issue):D108–D110, (2006).

Maurer M., Molidor R., Sturn A., Hartler J., Hackl H., Stocker G., Prokesch A., Scheideler M., Trajanoski Z.: MARS: microarray analysis, retrieval, and storage system. *BMC Bioinformatics*, 6(1):101, (2005).

McIndoe R., Lanzen A., Hurtz K.: MADGE: scalable distributed data management software for cDNA microarrays. *Bioinformatics*, 19(1):87–89, (2003).

McQuitty L.: Elementary linkage analysis for isolating orthogonal and oblique types and typal relevancies. *Educational and Psychological Measurement*, 17:207–229, (1957).

Méndez M. A., Hödar C., Vulpe C., González M., Cambiazo V.: Discriminant analysis to evaluate clustering of gene expression data. *FEBS Lett*, 522(1-3):24–28, (2002).

Meyer F., Goesmann A., McHardy A. C., Bartels D., Bekel T., Clausen J., Kalinowski J., Linke B., Rupp O., Giegerich R., Phler A.: GenDB–an open source genome annotation system for prokaryote genomes. *Nucleic Acids Res*, 31(8):2187–2195, (2003).

Michiels S., Koscielny S., Hill C.: Prediction of cancer outcome with microarrays: a multiple random validation strategy. *Lancet*, 365(9458):488–492, (2005).

Miron M., Nadon R.: Inferential literacy for experimental high-throughput biology. *Trends Genet*, 22(2):84–89, (2006).

Murtagh F.: *COMPSTAT Lectures 4*, chap. Multidimensional Clustering Algorithms. Physica-Verlag, Würzburg (1985).

Ntzani E. E., Ioannidis J. P. A.: Predictive ability of DNA microarrays for cancer outcomes and correlates: an empirical assessment. *Lancet*, 362(9394):1439–1444, (2003).

O'Malley M. A., Dupré J.: Fundamental issues in systems biology. *Bioessays*, 27(12):1270–1276, (2005).

Page G. P., Edwards J. W., Gadbury G. L., Yelisetti P., Wang J., Trivedi P., Allison D. B.: The PowerAtlas: a power and sample size atlas for microarray experimental design and research. *BMC Bioinformatics*, 7:84, (2006).

Pan W., Lin J., Le C. T.: How many replicates of arrays are required to detect gene expression changes in microarray experiments? A mixture model approach. *Genome Biol*, 3(5):research0022, (2002).

Park P. J., Cao Y. A., Lee S. Y., Kim J.-W., Chang M. S., Hart R., Choi S.: Current issues for DNA microarrays: platform comparison, double linear amplification, and universal RNA reference. *J Biotechnol*, 112(3):225–245, (2004).

Park T., Yi S.-G., Kang S.-H., Lee S., Lee Y.-S., Simon R.: Evaluation of normalization methods for microarray data. *BMC Bioinformatics*, 4:33, (2003).

Parkinson H., Sarkans U., Shojatalab M., Abeygunawardena N., Contrino S., Coulson R., Farne A., Lara G. G., Holloway E., Kapushesky M., Lilja P., Mukherjee G., Oezcimen A., Rayner T., Rocca-Serra P., Sharma A., Sansone S., Brazma A.: ArrayExpress–a public repository for microarray gene expression data at the EBI. *Nucleic Acids Res*, 33(Database issue):D553–D555, (2005).

Parmigiani G., Garrett E. S., Irizarry R. A., Zeger S. L., eds.: *The Analysis of Gene Expression Data*. Springer (2003).

Patterson T. A., Lobenhofer E. K., Fulmer-Smentek S. B., Collins P. J., Chu T.-M., Bao W., Fang H., Kawasaki E. S., Hager J., Tikhonova I. R., Walker S. J., Zhang L., Hurban P., de Longueville F., Fuscoe J. C., Tong W., Shi L., Wolfinger R. D.: Performance comparison of one-color and two-color platforms within the MicroArray Quality Control (MAQC) project. *Nat Biotechnol*, 24(9):1140–1150, (2006).

Pavlidis P., Weston J., Cai J., Noble W. S.: Learning gene functional classifications from multiple data types. *J Comput Biol*, 9(2):401–411, (2002).

Pedroso S., Guillen I. A.: Microarray and nanotechnology applications of functional nanoparticles. *Comb Chem High Throughput Screen*, 9(5):389–397, (2006).

Puskás L. G., Nagy Z. B., Kelemen J. Z., Rüberg S., Bodogai M., Becker A., Dusha I.: Wide-range transcriptional modulating effect of ntrR under microaerobiosis in Sinorhizobium meliloti. *Mol Genet Genomics*, 272(3):275–289, (2004).

Qin L., Beyer R., Hudson F., Linford N., Morris D., Kerr K.: Evaluation of methods for oligonucleotide array data via quantitative real-time pcr. *BMC Bioinformatics*, 7:23, (2006).

Qin L.-X., Kerr K. F., of the Toxicogenomics Research Consortium C. M.: Empirical evaluation of data transformations and ranking statistics for microarray analysis. *Nucleic Acids Res*, 32(18):5471–5479, (2004).

Quackenbush J.: Microarray data normalization and transformation. *Nat Genet*, 32 Suppl:496–501, (2002).

Quackenbush J.: Genomics. Microarrays–guilt by association. *Science*, 302(5643):240–241, (2003).

R Development Core Team: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria (2005), ISBN 3-900051-07-0.

Rahmann S., Müller T., Vingron M.: On the power of profiles for transcription factor binding site detection. *Stat Appl Genet Mol Biol*, 2(1):Article7, (2003).

Ramaswamy S., Golub T. R.: DNA microarrays in clinical oncology. *J Clin Oncol*, 20(7):1932–1941, (2002).

Rao C. V., Wolf D. M., Arkin A. P.: Control, exploitation and tolerance of intracellular noise. *Nature*, 420(6912):231–237, (2002).

Rees C. A., Demeter J., Matese J. C., Botstein D., Sherlock G.: GeneXplorer: an interactive web application for microarray data visualization and analysis. *BMC Bioinformatics*, 5:141, (2004).

Reiner A., Yekutieli D., Benjamini Y.: Identifying differentially expressed genes using false discovery rate controlling procedures. *Bioinformatics*, 19(3):368–375, (2003).

Repsilber D., Ziegler A.: Two-color microarray experiments. Technology and sources of variance. *Methods Inf Med*, 44(3):400–404, (2005).

Rey D. A., Nentwich S. S., Koch D. J., Rückert C., Pühler A., Tauch A., Kalinowski J.: The McbR repressor modulated by the effector substance S-adenosylhomocysteine controls directly the transcription of a regulon involved in sulphur metabolism of Corynebacterium glutamicum ATCC 13032. *Mol Microbiol*, 56(4):871–887, (2005).

Rey D. A., Pühler A., Kalinowski J.: The putative transcriptional repressor McbR, member of the TetR-family, is involved in the regulation of the metabolic network directing the synthesis of sulfur containing amino acids in Corynebacterium glutamicum. *J Biotechnol*, 103(1):51–65, (2003).

Ripley B. D.: *Pattern recognition and Neral Networks*. Cambridge University Press, Cambridge (1996).

Rüberg S., Tian Z.-X., Krol E., Linke B., Meyer F., Wang Y., Pühler A., Weidner S., Becker A.: Construction and validation of a Sinorhizobium meliloti whole genome DNA microarray: genome-wide profiling of osmoadaptive gene expression. *J Biotechnol*, 106(2-3):255–268, (2003).

Rückert C., Koch D. J., Rey D. A., Albersmeier A., Mormann S., Pühler A., Kalinowski J.: Functional genomics and expression analysis of the Corynebacterium glutamicum fpr2-cysIXHDNYZ gene cluster involved in assimilatory sulphate reduction. *BMC Genomics*, 6:121, (2005).

Rumbaugh J., Booch G., Jacobsen I.: *The UML Reference Guide*. Addison Wesley, Reading (1999).

Saal L. H., Troein C., Vallon-Christersson J., Gruvberger S., Borg A., Peterson C.: BioArray Software Environment (BASE): a platform for comprehensive management and analysis of microarray data. *Genome Biol*, 3(8):SOFTWARE0003, (2002).

Saeed A., Sharov V., White J., Li J., Liang W., Bhagabati N., Braisted J., Klapa M., Currier T., Thiagarajan M., Sturn A., Snuffin M., Rezantsev A., Popov D., Ryltsov A., Kostukovich E., Borisovsky I., Liu Z., Vinsavich A., Trush V., Quackenbush J.: TM4: A free, open-source system for microarray data management and analysis. *Bio Techniques*, 34:374–378, (2003).

Sásik R., Calvo E., Corbeil J.: Statistical analysis of high-density oligonucleotide arrays: a multiplicative noise model. *Bioinformatics*, 18(12):1633–1640, (2002).

Schena M.: *Microarray Analysis*. Wiley (2003).

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

Schulze-Kremer S.: Adding semantics to genome databases: towards an ontology for molecular biology. *Proc Int Conf Intell Syst Mol Biol*, 5:272–275, (1997).

Sebat J. L., Colwell F. S., Crawford R. L.: Metagenomic profiling: microarray analysis of an environmental genomic library. *Appl Environ Microbiol*, 69(8):4927–4934, (2003).

Sharan R., Maron-Katz A., Shamir R.: CLICK and EXPANDER: a system for clustering and visualizing gene expression data. *Bioinformatics*, 19(14):1787–1799, (2003).

Shaw M., Garlan D.: *Software architecture*. An Alan R. Apt book, Prentice Hall, Upper Saddle River, NJ (1996).

Sherlock G., Hemandez-Boussard T., Kasarskis A., Binkley G., Matese J., Dwight S., Kaloper M., Weng S., Jin H., Ball C., Eisen M., Spellman P., Brown P., Botstein D., Cherry J.: The stanford microarray database. *Nucleic Acids Res*, 29(1):152–155, (2001).

Shields R.: MIAME, we have a problem. *Trends Genet*, 22(2):65–66, (2006).

Siegel S.: *Nonparametric statistics for the behavioral sciences*. McGraw-Hill, New York (1956).

Silberbach M., Burkovski A.: Application of global analysis techniques to Corynebacterium glutamicum: New insights into nitrogen regulation. *J Biotechnol*.

Silberbach M., Hüser A., Kalinowski J., Pühler A., Walter B., Krämer R., Burkovski A.: DNA microarray analysis of the nitrogen starvation response of Corynebacterium glutamicum. *J Biotechnol*, 119(4):357–367, (2005a).

Silberbach M., Schäfer M., Hüser A. T., Kalinowski J., Pühler A., Krämer R., Burkovski A.: Adaptation of Corynebacterium glutamicum to ammonium limitation: a global analysis using transcriptome and proteome techniques. *Appl Environ Microbiol*, 71(5):2391–2402, (2005b).

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

Smyth G. K.: *Bioinformatics and Computational Biology Solutions using R and Bioconductor*, chap. Limma: linear models for microarray data, pages 397–420. Springer, New York (2005).

Smyth G. K., Speed T.: Normalization of cDNA microarray data. *Methods*, 31(4):265–273, (2003).

Sneath P. H.: The application of computers to taxonomy. *J Gen Microbiol*, 17(1):201–226, (1957).

Sokal R. R., Michener C. D.: *Univ. Kansas Sci. Bull.*, vol. 38, chap. A statistical method for evaluating systematic relationships, pages 1409–1438. Lawrence, K. : University of Kansas (1958).

Sonnhammer E. L., Eddy S. R., Birney E., Bateman A., Durbin R.: Pfam: multiple sequence alignments and HMM-profiles of protein domains. *Nucleic Acids Res*, 26(1):320–322, (1998).

Spellman P., Sherlock G., Zhang M., Iyer V., Anders K., Eisen M., Brown P., Botstein D., Futcher B.: Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9:3273–3297, (1998).

Stoeckert C. J., Causton H. C., Ball C. A.: Microarray databases: standards and ontologies. *Nat Genet*, 32 Suppl:469–473, (2002).

Stuart J. M., Segal E., Koller D., Kim S. K.: A gene-coexpression network for global discovery of conserved genetic modules. *Science*, 302(5643):249–255, (2003).

Tamayo P., Slonim D., Mesirov J., Zhu Q., Kitareewan S., Dmitrovsky E., Lander E. S., Golub T. R.: Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc Natl Acad Sci U S A*, 96(6):2907–2912, (1999).

Tatusov R. L., Fedorova N. D., Jackson J. D., Jacobs A. R., Kiryutin B., Koonin E. V., Krylov D. M., Mazumder R., Mekhedov S. L., Nikolskaya A. N., Rao B. S., Smirnov S., Sverdlov A. V., Vasudevan S., Wolf Y. I., Yin J. J., Natale D. A.: The COG database: an updated version includes eukaryotes. *BMC Bioinformatics*, 4:41, (2003).

Tatusov R. L., Galperin M. Y., Natale D. A., Koonin E. V.: The COG database: a tool for genome-scale analysis of protein functions and evolution. *Nucleic Acids Res*, 28(1):33–36, (2000).

Tauch A., Kaiser O., Hain T., Goesmann A., Weisshaar B., Albersmeier A., Bekel T., Bischoff N., Brune I., Chakraborty T., Kalinowski J., Meyer F., Rupp O., Schneiker S., Viehoever P., Pühler A.: Complete genome sequence and analysis of the multiresistant nosocomial pathogen Corynebacterium jeikeium K411, a lipid-requiring bacterium of the human skin flora. *J Bacteriol*, 187(13):4671–4682, (2005).

Taylor C. F.: Minimum Reporting Requirements for Proteomics: A MIAPE Primer. *Proteomics*, 6 Suppl 2:39–44, (2006).

Tettelin H., Saunders N. J., Heidelberg J., Jeffries A. C., Nelson K. E., Eisen J. A., Ketchum K. A., Hood D. W., Peden J. F., Dodson R. J., Nelson W. C., Gwinn M. L., DeBoy R., Peterson J. D., Hickey E. K., Haft D. H., Salzberg S. L., White O., Fleischmann R. D., Dougherty B. A., Mason T., Ciecko A., Parksey D. S., Blair E., Cittone H., Clark E. B., Cotton M. D., Utterback T. R., Khouri H., Qin H., Vamathevan J., Gill J., Scarlato V., Masignani V., Pizza M., Grandi G., Sun L., Smith H. O., Fraser C. M., Moxon E. R., Rappuoli R., Venter J. C.: Complete genome sequence of Neisseria meningitidis serogroup B strain MC58. *Science*, 287(5459):1809–1815, (2000).

Thieme F., Koebnik R., Bekel T., Berger C., Boch J., Büttner D., Caldana C., Gaigalat L., Goesmann A., Kay S., Kirchner O., Lanz C., Linke B., McHardy A. C., Meyer F., Mittenhuber G., Nies D. H., Niesbach-Klösgen U., Patschkowski T., Rückert C., Rupp O., Schneiker S., Schuster S. C., Vorhölter F.-J., Weber E., Pühler A., Bonas U., Bartels D., Kaiser O.: Insights into genome plasticity and pathogenicity of the plant pathogenic bacterium Xanthomonas campestris pv. vesicatoria revealed by the complete genome sequence. *J Bacteriol*, 187(21):7254–7266, (2005).

Thompson R., Ratet P., Küster H.: Identification of gene functions by applying tilling and insertional mutagenesis strategies on microarray-based expression data. *Grain Legumes*, 41:20–22, (2005).

Törönen P., Kolehmainen M., Wong G., Castrén E.: Analysis of gene expression data using self-organizing maps. *FEBS Lett*, 451(2):142–146, (1999).

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

van de Vijver M.: Gene-expression profiling and the future of adjuvant therapy. *Oncologist*, 10 Suppl 2:30–34, (2005).

van de Vijver M. J., He Y. D., van't Veer L. J., Dai H., Hart A. A. M., Voskuil D. W., Schreiber G. J., Peterse J. L., Roberts C., Marton M. J., Parrish M., Atsma D., Witteveen A., Glas A., Delahaye L., van der Velde T., Bartelink H., Rodenhuis S., Rutgers E. T., Friend S. H., Bernards R.: A gene-expression signature as a predictor of survival in breast cancer. *N Engl J Med*, 347(25):1999–2009, (2002).

van 't Veer L. J., Dai H., van de Vijver M. J., He Y. D., Hart A. A. M., Mao M., Peterse H. L., van der Kooy K., Marton M. J., Witteveen A. T., Schreiber G. J., Kerkhoven R. M., Roberts C., Linsley P. S., Bernards R., Friend S. H.: Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, (2002).

Vapnik V. N.: *Statistical Learning Theory*. Wiley, New York (1998).

Velculescu V. E., Zhang L., Vogelstein B., Kinzler K. W.: Serial analysis of gene expression. *Science*, 270(5235):484–487, (1995).

von Bertalanffy L.: *General System Theory: Foundations, Development, Applications*. George Braziller, New York (1968).

Wall L., Christiansen T., Orwant J.: *Programming Perl*. O'Reilly, third edition Edn. (2000).

Wang Y., Klijn J. G. M., Zhang Y., Sieuwerts A. M., Look M. P., Yang F., Talantov D., Timmermans M., van Gelder M. E. M., Yu J., Jatkoe T., Berns E. M. J. J., Atkins D., Foekens J. A.: Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365(9460):671–679, (2005).

Ward J. H.: Hierarchical grouping to optimize an objective function. *J Am Stat Assoc*, 58:236–244, (1963).

Wendisch V. F., Bott M., Kalinowski J., Oldiges M., Wiechert W.: Emerging Corynebacterium glutamicum systems biology. *J Biotechnol*, 124(1):74–92, (2006).

Wilkinson M., Gessler D., Farmer A., Stein L.: The BioMOBY project explores open-source, simple, extensible protocols for enabling biological database interoperability. In: *Proceedings of the Virtual Conference on Genomics and Bioinformatics*, pages 17–27 (2004).

Wilkinson M. D., Links M.: BioMOBY: an open source biological web services proposal. *Brief Bioinform*, 3(4):331–341, (2002).

Williamson M. P.: Systems biology: will it work? *Biochem Soc Trans*, 33(Pt 3):503–506, (2005).

Wong K., Sayo P.: *Free/Open Source Software*. UNDP Asia-Pacific Development Information Programme (2004).

Wong M. L., Medrano J. F.: Real-time PCR for mRNA quantitation. *Biotechniques*, 39(1):75–85, (2005).

Wu W., Xing E. P., Myers C., Mian I. S., Bissell M. J.: Evaluation of normalization methods for cDNA microarray data by k-NN classification. *BMC Bioinformatics*, 6:191, (2005).

Yahyaoui F. E., Küster H., Amor B. B., Hohnjec N., Pühler A., Becker A., Gouzy J., Vernié T., Gough C., Niebel A., Godiard L., Gamas P.: Expression profiling in Medicago truncatula identifies more than 750 genes differentially expressed during nodulation, including many potential regulators of the symbiotic program. *Plant Physiol*, 136(2):3159–3176, (2004).

Yang Y., Dudoit S., Luu P., Lin D., Peng V., Ngai J., Speed T.: Normalization for cDNA microarray data: a robust composite method adressing single and multiple slide systematic variation. *Nucleic Acids Res*, 30(4):?, (2002).

Yang Y. H., Buckley M. J., Speed T. P.: Analysis of cDNA microarray images. *Brief Bioinform*, 2(4):341–349, (2001).

Yang Y. H., Speed T.: Design issues for cDNA microarray experiments. *Nat Rev Genet*, 3(8):579–588, (2002).

Yao S.-Y., Luo L., Har K. J., Becker A., Rüberg S., Yu G.-Q., Zhu J.-B., Cheng H.-P.: Sinorhizobium meliloti ExoR and ExoS proteins regulate both succinoglycan and flagellum production. *J Bacteriol*, 186(18):6042–6049, (2004).

Yauk C., Berndt L., Williams A., Douglas G. R.: Automation of cDNA microarray hybridization and washing yields improved data quality. *J Biochem Biophys Methods*, 64(1):69–75, (2005).

Yauk C. L., Berndt M. L., Williams A., Douglas G. R.: Comprehensive comparison of six microarray technologies. *Nucleic Acids Res*, 32(15):e124, (2004).

Yeung K. Y., Fraley C., Murua A., Raftery A. E., Ruzzo W. L.: Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17(10):977–987, (2001).

Yin W., Chen T., Zhou S. X., Chakraborty A.: Background correction for cDNA microarray images using the TV+L1 model. *Bioinformatics*, 21(10):2410–2416, (2005).

Zhu H., Bilgin M., Bangham R., Hall D., Casamayor A., Bertone P., Lan N., Jansen R., Bidlingmaier S., Houfek T., Mitchell T., Miller P., Dean R. A., Gerstein M., Snyder M.: Global analysis of protein activities using proteome chips. *Science*, 293(5537):2101–2105, (2001).

Acknowledgements

During the course of my thesis I have met many people without whom I would never have reached this point. I would like to express my gratitude to all those people who have helped me on the way.

First of all, I wish to thank Prof. Dr. Robert Giegerich and Prof. Dr. Alfred Pühler for the opportunity to do my PhD. thesis under their supervision. I am very grateful for their support, fruitful and critical discussions, inspiration and their trust that this project would finally come to a good end.

Also, I am very grateful to Dr. Alexander Goesmann for the opportunity to develop EMMA2 within the BRF and for his overall support in pushing this project forward. I also thank all the people of the BRF, for proof-reading the manuscripts, providing simply good mood and positive atmosphere.

I also feel indebted to Dr. Helge Küster for his patience in being a beta-tester for EMMA, reading the manuscripts, and being a reliable source of help and support for all kinds of problems one might encounter in the course a PhD thesis. Furthermore, I would like to thank Dr. Anke Becker, Dr. Jörn Kalinowski, and Dr. Andrea Hüser for fruitful discussions and explanations and for their input. In general, I feel grateful to all the people who have helped by reading the manuscripts and testing the software.

I would like to acknowledge the International Graduate School in Bioinformatics and Genome Research, Dr. Dirk Evers, the executive director, and all PhD students of the graduate school for their support. I have enjoyed the nice atmosphere very much, not to mention the numerous great events at Haus Reineberg.

Finally, I want to express my gratitude to all my friends who have been always there when needed. Most of all I wish to thank my mother and all of family for their love, trust, and support.

Bielefeld, January 2007

Michael Dondrup