DISSERTATIONES
INFORMATICAE
UNIVERSITATIS
TARTUENSIS
**27**

LIIS KOLBERG

# Developing and applying bioinformatics tools for gene expression data interpretation

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS
**27**

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS
**27**

**LIIS KOLBERG**

# Developing and applying bioinformatics tools for gene expression data interpretation

UNIVERSITY OF TARTU
Press

Institute of Computer Science, Faculty of Science and Technology, University of Tartu, Estonia.

Dissertation has been accepted for the commencement of the degree of Doctor of Philosophy (PhD) in informatics on May 18th, 2021 by the Council of the Institute of Computer Science, University of Tartu.

*Supervisor*

Dr. Hedi Peterson
Institute of Computer Science, University of Tartu
Tartu, Estonia

*Opponents*

Dr. Martina Summer-Kutmon
Maastricht University
Maastricht, Netherlands

Dr. Kerrin Small
King's College London
London, United Kingdom

The public defense will take place on June 21st, 2021 at 11:15 via Zoom.

The publication of this dissertation was financed by the Institute of Computer Science, University of Tartu.



ISSN 2613-5906
ISBN 978-9949-03-620-2 (print)
ISBN 978-9949-03-621-9 (PDF)

University of Tartu Press
`http://www.tyk.ee/`

*"Things can change. Life is all a learning process."*
*—Mike Tyson*

# ABSTRACT

Laboratories worldwide produce a sheer amount of experimental biological data. Data analysis and interpretation are the key elements in gaining actionable information from these data. Furthermore, due to the advent of high-throughput technologies, the dimensionality of these datasets has substantially increased. Analysing such high-dimensional data requires good programming skills or the help of a bioinformatician. However, the researchers who produce the data and ask scientific questions about it outnumber the bioinformaticians. Therefore, instead of performing custom analyses case-by-case, computational researchers develop easy-to-use tools that enable numerous biologists to perform the analyses themselves. These tools empower researchers to mine large-scale biological data, such as gene expression data, and assemble the knowledge into a working model of the system studied.

In this thesis, we develop two bioinformatics tools designed for gene expression data interpretation. First, we present the updated g:Profiler toolset for gene list functional enrichment analysis. Specifically, we focus on the advanced features and visual presentation of the functional characterisation results using a novel Manhattan plot approach. In addition, we introduce an accompanying R package *gprofiler2* that enables easy integration of g:Profiler with automated analysis pipelines. Next, we present the funcExplorer web tool that combines gene expression clustering and functional enrichment analysis using g:Profiler to detect co-expressed gene modules and provide a global overview of the experimental data. Thus, funcExplorer assembles the whole analysis process from gene expression data to biological interpretation.

Finally, we demonstrate the benefits of these tools by employing them in a systematic analysis of genetic variants that are associated with gene expression levels. Namely, we performed association analysis between genetic variants and co-expression gene modules. Among other methods, we used funcExplorer to detect the co-expressed gene groups, and we used g:Profiler to describe the associated gene modules via shared biological functions and pathways. Our results confirmed several previously known associations. Furthermore, applying funcExplorer to data obtained from stimulating human monocytes with lipopolysaccharides (LPS) for 24 hours led to discovering a novel association where a genetic variant near *SLC39A8* regulates a module of co-expressed metallothionein genes. Interestingly, we found that this effect was mediated by a transient association present only in early LPS response and lost before the gene module effect appeared. Specifically, the genetic variant affected the expression of *SLC39A8* hours earlier, shortly after stimulation with LPS.

The tools developed in this work have proven to be useful to the life science community, as demonstrated by millions of queries each year and close to thousand citations they have received since the latest publications. The tools are meant to help researchers to perform common analysis tasks by themselves. However,

we also conducted an extensive application study by ourselves that further highlighted the utility of these tools for unravelling novel biological insight from gene expression data in a data-driven manner.

# CONTENTS

# LIST OF ORIGINAL PUBLICATIONS

## Publications included in the thesis

I. Raudvere U[*], **Kolberg L**[*], Kuzmin I, Arak T, Adler P, Peterson H, Vilo J. **g:Profiler: a web server for functional enrichment analysis and conversions of gene lists (2019 update).** Nucleic Acids Research. 2019 Jul 2;47(W1):W191-8. Full text given on page 96.

[*] Authors contributed equally.

II. **Kolberg L**, Raudvere U, Kuzmin I, Vilo J, Peterson H. **gprofiler2–an R package for gene list functional enrichment analysis and namespace conversion toolset g: Profiler.** F1000Research. 2020 Jul 15;9(709):709. Full text given on page 106.

III. **Kolberg L**, Kuzmin I, Adler P, Vilo J, Peterson H. **funcExplorer: a tool for fast data-driven functional characterisation of high-throughput expression data.** BMC Genomics. 2018 Dec;19(1):1-7. Full text given on page 126.

IV. **Kolberg L**, Kerimov N, Peterson H, Alasoo K. **Co-expression analysis reveals interpretable gene modules controlled by *trans*-acting genetic variants.** Elife. 2020 Sep 3;9:e58705. Full text given on page 145.

## My contributions to these publications

Publication I – I am a member of the g:Profiler development team. My contribution to g:Profiler is broad, starting from the strategic planning and feature development to revising statistical methods and visualisations.
For the publication, I conducted the use case, prepared the figures and wrote the manuscript together with H. Peterson.

Publication II – I led the development of the *gprofiler2* R package. I designed and implemented the majority of the functionality of the package, including the novel visualisation functions. I documented the functionality, conducted the use cases, and wrote the manuscript.

Publication III – I led the development of the funcExplorer web tool. I devised and implemented the computational methods, developed the web tool, conducted all the analyses, prepared the figures, case studies, and comparisons for evaluation, and wrote the manuscript.

Publication IV – I conducted vast majority of the analyses, implemented the computational pipelines, aggregated and interpreted the results, and performed the literature-based replication. I designed the project and wrote the manuscript together with K. Alasoo. I prepared the figures for the manuscript.

## Publications not included in the thesis

This list covers other publications with my contribution that are not discussed in this thesis.

V. Kerimov N, Hayhurst JD, Manning JR, Walter P, **Kolberg L**, Peikova K, Samoviča M, Burdett T, Jupp S, Parkinson H, Papatheodorou I, Zerbino DR, Alasoo K. **eQTL Catalogue: a compendium of uniformly processed human gene expression and splicing QTLs.** BioRxiv. 2020 Jan 1.

VI. Reisberg S, Galwey N, Avillach P, Sahlqvist AS, **Kolberg L**, Mägi R, Esko T, Vilo J, James G. **Comparison of variation in frequency for SNPs associated with asthma or liver disease between Estonia, HapMap populations and the 1000 genome project populations.** International Journal of Immunogenetics. 2019 Apr;46(2):49-58.

VII. James G, Reisberg S, Lepik K, Galwey N, Avillach P, **Kolberg L**, Mägi R, Esko T, Alexander M, Waterworth D, Loomis AK. **An exploratory phenome wide association study linking asthma and liver disease genetic variants to electronic health records from the Estonian Biobank.** PloS One. 2019 Apr 12;14(4):e0215026.

VIII. Reimand J, Arak T, Adler P, **Kolberg L**, Reisberg S, Peterson H, Vilo J. **g:Profiler – a web server for functional interpretation of gene lists (2016 update)**. Nucleic Acids Research. 2016 Apr 20;44(W1):W83-9.

IX. Altmäe S, Tamm-Rosenstein K, Esteban FJ, Simm J, **Kolberg L**, Peterson H, Metsis M, Haldre K, Horcajadas JA, Salumets A, Stavreus-Evers A. **Endometrial transcriptome analysis indicates superiority of natural over artificial cycles in recurrent implantation failure patients undergoing frozen embryo transfer.** Reproductive BioMedicine online. 2016 Jun 1;32(6):597-613.

# LIST OF ABBREVIATIONS

| | |
|---|---|
| API | Application programming interface |
| bp | base pair |
| CSV | Comma-separated text file |
| DNA | Deoxyribonucleic acid |
| eQTL | Expression quantitative trait loci |
| FC | Fold change |
| FDR | False Discovery Rate |
| GEO | Gene Expression Omnibus |
| GMT | Gene Matrix Transposed file format |
| GO | Gene Ontology |
| GSEA | Gene set enrichment analysis |
| GTEx | Genotype-tissue expression project |
| GUI | Graphical user interface |
| GWAS | Genome-wide association study |
| ICA | Independent component analysis |
| KEGG | Kyoto Encyclopedia of Genes and Genomes |
| LD | Linkage disequilibrium |
| LMM | Linear mixed model |
| LPS | Lipopolysaccharides |
| MDS | Multidimensional scaling |
| mRNA | Messenger ribonucleic acid |
| ORA | Over-representation analysis |
| PCA | Principal component analysis |
| PEER | Probabilistic estimation of expression residuals |
| PLIER | Pathway-level information extractor |
| PIP | Posterior inclusion probability |
| QTL | Quantitative trait loci |
| RNA | Ribonucleic acid |
| SDA | Sparse decomposition of arrays |
| SNP | Single-nucleotide polymorphism |
| SuSiE | Sum of single effects |
| SVD | Singular value decomposition |
| WGCNA | Weighted correlation network analysis |

# INTRODUCTION

The main focus of biologists is understanding the underlying mechanisms responsible for the functioning of living organisms and how they interact with each other and the environment. Starting from the hypothesis formation, researchers conduct experiments and data analysis to evaluate the validity of a scientific theory and to translate the understanding to actionable insights and new hypotheses. For example, deciphering the complex biological processes underlying a common disease has the potential to uncover the medical means to intervene or prevent the disease development process.

Technological advancements in the life sciences, such as the advent of microarrays and next-generation sequencing, have offered new avenues to ask and answer research questions in a more detailed and less time-consuming manner. The abundance of publicly available experimental gene expression data suggests that, at first, the focus was on conducting experiments and gathering the data. By now, it is clear that collecting a massive amount of data alone is not enough for biological discovery. Researchers turn to computational data analysis to convert huge volumes of gene expression data to meaningful insights and interpretation. However, data analysis usually involves the successive or simultaneous application of several scripts and sophisticated methods. Furthermore, to make the most of existing knowledge, the analysis often combines multiple data types and sources. To simplify this process for a researcher without extensive programming skills, various bioinformatics tools that automate these steps have emerged. Besides, such tools are also beneficial to bioinformaticians for obtaining a quick overview of data before conducting a thorough analysis.

Creating helpful software for biologists is one of the most visible and influential results of bioinformatics research and development. However, from the developer's perspective, several challenges need to be considered while developing a tool for biological interpretation. First, to make life easier for biologists, the tool should be intuitive to use. Providing good documentation and practical examples plays an essential role in this purpose by ensuring that the user can understand what the tool does and can make the right decisions while applying it. Second, although interpretation requires a good insight and a bit of creativity, a comprehensive and easy-to-understand visual overview provided by the analysis software can have a fundamental role in this process. Third, tools that provide hypothesis-free exploration by letting the data speak for themselves enable discovering interesting patterns without pre-notion of what to look for. Fourth, the results need to be conveniently shareable with colleagues to initiate relevant discussions. Finally, to facilitate the publication of the results, the tools should ensure scientific reproducibility by providing all the used parameters and data versions.

Although there is a wide selection of tools for biological interpretation, many of these have shortcomings in one or more of the qualities mentioned above. To alleviate this, the main goal of this thesis is to develop convenient bioinformatics

tools for the biological interpretation of gene expression data that enable to move from expression changes to an understanding of implicated biological processes. Specifically, we introduce different properties of the tools presented in this thesis to emphasise the relevance of versatile interfaces, proper visualisations, and interactive features in the software that focus on data exploration and interpretation. Finally, we demonstrate the utility of the developed tools by applying these in a large-scale genetic associations study.

The thesis is organised as follows. The first chapter gives a short overview of gene expression data and analysis methods essential to understand the rest of the thesis. The chapter focuses mainly on different co-expression analysis methods followed by functional enrichment analysis to map the co-expressed genes to existing biological knowledge. The following chapters give more specific context and summarise shortly the publications I-IV, mainly from the perspective of my contribution. Specifically, Chapters 2 and 3 cover two tools developed for biological interpretation, updated functional enrichment analysis toolset g:Profiler (Publications I-II) and co-expression analysis web tool funcExplorer (Publication III). Chapter 4 first introduces the standard methods and data used to associate genetic variants with gene expression levels. This introduction is followed by demonstrating how co-expression analysis methods combined with functional enrichment analysis help to interpret genetic associations that affect gene expression (Publication IV), which in turn has the potential to provide biological context for common diseases. The thesis is concluded with a discussion on future work and some reflections on developing academic software in the bioinformatics field.

# CHAPTER 1

# PRELIMINARIES

This chapter provides a short overview of high-throughput gene expression data and analysis methods applied in this thesis for the biological interpretation of these data.

## 1.1. Gene expression data

The genetic information of most living organisms is stored in deoxyribonucleic acid (DNA) that resides in the cell's nucleus. DNA usually occurs as chromosomes, and the set of all chromosomes in a cell makes up its genome. The genome contains genes that are regions of DNA that carry instructions on how to produce a functional product. These products are needed to perform different jobs in the cells and are essential for the structure, function, and regulation of tissues and organs. In most cases, the product of a gene is a protein. The process of producing functional products involves two main steps. First, DNA is transcribed into messenger ribonucleic acid (mRNA), and then, in the case of protein-coding genes, the sequence of mRNA is used to produce proteins via a process called translation. In the case of non-protein-coding genes, RNA is not translated into a protein. Regardless of the type of gene, the process of going from DNA to a functional gene product is known as gene expression. By controlling which genes are expressed, the cell controls the protein production and thereby regulates its size, shape, and activity. This, in turn, affects higher level phenotypes such as height, weight, skin color, or characteristics of certain types of diseases. Therefore, to understand how an organism functions at the cellular and molecular level, we study its gene expression levels. Moreover, there can be thousands of genes expressed in a particular cell that potentially determine what that cell can do, and a single gene can result in a gene product that plays a role in many different phenotypes. Thus, we are often interested in a more global view on how different genes act together to produce and maintain a fully functioning organism.

Gene expression profiling experiments have enabled the investigation of thousands of genes simultaneously. Often this process includes measuring the expression of every gene present in a particular cell. Ideally, gene expression levels would be detected by quantifying the final gene product, which for many genes is the protein. However, for gene expression profiling, it has been a common practice to measure the intermediate step between the genes and proteins, the mRNA transcripts, as a proxy for protein levels. Although the extent of correlation between the protein and mRNA abundance has been debated (Fortelny et al., 2017), obtaining and analysing mRNA-level data has been the easiest way to get the most global

picture possible in a single experiment. The reason is that large-scale screening of protein levels remains challenging. At the same time, there are well-established and cheaper high-throughput profiling technologies for measuring mRNA levels such as microarrays (Schena et al., 1995) and RNA sequencing (RNA-seq) (Wang et al., 2009). A wide selection of gene expression profiling data under different biological conditions is available from public repositories such as Gene Expression Omnibus (GEO) (Edgar et al., 2002) and ArrayExpress (Athar et al., 2019). A typical outcome of a gene expression profiling experiment is a numeric matrix where the probes or transcripts (hereinafter genes) are in the rows, samples in the columns, and corresponding activity levels in the cells. This matrix is a common starting point for applying computational methods to extract biological knowledge from the data. From now on, we refer to the matrix format when talking about gene expression data. Usually, the dimensions of a human gene expression matrix are around 22,000 rows and few hundred columns.

## 1.2. Gene expression analysis

Extracting meaningful information from high-dimensional gene expression data can be quite challenging, mostly because mining these data requires several computational steps to gain insights. A standard gene expression analysis pipeline starts from pre-processing and normalisation of the gene expression matrix. Then, gene lists that exhibit similar expression patterns across the samples are identified. For example, these gene lists are detected by performing differential gene expression analysis or co-expression clustering. Finally, the obtained gene lists are characterised by investigating the shared biological functions based on the pre-existing knowledge about these genes. In addition, different parts of this pipeline include composing supporting visualisations.

The abundance of gene expression profiling experiments has given rise to statistical methods for analysing and interpreting gene expression data. However, there is no one-size-fits-all solution. The choice of the analysis technique depends on the data and the goal of the experiment. Fortunately, with the joint efforts of biologists, statisticians, computer scientists, and software engineers, a wealth of computational tools are available for researchers who want to mine the data for knowledge. The following sections give a brief overview of the common methods and tools applied for these fundamental tasks in the context of this thesis.

### 1.2.1. Differential gene expression analysis

The most basic gene expression analysis identifies genes whose expression levels present significant changes between two or more experimental settings, for example, between samples from healthy and diseased individuals or between stimulated and unstimulated cells. These genes can be used to describe the differences between the conditions and are called differentially expressed genes. There are

various statistical methods used to detect such genes. Generally, the methods consist of three parts: calculating a test statistic, determining the significance of the observed statistic, and adjusting for multiple testing.

In the simplest case, a t-test is performed for each gene. Simply put, this test compares the mean expression levels of the specific gene between two experimental groups at a time and estimates whether the difference is statistically significant. In addition, a fold change (FC) of the mean expression levels between the two groups is calculated to evaluate the change in the expression level of a gene. The fold change is usually presented in a log2 scale that makes the measure symmetric when the change decreases by an equivalent amount. This approach is applied on every row of the gene expression matrix returning corresponding p-values that are then used to identify significant genes. This identification is made after adjusting for multiple testing to limit the number of false-positive results. In addition, there are more sophisticated approaches for differential gene expression analysis depending on the specific properties of the data-producing technology, such as different linear models for microarrays (Ritchie et al., 2015) or model-based estimations using the negative binomial distribution for RNA-seq data (Love et al., 2014; McCarthy et al., 2012). Regardless of the method, the outcome of differential expression analysis is a gene list or two gene lists representing up and down-regulated genes separately. If there are more than two experimental groups, the same procedure is usually repeated for each possible pair of conditions.

### 1.2.2. Regression analysis

As we mentioned in the previous section, linear models can be used to find differentially expressed genes. Furthermore, linear models are also employed in various other contexts when studying gene expression data. This section gives the basics about linear regression for analysing gene expression data. Although linear regression is often used in the prediction context, here, we consider linear regression to quantify a relationship between two variables. For example, such models are used in different association studies that are further discussed in Chapter 4 of this thesis.

Simple linear regression is the most basic approach to study the relationship between a quantitative response variable $Y$ and a single explanatory variable $X_1$, also known as dependent and independent variables, respectively. Regression analysis attempts to model the relationship between the two variables by fitting the linear equation 1.1 to the observed data. In the case of gene expression data, the regression models are used to identify the strength of the effect that the independent variable has on the gene expression. The method models the response variable $Y$, i.e. the gene expression levels, in relation to the explanatory variable $X_1$ as

$$Y = \beta_0 + \beta_1 X_1 + \varepsilon, \tag{1.1}$$

where $\beta_0$ and $\beta_1$ are the coefficients, i.e., an intercept and a slope, and $\varepsilon$ is an error term.

The independent variable can be continuous or categorical. For example, it can be the age or weight of an individual or other measured phenotypes on a sample. When the variable $X_1$ is binary, e.g., a disease state, this model can be used to evaluate differential expression between the two groups. In this case, $X_1$ is an identifier variable with a value of 0 for healthy and 1 for diseased samples. Another example is using linear regression to estimate the effect size $\beta_1$ of a genetic variant on gene expression where genotypes are coded as (0, 1, 2).

Simple linear models can be extended to multiple explanatory variables. For example, $p$ variables are included in equation 1.2. This model is called multiple linear regression. In addition to including these variables whose effects on the gene expression we are particularly interested in, different confounding variables are included as other covariates in the multiple linear regression model to account for these effects. Depending on the study, these covariates can be age, sex, population membership, or batch variables, to name a few.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon \tag{1.2}$$

Alternatively, linear mixed models (LMMs) are sometimes used to control for confounding factors. LMMs differ from simple linear models by incorporating random effects among the explanatory variables (Silk et al., 2020). In this case, factors that are not of particular interest to the study but confound the dependent variable are treated as random effects. In contrast, the fixed effects represent variables with intercepts and slopes to be estimated. We refer to (Silk et al., 2020) for a review about the applications of LMMs in biology.

### 1.2.3. Co-expression analysis

Co-expression is defined as a simultaneous expression of two or more genes across several conditions. Different metrics are used to measure co-expression, the most common ones being Pearson's correlation and Euclidean distance-based similarity. The higher the similarity, the more strongly co-expressed the two genes are. Co-expression analysis is an approach that uses this information to identify groups of co-expressed genes. In this thesis, we consider gene lists resulting from any co-expression analysis as gene modules, and we use the words 'cluster' and 'module' as synonyms.

Co-expression analysis identifies trends and patterns from gene expression profiles. In addition, this analysis is often used as a tool to study genes without known function. Namely, the idea of finding groups of co-expressed genes relies on the assumption that if two genes have similar expression profiles across different biological conditions, then these two genes are considered to be similar also at the functional level (Eisen et al., 1998; Oliver, 2000). Based on this assumption, less studied genes are characterised using the available information of well-known genes within the same co-expression group. This approach is also known as the

guilt-by-association approach (Oliver, 2000), and the underlying assumptions imply the subsequent analysis steps for identifying enriched biological functions and pathways in the co-expressed gene lists (described in section 1.3).

Various computational techniques are used for co-expression analysis, including matrix factorisation methods such as singular value decomposition (SVD), different clustering methods, and co-expression networks. A common feature of all these methods is that they simplify the data while still retaining the inherent structure (Slonim, 2002). Ideally, these are data-driven and hypothesis-free methods that derive the underlying patterns solely from the gene expression data rather than reflecting the researcher's intuition or experience. The following sections introduce the most common clustering and matrix factorisation methods used for co-expression analysis.

### Clustering methods.

Clustering is one of the most popular analysis approaches to get a quick overview of patterns present in the gene expression data (D'haeseleer, 2005; Jiang et al., 2004). Clustering aims to partition a set of items, in our case the genes, into subgroups called clusters so that, based on a pre-selected similarity measure, similar genes belong to the same cluster and dissimilar genes fall in different clusters. A variety of clustering algorithms can be used for this task, the most common ones being hierarchical clustering and K-means clustering (D'haeseleer, 2005; Jiang et al., 2004).

Hierarchical clustering was the first algorithm used in gene expression data clustering (Eisen et al., 1998). Agglomerative hierarchical clustering starts with every gene as a separate cluster and successively joins the most similar clusters until all the genes are organised as a tree-like arrangement called dendrogram (Jiang et al., 2004). Thereby, in addition to forming the gene clusters, the dendrogram also illustrates the similarity between different clusters. Specific clusters are detected based on cutting the dendrogram at some level that depicts a fixed threshold for expression similarity (see Figure 1). As a result, the dendrogram is partitioned into sub-branches representing clusters of highly co-expressed genes. However, the cutting point is often decided by the researcher based on visual observation and thus is subjective.

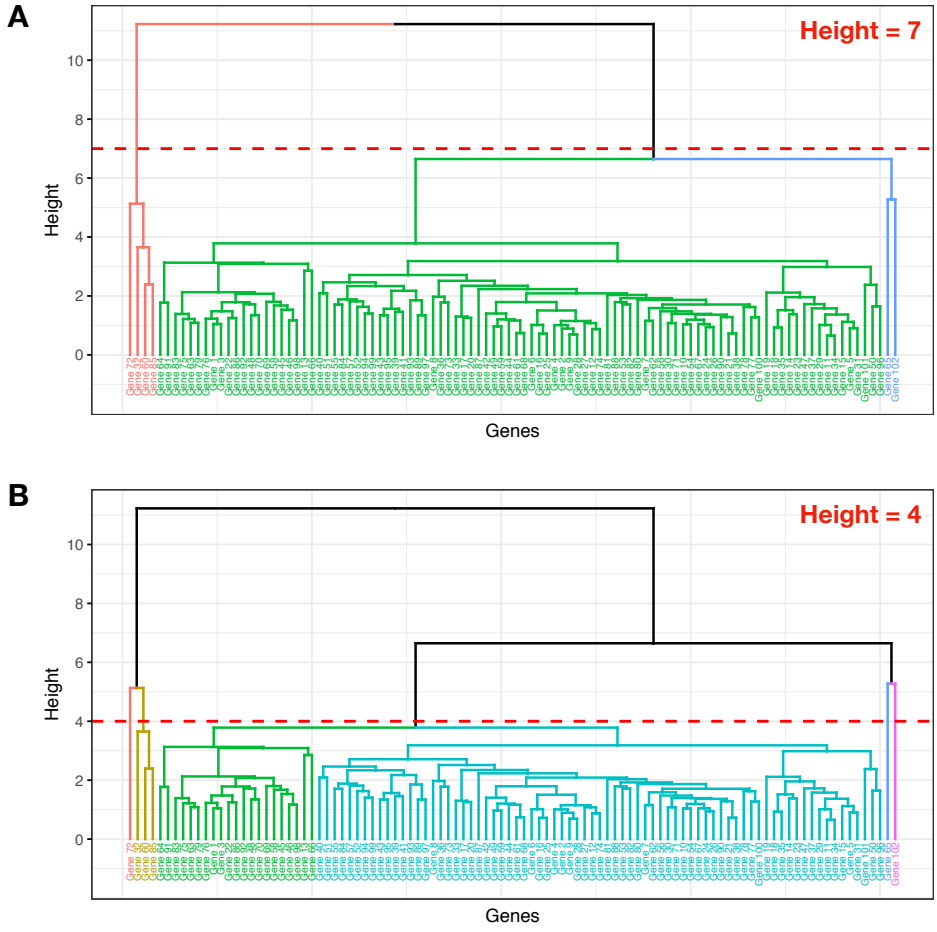



**Figure 1. Detecting clusters from a dendrogram.** Hierarchical clustering arranges genes as a dendrogram based on the similarity of their expression profiles shown on the y-axis as the height of the dendrogram. In hierarchical clustering, the clusters are usually extracted based on cutting the dendrogram at some fixed level. **A** Cutting a dendrogram at height seven results in detecting three gene clusters (colored in pink, green, and blue). **B** Cutting the same dendrogram at height four results in six gene clusters, three of which include only a single gene (colored in pink, olive green, green, light blue, blue, and purple).

Another approach is to determine the number of clusters beforehand. K-means is a well-known algorithm used for this purpose, where, given a pre-specified number $K$, the algorithm randomly selects $K$ genes as cluster centroids and divides all the remaining genes between the $K$ clusters based on the expression profile similarity (Jiang et al., 2004). In the next iteration, the algorithm resets the centroids to the average of each cluster's gene profiles and re-assigns the genes to the clusters. This process repeats until there are no changes in gene clusters. However, similar to hierarchical clustering, the key limitation of this approach is that the value of $K$ also relies heavily on the researcher's prior knowledge or

assumptions and judgment and thus is biased (Jiang et al., 2004; Quackenbush, 2001; Slonim, 2002).

In recent years, the weighted correlation network analysis (WGCNA) (Langfelder and Horvath, 2007, 2008; Zhang and Horvath, 2005) has become popular for gene expression clustering. Simply put, the method first constructs a weighted gene co-expression network where the genes are nodes, and edges represent the correlation between the corresponding pairs of genes. Next, WGCNA searches for densely interconnected gene groups called modules using hierarchical clustering followed by the Dynamic Tree Cut method (Langfelder et al., 2008). In the case of high-dimensional gene expression data, a variant of K-means is used to pre-cluster the data into large clusters, referred to as blocks, and hierarchical clustering and module detection is applied to each block. Finally, highly similar modules from different blocks are merged.

A common feature of standard clustering methods is that they partition the gene expression matrix into non-overlapping groups of genes (Saelens et al., 2018). Moreover, clustering algorithms always find clusters, regardless of whether there are any biologically meaningful patterns in the data (Altman and Krzywinski, 2017). However, considering the connections between biological processes and that genes can participate in several functions, such methods might not capture the complexity of the underlying data structure. Alternatively, there are also clustering methods that are so-called fuzzy, such as the fuzzy C-means algorithm (Dembele and Kastner, 2003), where each gene can potentially belong to multiple clusters, which is evaluated by a membership score. Finally, clustering methods look at co-expression across all samples and thus might miss local co-expression patterns present in only a subset of samples (Brunet et al., 2004; Saelens et al., 2018).

## Matrix factorisation methods.

Besides clustering methods, matrix factorisation methods are used to detect co-expressed gene modules from the gene expression data (Saelens et al., 2018). The idea of factorisation methods is to project the input data matrix into a product of two smaller matrices while preserving as much information as possible from the original data (see Figure 2) (Saelens et al., 2018; Stein-O'Brien et al., 2018). For this reason, matrix factorisation is sometimes also called matrix decomposition.

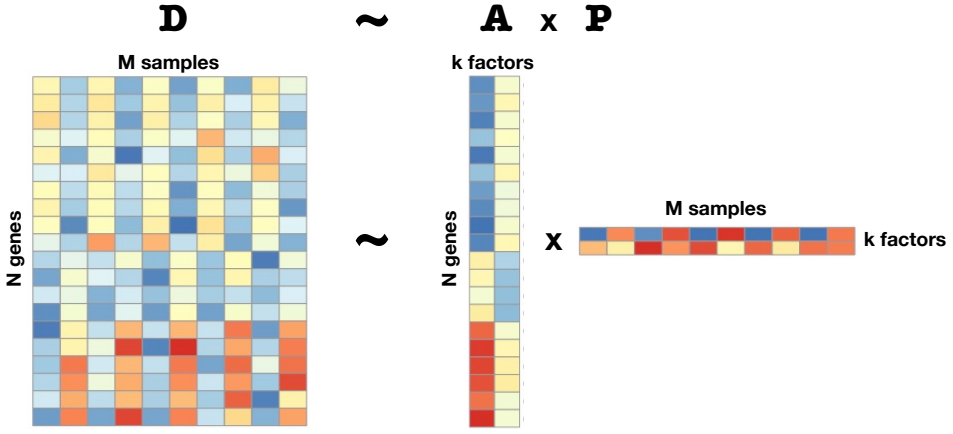



**Figure 2. Illustration of matrix factorisation technique.** Given a gene expression matrix $D$ with dimensions $N \times M$, where $N$ is the number of rows and $M$ is the number of columns, matrix factorisation methods try to decompose this data matrix into two related matrices: gene-level matrix $A$ and sample-level matrix $P$. The values in the matrices $A$ and $P$ correspond to the weight by which the given gene or sample contributes to the particular inferred latent factor. These $k$ factors represent the detected gene modules. In the case of matrix $P$, we call these weights as factor loadings.

The goal of matrix factorisation methods is to infer a set of new variables using a linear combination of the original variables, such that the number of new variables is smaller than in the initial data (Meng et al., 2016). Depending on the specific method, these new variables are called latent factors, latent variables, or components (Stein-O'Brien et al., 2018). Here, we consider these factors as representatives of gene modules and thus use these terms as synonyms. As a result, two matrices are obtained. One of these matrices contains weights for every gene and a particular latent factor, and the other matrix contains the weights for every sample and corresponding factor (see Figure 2). The latter are sometimes also called factor loadings. The gene-level weights are then used to extract gene modules corresponding to the factors. A way to define the genes that characterise each factor is by choosing the ones that contribute the most, i.e., the genes at both extremes of the weight values. For example, genes that are two standard deviations away from the mean weights in the given factor.

As described above, the gene-level matrix assigns a weight for contributing to a module for each gene. However, for a specific module, a vast majority of the

genes are given close-to-zero weights (yellow shades in the matrix $A$ in Figure 2). In the context of clustering, one could think as if the weights of genes not belonging to a particular cluster are set to zero. Furthermore, in matrix factorisation, a specific gene can contribute to multiple modules. Consequently, different modules can include overlapping sets of genes, better reflecting the complex nature of biological systems. Moreover, samples can contribute to a particular module only to a certain degree (yellow shades in the matrix $P$ in Figure 2), and thus matrix factorisation methods have the potential to detect local co-expression, unlike clustering methods (Saelens et al., 2018).

Various matrix factorisation techniques have been used for gene expression data, the best known being SVD and the closely related principal component analysis (PCA), independent component analysis (ICA) (Hyvärinen and Oja, 2000), and non-negative matrix factorisation (Stein-O'Brien et al., 2018; Way et al., 2020). While each of these methods is defined by a distinct mathematical formulation, the common goal remains the same, to reveal low-dimensional structure from high-dimensional data. For a detailed review of matrix factorisation methods and their applications in omics data, we refer to (Stein-O'Brien et al., 2018).

In addition, there are also matrix factorisation methods that are designed for analysing biological data. For example, probabilistic estimation of expression residuals (PEER) is a factor analysis method that uses Bayesian approaches to infer hidden factors from gene expression data that explain a large proportion of expression variability (Stegle et al., 2012). Pathway-level information extractor (PLIER) is a matrix decomposition method that uses prior biological knowledge of pathways and gene sets to deconvolve gene expression profiles as a product of a small number of latent variables (factors) and their gene weights (Mao et al., 2019).

### 1.2.4. Eigengenes

Every co-expression gene module represents a group of genes with high similarity in expression profiles, thus indicating potential similarities in biological processes. Intuitively, from the expression of each gene in the module, one could infer a single characteristic expression profile that summarises the correlated behavior of the module's genes across the samples. This representative expression profile is called eigengene (Alter et al., 2000; Langfelder and Horvath, 2007). Similar to the expression profile of a single gene, an eigengene is a vector that contains as many elements as there are samples in the data.

There can be different approaches to derive such profiles. The simplest one is selecting a random representative gene from the module and using its expression levels as an eigengene profile. However, none of the actual genes in the module need to have the exact expression value presented in the eigengene. Therefore, another naive option would be to calculate the average gene expression within the module. But this approach gives each gene in the module an equal weight which

might not always be justified. For example, since most clustering algorithms assign every gene in the data to some cluster, some genes might belong to a module due to a random chance. To overcome this shortcoming, a weighted average expression profile can be used (Langfelder and Horvath, 2008). The equation 1.3 represents an eigengene vector $\mathbf{e}$ for a module of $p$ genes, where $\mathbf{g_i}$ is the expression vector of gene $i$ in that module and $\alpha_i$ are the corresponding weights. Note that in the case of standard average value, the weights $\alpha_i = \frac{1}{p}$.

$$\mathbf{e} = \alpha_1 \mathbf{g_1} + \alpha_2 \mathbf{g_2} + \ldots + \alpha_p \mathbf{g_p} \tag{1.3}$$

A common approach for finding such weighted profiles was first described by (Alter et al., 2000) who applied the singular value decomposition method on the gene expression data to reduce a high-dimensional data set into fewer dimensions while retaining important information. In essence, SVD is a matrix factorisation method that defines a linear transformation from the genes $\times$ samples space to the reduced eigensamples $\times$ eigengenes space, i.e., it obtains two matrices, just as shown in Figure 2. The first matrix describes the 'eigensamples' and the second one describes the 'eigengenes'. Thus, the factor loading vectors represent the weighted eigengenes in this case. The columns in the 'eigensamples' matrix correspond to the weights $\alpha_i$. Similarly, in the case of other matrix factorisation methods, the factor loading vectors by definition capture the characteristic linear transformations essential for defining the eigengene profiles (Alter et al., 2000; Liebermeister, 2002; Stein-O'Brien et al., 2018).

The eigengene concept can also be transferred to the clustering context. Namely, when performing SVD analysis on the sub-expression matrix of a single gene module, then the corresponding eigengene is defined by the first vector in the decomposition matrix, i.e., the first latent factor. Equivalently, this vector is the first principal component of the PCA. However, since the sign of each principal component scores vector is arbitrary, the orientation of each eigengene is then fixed by constraining it to have a positive correlation with the module's average gene expression profile (Langfelder and Horvath, 2007). This ensures that the eigengene profile corresponds to the direction of expression changes in the module. For example, such an eigengene detection approach is implemented in the WGCNA clustering method (Langfelder and Horvath, 2008). However, this approach can be applied to gene clusters from any method.

Regardless of the co-expression method or the eigengene definition used, eigengene profiles are linear combinations of expression levels of the genes in the modules. Therefore, the eigengene is correlated with the expression profiles of the module's genes and shows how the expression level of that group of genes changes between conditions. In the presence of additional meta-information, these expression profiles can provide further context for co-expression interpretation. For example, if an eigengene has increased expression levels in the group of diseased samples compared to the healthy ones, one could assume that the module's genes

may be affected by the disease status. If there are no changes in the profiles between healthy and diseased samples, these genes are probably not interesting when studying that particular disease. Eigengenes have also been used to compose co-expression networks to describe the relationships between different gene modules instead of individual genes (Langfelder and Horvath, 2007).

Furthermore, eigengene profiles can be used to relate the gene modules to other traits (Langfelder and Horvath, 2008). Genes with high weights in modules associated with some characteristics are natural candidates for further studies (Langfelder and Horvath, 2008). For example, (Tian et al., 2020) detected a module where the eigengene was correlated with the pathological grade of breast cancer across the patients. Another study used various physiological traits such as body weight, cholesterol level, and insulin level to describe different gene modules (Ghazalpour et al., 2006). A use case where eigengene profiles are associated with genotypes data is further discussed in Chapter 4.

### 1.2.5. Gene expression visualisation

Simply by looking at the numeric vectors or matrices, it is almost impossible to comprehend all the information, especially in the case of multidimensional and complex gene expression data. Therefore, different visualisation techniques are used to get an overview of the data at a glance.

The most common way to visualise gene expression data is a matrix where the numeric expression values in a cell are shown with a color gradient (see Figure 3A). This type of visualisation is called a heatmap. The color gradient in a heatmap usually ranges from one color to another, corresponding to values from the minimum expression through the average expression to the maximum expression in the data. For example, shades of blue are used for low and shades of red for high expression values. As a result, the groups of genes or samples with similar expression levels are shown with similar color, and thus such groups are easier to discern from the matrix by eye.

To better grasp the patterns from the heatmap, it is beneficial to reorder the rows and columns of the matrix according to some co-expression measure and method so that more similar expression profiles are near each other. The most typical approach for this is using hierarchical clustering and a corresponding dendrogram that highlights the gene (or sample) clusters and the relations between them (see Figure 3B).

However, the heatmap size grows with the growing dimensions of gene expression experiments so that the available screen space becomes a limiting factor for this visualisation. Large numerical matrices, including tens of thousands of genes and hundreds of samples, lead to overwhelming visuals. One option to achieve a more compact picture could be visualising only the eigengene profiles in a heatmap showing differences between the gene modules. Besides heatmaps, line plots where the x-axis shows the samples and the y-axis shows the expres-

sion values are often used to visualise expression profiles of genes or eigengenes (see Figure 3C). Line plots provide insight into the patterns of correlation between samples and expression levels. In addition, these plots enable to see trends in time or some particular behavior such as low expression in one group of samples and high in another group.

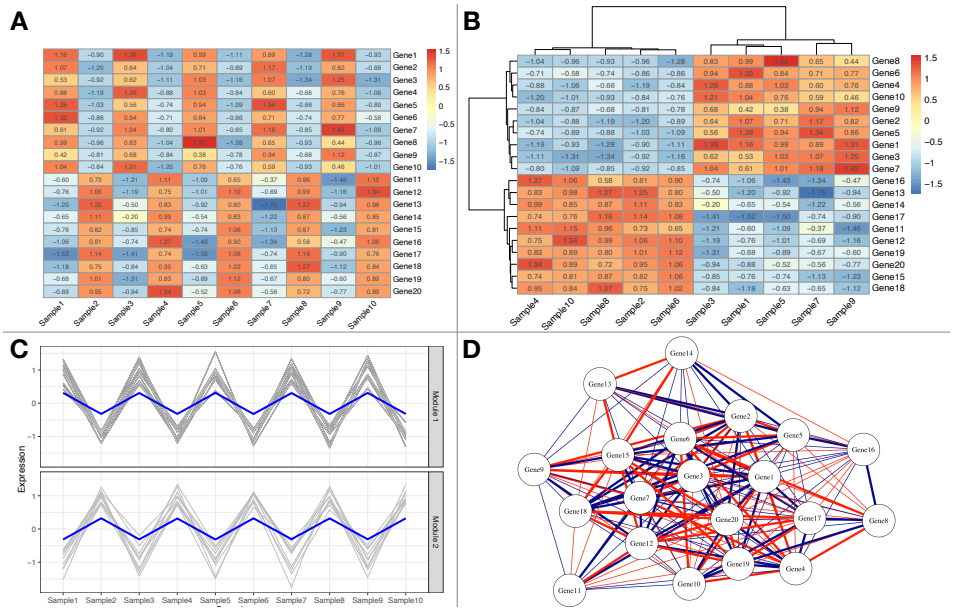



**Figure 3. Examples of visualising gene expression data.** **A** Heatmap showing the numeric values in a gene expression matrix by using a color gradient from blue to red. **B** Hierarchically reordered heatmap with corresponding dendrograms highlighting different co-expression groups in the data. **C** Line plots of expression profiles for two gene modules and corresponding eigengene expression profiles defined by the first principal component of the corresponding module are shown in blue. The x-axis corresponds to the samples, and the y-axis shows the expression levels. **D** Co-expression network where nodes illustrate the genes and edges represent Pearson's correlation between the corresponding genes. Edges with absolute correlation smaller than 0.9 are removed from the network. The blue and red edges indicate a negative and positive correlation, respectively. The edges corresponding to the correlation greater than 0.95 are highlighted with greater thickness.

In the case of co-expression network analyses, a natural way of visualisation is to compose a graph where nodes represent individual genes, and edges represent co-expression between the two genes or some functional connection (Merico et al., 2009) (see Figure 3D). Ideally, the co-expression gene modules in the network appear in visually distinguishable subgraphs in which each pair of nodes is connected with each other via an edge. Applying a significance threshold to the co-expression similarity to remove edges with similarity below a certain value (e.g. absolute correlation $< 0.9$) further helps reduce the complexity of the network and bring out the strongly correlated gene modules.

Finally, as the data are complex and static images provide limited benefits, interactive and navigable visualisations are becoming more and more popular by making data exploration easier (Cruz et al., 2019; Pavlopoulos et al., 2015).

## 1.3. Gene list functional interpretation

As demonstrated above, gene expression analysis usually results in lists of genes grouped based on similar expression patterns across the measured conditions. Regardless of the analysis method, or whether the results are gene modules from clustering or up and down-regulated genes from differential expression analysis, the next steps in the analysis pipeline find a functional characterisation for the observed behavior. One way to achieve this characterisation is to apply functional enrichment analysis tools that associate the genes with the existing biological knowledge from annotation databases to find the shared functions.

### 1.3.1. Functional annotations

Functional annotation is a statement about the function of a particular gene that is usually discovered by conducting various detailed experiments and analyses. Several dedicated databases gather and store knowledge about biological functions, their relations, and corresponding annotations of genes and gene products from published datasets and manuscripts. Here we introduce some of the most common annotation databases that are often used in functional enrichment analysis.

The most comprehensive annotation database used for functional enrichment analysis is the Gene Ontology (GO) that provides curated information from three different domains: molecular functions, biological processes, and cellular components (The Gene Ontology Consortium, 2021). Namely, the GO annotations capture the knowledge about how a gene functions at the molecular level, where in the cell it functions, and what biological processes it helps to carry out. All the functions in the database (also referred to as GO terms) have explicit definitions and unique identifiers. These functions are arranged in an ontology, a structured hierarchy with defined relationships between the terms. The ontology is represented as a directed acyclic graph depicted in a tree-like view where general biological functions are placed on the top and more specific terms to the bottom of the tree (Figure 4). Every node in the graph corresponds to a term, and the relations between the terms are shown with edges. The terms that are closer to the root of the graph are referred to as parents, and the terms more proximate to the leaf nodes, or bottom of the graph, are called child terms. An important feature is that the gene annotations in this hierarchical structure are propagated from the child term to the parent. That is, a gene annotation to a GO term implies that this gene is also annotated to all its parents. For example, the genes annotated to the lipid metabolic process (GO:0006629) are also annotated to the primary metabolic process (GO:0044238) and metabolic process (GO:0008152) (see Figure 4). Besides Gene Ontology, there are other structured databases. For

example, the Disease Ontology (Schriml et al., 2019) provides the relations between the diseases while the corresponding annotations can be mapped using text mining approaches (Osborne et al., 2009; Peng et al., 2012), for example. Similarly, Human Phenotype Ontology provides an ontology and annotations to terms that describe clinical abnormalities (Köhler et al., 2021).

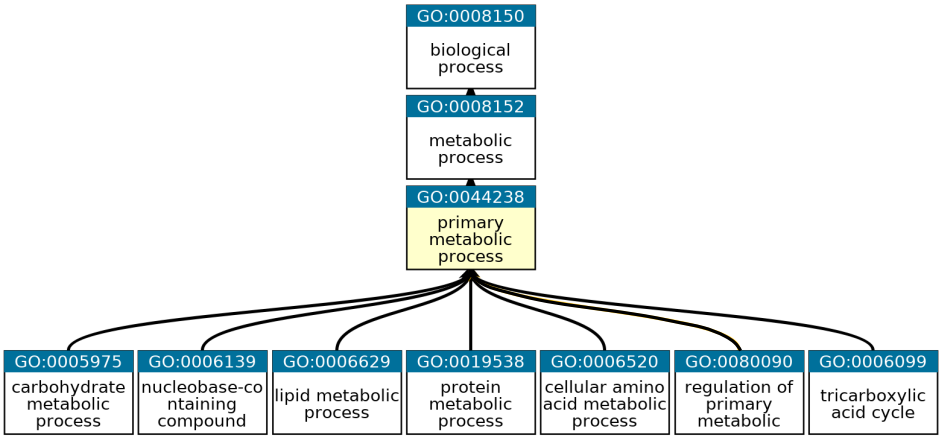



**Figure 4. Parent and child GO terms of primary metabolic process.** Each rectangle represents a GO term that is defined by a unique ID and human-readable description. The edges represent the relationships between the terms. General terms are shown on the top of the sub-tree, specific terms are shown on the bottom. The parent terms of the 'primary metabolic process' (in yellow) are 'metabolic process' and 'biological process'. The child terms are 'carbohydrate metabolic process', 'lipid metabolic process', etc. Figure adapted from the QuickGO tool (Binns et al., 2009).

Another and more detailed way to characterise biological processes is via pathways. In essence, pathways are models that represent the series of reactions and interactions between genes, proteins, or metabolites within cells, tissues, or organisms. Pathways are graphically illustrated as networks where nodes correspond to entities participating in the reactions (proteins, nucleic acids, complexes, small molecules) and edges represent the reactions (see `https://reactome.org/content/detail/R-HSA-1221632` for an example). The edges are often directed to represent the direction of the reaction. The genes that are present in a given pathway are used for the enrichment analysis. Pathways are collected to major dedicated databases where they are curated, queried, and visualised. A well-known pathway database is Reactome (Jassal et al., 2020) which includes biological pathways such as signaling, innate and acquired immune function, regulation, metabolism, and disease. Another such knowledge base that contains information about biochemical pathways and other types of molecular interactions is the Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa et al., 2021). In addition, there is a community resource dedicated to different biological pathways called the WikiPathways (Martens et al., 2021).

Finally, there are numerous other specific data sources such as transcription

factors and their motifs from Transfac (Wingender, 2008), miRNA-target interactions from mirTarBase (Chou et al., 2018) or protein complexes from CORUM (Giurgiu et al., 2019), to name a few. Furthermore, dedicated databases conveniently combine annotation data from multiple sources, making them especially useful for performing functional enrichment analysis described in the following subsection. Ensembl BioMart (Kinsella et al., 2011) consolidating several Ensembl databases, Molecular Signatures Database (MSigDB) (Liberzon et al., 2011) including a wide range of different gene sets, and DisGeNET (Piñero et al., 2020) focusing on human diseases are few examples of resources that provide different collections of annotated gene sets.

### 1.3.2. Functional enrichment analysis

Functional enrichment analysis, sometimes also known as the gene set analysis, is a method that performs statistical tests to evaluate if any biological functions are significantly enriched in the gene list of interest, as compared to all the genes in the genome. The gene list can be from a particular co-expression module or significant differentially expressed genes from the differential expression analysis. The biological functions are usually taken from the available annotation databases.

A common approach is over-representation analysis (ORA) which usually uses hypergeometric test or its equivalent: one-tailed Fisher's exact test. For every function and pathway, the test determines whether the fraction of genes annotated with the function in the given gene list is higher than expected by a random chance. The corresponding enrichment p-value $p_T$ of a single process, pathway, or other annotation $T$ for the gene list of interest $\mathbb{L}$ is calculated using the information shown in the contingency Table 1. Specifically, assuming that $k$ genes out of the total of $n$ genes in the gene list $\mathbb{L}$ are annotated with the function $T$, and there are $M$ genes annotated with $T$ among the total of $N$ genes in the genome (or the dataset at hand, depending on the background preference), the p-value $p_T$ is computed using the equation 1.4.

$$p_T = \sum_{i=k}^{\min(n,M)} \frac{\binom{M}{i}\binom{N-M}{n-i}}{\binom{N}{n}}. \tag{1.4}$$

These values are calculated for each function in an annotation database of interest, and a list of significantly enriched functions is returned.

**Table 1.** Contingency table of over-representation test. The values are used to calculate the enrichment p-value in Eq. 1.4 of function $T$ in a gene list $\mathbb{L}$.

| | genes in $\mathbb{L}$ | genes not in $\mathbb{L}$ | Total |
|---|---|---|---|
| genes in T | k | M-k | M |
| genes not in T | n-k | N-n-M+k | N-M |
| Total | n | N-n | N |

In the case of differential expression analysis, the gene list $\mathbb{L}$ is usually determined based on a cut-off on the p-value in combination with the $\log_2$ fold change

(e.g. adjusted p-value < 0.05 and $\log_2 FC > 2$). Thus, ORA focuses on a handful of genes from the top or bottom of a ranked gene list showing the largest difference between the two biological classes. However, this approach can make ORA results sensitive to the selected thresholds. That is, some thresholds can retrieve no genes, and others retain a very long list of genes without any unifying biological theme. In addition, all the genes are treated equally in the analysis.

An alternative enrichment analysis approach is Gene Set Enrichment Analysis (GSEA) (Subramanian et al., 2005). While ORA focuses only on significantly differentially expressed genes, the input of GSEA is a list of all the genes from the platform ordered by some suitable metric such as differential expression significance or fold change. For each function in the annotation database, the method determines whether the genes annotated with this function are accumulated to the top (or bottom) of this ranked list or are they distributed over the list randomly. If a function $T$ falls at either the top (over-expressed) or bottom (under-expressed) of the gene list, it is considered to be correlated with the phenotypic class distinction (Subramanian et al., 2005). Similarly to ORA, this method yields a list of statistically significant functions to interpret.

Since the enrichment tests are usually performed for tens of thousands of gene sets at a time (e.g., more than 40,000 GO terms can be considered for analysis of human gene lists), multiple testing correction is applied to reduce the amount of potential false-positive results. Different correction methods are used for this task. The most common ones being the Bonferroni correction method and Benjamini–Hochberg False Discovery Rate (FDR) (Benjamini and Hochberg, 1995). However, one of the shortcomings of these methods is that they assume that the performed tests are independent. But biological functions are not typically independent because of their overlapping genes and complex association structures. There are custom correction methods to take this dependence into account. For example, the enrichment tool g:Profiler (Reimand et al., 2007) provides a tailor-made algorithm g:SCS (Set Counts and Sizes) that estimates the significance threshold by considering the size of the input gene list and the intersections between the annotation gene sets. Other approaches account for the term dependencies already while performing the statistical tests. For example, the *elim* method iteratively removes the genes mapped to the significant GO terms from more general GO terms before calculating the p-value (Alexa et al., 2006). Nevertheless, the standard methods are still useful for filtering the enrichment results and keeping only the strongest signals.

Finally, a function is usually considered to be significantly over-represented in the input gene list if the corresponding $p\text{-}value \leq 0.05$ after multiple comparison correction. And a list of significant functional terms from the enrichment analysis is used to characterise the input genes.

### 1.3.3. Functional enrichment tools

Performing enrichment analysis requires a lot of computational work. This process includes accessing and combining different annotation databases, converting gene identifiers to the same namespace, counting numerous gene list overlaps, and finally performing the statistical tests. Fortunately, various existing web and stand-alone tools simplify these tasks for the researchers by automatically calculating and visualising the results for the input gene list. For example, g:Profiler (Raudvere et al., 2019), DAVID (Jiao et al., 2012), WebGestalt (Liao et al., 2019), Enrichr (Kuleshov et al., 2016) and Metascape (Zhou et al., 2019) are popular web tools used for this task. GSEA is performed by a stand-alone desktop application of the same name (Subramanian et al., 2005). In this thesis, we mostly focus on web tools.

The landscape of enrichment analysis tools is diverse, covering different annotation databases, species, identifier types, and analysis methods. While most of the services provide widely used resources such as GO, KEGG, and Reactome, other annotation sources vary between the tools. For example, biological pathways from WikiPathways (Martens et al., 2021) are available in Enrichr, WebGestalt, and g:Profiler, while transcription factors from Transfac database (Wingender, 2008) are included only by a few tools, e.g., Enrichr and g:Profiler. There are also species-specific tools, such as the agriGO platform that is mostly dedicated to plants and agricultural species (Tian et al., 2017). Most of the tools perform over-representation analysis. Some also offer GSEA-like approaches, such as WebGestalt and g:Profiler.

Besides different annotation sources, the enrichment tools consider different gene identifier types. Namely, the input gene lists of enrichment tools originate from a broad range of experimental platforms, each having unique default identifiers. As a result, there are numerous identifier types, some common ones being Ensembl, RefSeq, Entrez and UniProt IDs. For example, there are around hundred different identifiers for human data. Thus, enrichment web tools need to solve the challenge of converting between various gene, protein, microarray probe, or other types of identifiers. Specifically, this step aims to map the input gene list to the same namespace with the annotation sources to calculate the overlaps between the gene sets. However, most of the enrichment tools handle only a limited subset of possible identifier types, thus creating an obstacle that the users need to overcome manually or via external tools. For example, Enrichr works only with the gene symbols; Metascape also accepts RefSeq, Entrez and Ensembl identifiers as input, while other popular tools handle more extensive sets of identifier types. Therefore, the availability of suitable identifiers is an important quality to consider when choosing the right tool. Besides, different tools handle identifier mapping differently. For example, Metascape and WebGestalt map all the identifiers to Entrez IDs as a reference before the computations but g:Profiler maps to Ensembl identifiers. Because of this difference, the compared gene sets can vary in size be-

tween individual tools. Furthermore, the conversion is not always straightforward as there can be one-to-many mappings and vice versa. For example, the same gene can have multiple Ensembl identifiers, e.g. gene symbol RGS5 corresponds to two Ensembl IDs, ENSG00000143248 and ENSG00000232995. Services such as the Ensembl BioMart (Kinsella et al., 2011), or UniProt mapping (The UniProt Consortium, 2021) consolidate a wide range of identifiers that are often used as a mapping resource by the enrichment tools.

In addition to the range of data and identifier sources, user-friendly interface, and modern look, one should also consider reproducibility and transparency. A significant issue with the enrichment tools has been that most of them rely on outdated annotation data (Wadi et al., 2016), with some exceptions such as g:Profiler and PANTHER (Mi et al., 2021). Although awareness of this problem has increased, and more and more tools have started to update their databases, these updates are often not consistent. Besides, very few of them provide access to previous data versions for reproducibility like WebGestalt and g:Profiler do.

### 1.3.4. Enrichment analysis visualisation

The number of different biological functions and pathways in annotation databases can be vast. Performing enrichment analysis against all these functions often results in long lists of significantly enriched terms even when multiple testing correction procedures are applied. Therefore, reading and interpreting such results can be a challenging task. Different visualisation approaches are essential to achieve a comprehensive overview from enrichment results quickly.

Enrichment results are a collection of functional terms with corresponding p-values. Hence, the most widely used visualisation method to show enriched terms is a bar plot where the column height corresponds to the p-value (often in -log10 scale) (Figure 5A). Also, different variations and other basic plot types are sometimes used. For example, using dots instead of the bars or showing the intersection between the input gene list and the function instead of the p-value. However, such plots do not show the full results but present only a selection of top functions.

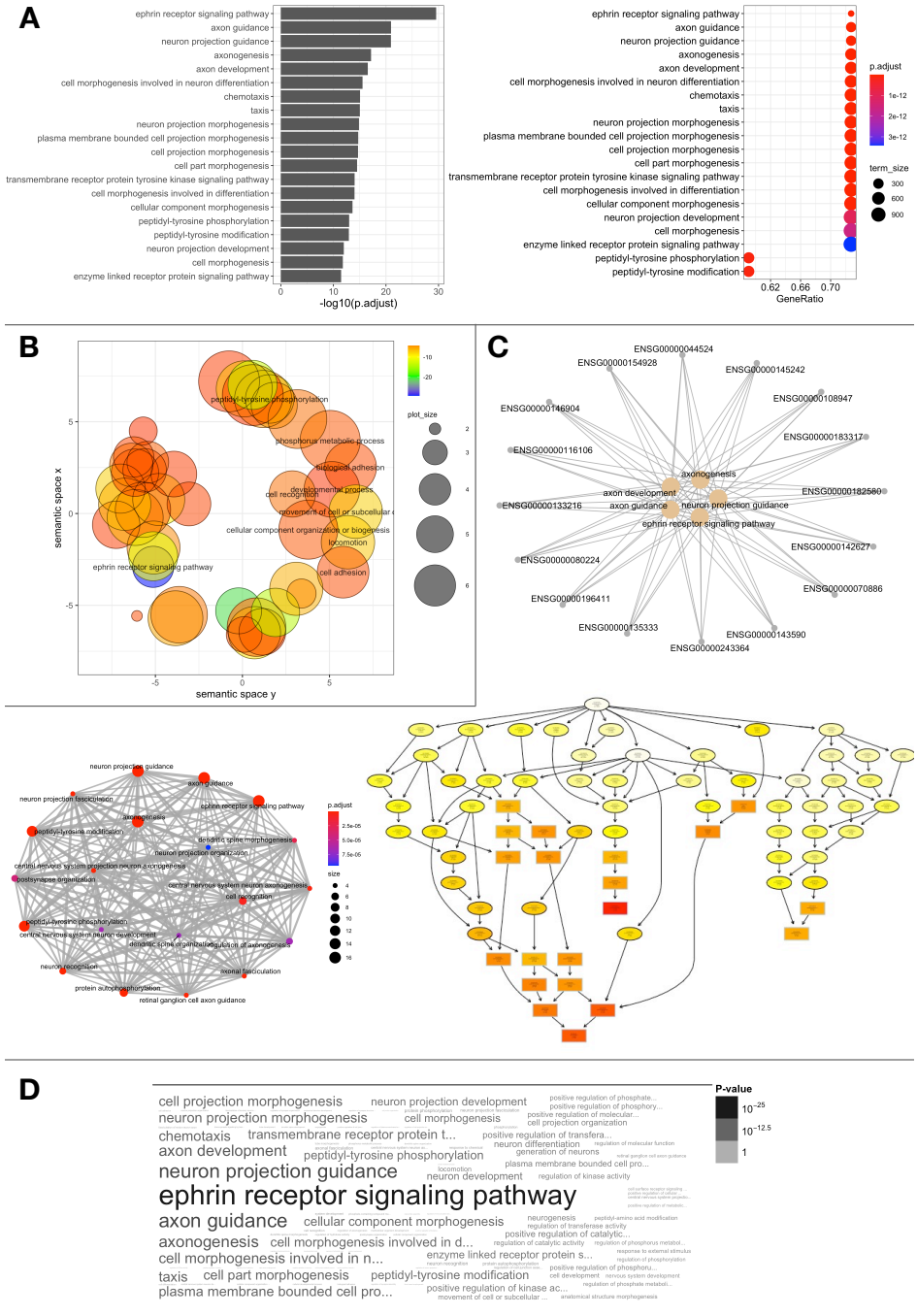



**Figure 5. Visualising enrichment results.** **A** Classical visualisation methods such as bar plot and dot plot for presenting enrichment results. **B** Semantic similarity-based scatterplot of enriched terms from REVIGO (Supek et al., 2011). **C** Different versions of networks and graphs highlight the relations between the enriched functions. Shown examples were created using the *topGO* and *enrichplot* packages (Alexa and Rahnenfuhrer, 2019; Yu, 2018). **D** Word cloud of enriched functions obtained from the *GOsummaries* package (Kolde and Vilo, 2015). The textual descriptions are size- and color-coded by the enrichment p-value.

More sophisticated approaches also include the relations between the functions, for example, by clustering the enriched functions based on some semantic similarity measure and showing these in a scatterplot (Supek et al., 2011) (Figure 5B). The semantic similarity can be based on the number of edges in the graph path between two terms, for example. The hierarchical structure of functional annotations, especially the ontological data sources, offers a natural application for network and graph-based visualisations (Figure 5C). Usually, each function is a node, and edges represent gene overlap between functions. Sometimes also the associated genes are included in the network. Ontology graphs are often composed to show how the significant GO terms are distributed over the GO graph (Eden et al., 2009; Supek and Škunca, 2017).

Word clouds are used to summarise the enriched terms with textual expressions where the most important keywords are highlighted with size and color (Figure 5D). For example, R package *GOsummaries* (Kolde and Vilo, 2015) and stand-alone application WocEA (Ning et al., 2018) are dedicated to compose word clouds of enrichment results.

To summarise, in this chapter we gave a brief overview of analysis and visualisation methods often applied to gene expression profiling data. In the following chapters, we elaborate on using these methods in the tools and studies underlying this thesis.

# CHAPTER 2

# GENE LIST FUNCTIONAL INTERPRETATION WITH G:PROFILER

Convenient tools that rely on most up-to-date data are essential for gene list functional interpretation. As we mentioned before, there is a wide range of diverse enrichment tools covering different data sources, species, and methods. In this chapter, we describe g:Profiler, a toolset developed in our research group at the University of Tartu.

The g:Profiler toolset (`https://biit.cs.ut.ee/gprofiler`) is widely used for handling gene lists in various ways. This toolset performs functional enrichment analysis and maps between identifier namespaces or orthologous organisms. There are four dedicated tools that enable these functionalities:

- g:GOSt for gene list functional enrichment analysis;
- g:Convert for mapping gene identifiers between numerous identifier namespaces;
- g:Orth for mapping genes of interest to homologous genes in another related organism;
- g:SNPense for mapping single nucleotide polymorphism (SNP) rs-identifiers to overlapping protein coding gene names, chromosomal coordinates and predicted variant effects.

The g:Profiler web tool was first developed in 2007 by Reimand and colleagues (Reimand et al., 2007), and has been under constant development ever since (Raudvere et al., 2019; Reimand et al., 2011, 2016).

## 2.1. Functional enrichment analysis in g:Profiler

Functional enrichment analysis for gene list interpretation is the most popular use case of g:Profiler toolset. For a list of user-supplied query genes, g:GOSt performs over-representation analysis using the hypergeometric test. The statistical test enables to infer shared functions in the input gene list by matching against previous knowledge from most common annotation databases such as the GO (The Gene Ontology Consortium, 2021), KEGG (Kanehisa et al., 2021) and Reactome (Jassal et al., 2020), to name a few. Since the input gene list is tested against thousands of functions, it is essential to consider the total number of tests to control the false-positive results. For this reason, g:GOSt by default applies a custom correction method that takes into account the set intersections between annotations defined as the g:SCS (Set Counts and Sizes) method in g:Profiler (Reimand et al., 2007). Nevertheless, users can also choose to apply more common methods like

Bonferroni correction or Benjamini–Hochberg False Discovery Rate (FDR) (Benjamini and Hochberg, 1995). Finally, statistically significant biological functions ($p\text{-}value_{adj} \leq 0.05$) together with corresponding adjusted p-values are reported to the user.

In addition to the most popular gene set and pathway databases, all available data sources in g:GOSt are constantly revised and updated. As of the latest update article in 2019 (Raudvere et al., 2019), Publication I in this thesis, g:GOSt includes various other sources like biological pathways from WikiPathways (Martens et al., 2021), regulatory targets from miRTarBase (Chou et al., 2018) and TRANSFAC (Wingender, 2008), protein databases Human Protein Atlas (Uhlén et al., 2015) and CORUM (Giurgiu et al., 2019), and human disease phenotype associations from Human Phenotype Ontology (Köhler et al., 2021). General principles in the default data source selection are that these should be well-curated collections that are regularly updated. Furthermore, g:Profiler enables to perform enrichment analysis with custom annotation data using the Gene Matrix Transposed (GMT) file format. A GMT file is a tab-separated text file where every row corresponds to a single function followed by the list of genes annotated with this function. This GMT file option becomes relevant when users want to analyse functional terms or organisms specific to their research topic that are not available from the default data sources.

Given the number of different annotation databases that are ever-evolving with new knowledge, the constant data changes pose a challenge of keeping the interpretation tools up to date while providing access to previous data versions for reproducibility. g:Profiler is a rare tool that frequently updates its sources and at the same time keeps previous versions accessible via dedicated archives (Wadi et al., 2016). Furthermore, publishing a series of update articles (Kolberg et al., 2020; Raudvere et al., 2019; Reimand et al., 2007, 2011, 2016) serves its purpose of informing g:Profiler's long-term users about the recent changes as well as making it discoverable for new users. Only in 2020, the users worldwide performed more than 10 million queries in g:Profiler. In addition, the primary g:Profiler publications have been cited by more than 3,000 papers in total. This number also includes the recent update article (Raudvere et al., 2019), Publication I, that has been cited by 774 papers within only two years after its publication (data retrieved from Google Scholar, 06.05.2021). Moreover, g:Profiler has been recognised by European Life Science Infrastructure ELIXIR as a Recommended Interoperability Resource to be of fundamental importance to the research infrastructure of the life sciences (`https://www.elixir-europe.org/platforms/interoperability/rirs`).

Relevant annotation sources and intuitive visualisations play a fundamental role in gene list interpretation and greatly impact the researcher's conclusions. We have developed several improvements in g:Profiler keeping these requirements in mind. The following sections discuss some of the recent updates in which I have had a significant role.

## 2.2. Taking gene list interpretation to the next level– g:Profiler web tool update (Publication I)

A number of new software development technologies have emerged since the first publication of g:Profiler in 2007. The wide availability of modern software and interfaces has made users more demanding, especially in terms of speed, appearance, and interaction. Therefore, any web tool, including g:Profiler, should keep up with the trends to retain and attract users. However, over time the initial codebase of g:Profiler became very complex and multi-layered, consisting of multiple intertwined programming languages. At the same time, the complexity and size of the underlying data in g:Profiler continued to grow. The existing solutions left us little room for adding new features and made it harder to maintain a fast and stable service. To scale up the underlying infrastructure and attain more flexibility for future changes, we introduced a complete technical rewrite of the g:Profiler toolset in Publication I. As a result, g:Profiler now has a faster and more customisable back-end and a more convenient and modern user interface that provides a responsive detailed results table and interactive visualisations (see Figure 6).

### 2.2.1. Manhattan plot for visualising enrichment results

Comprehensive visualisations are essential for identifying and presenting the key elements of the analysis. However, functional enrichment analysis often ends with long lists of enriched functions, making it challenging to visually summarise the results. As we mentioned earlier, the most common approach is to present a selected number of top functions on a bar plot. Several enrichment tools and packages provide this option, such as Enrichr (Kuleshov et al., 2016), Metascape (Zhou et al., 2019), and enrichplot (Yu, 2018). But this approach has its limitations. First, the selection of functions shown on a graph is often subjective or relies on arbitrary thresholds, and therefore does not represent the full information. Second, such a presentation does not convey the interrelations between the enriched functions. For example, in our analyses, we have seen that the GO enrichment results are often flooded with strongly related terms that mostly stem from different specificity levels of the ontology and are defined by a single group of genes. Therefore, due to the propagation of the GO annotations, selecting the top 10 terms for a bar plot might end up showing a single branch from the ontology tree representing essentially the same biological process and hiding other potentially relevant ones. As a solution, in Publication I, we introduced a novel Manhattan plot that compactly visualises all the enrichment results (Figure 6C).

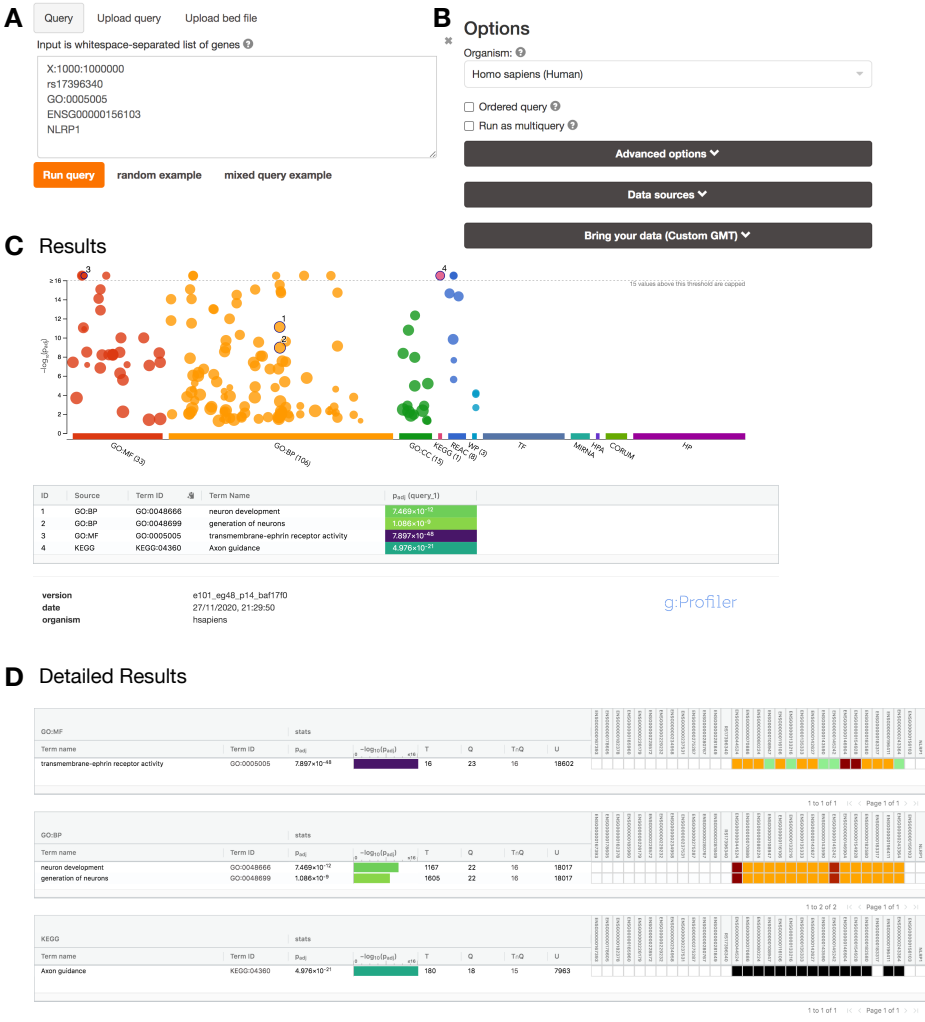
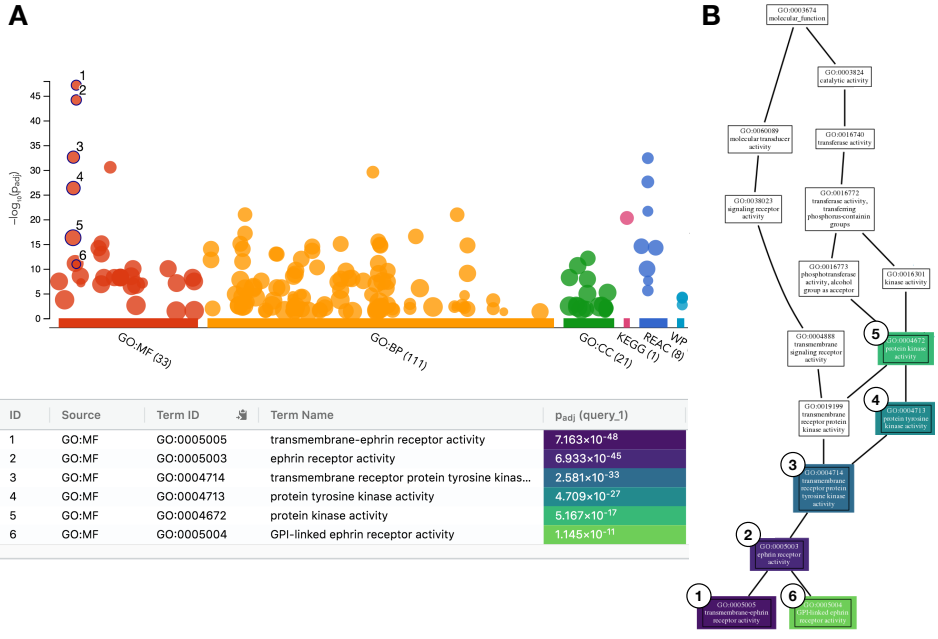



**Figure 6. Overview of g:Profiler enrichment analysis results. A-B** Modern interface of g:Profiler enables flexible input and provides various parameter options for customised analyses. **C** A novel Manhattan plot showing all the significantly enriched functions in a single view. Each circle represents a single term. The x-axis corresponds to a unique ID of the term and the y-axis shows the corresponding enrichment p-value in a -log10 scale. Clicking on a term circle will highlight it on the plot and a table with highlighted terms is created below the plot for more details. **D** A detailed results table with the enriched functions is available from a separate tab. The interactive and searchable table can be modified before exporting the results into a text file or graphical image. Selecting a term in the table will highlight it also on the Manhattan plot.

Although the Manhattan plot is well-known from the genome-wide association studies, to our best knowledge, this is the first time of using this plot for enrichment analysis. By now, other enrichment tools have started to provide such graphs as well (Clarke et al., 2021). The plot illustrates all the significantly en-

riched functions across user-selected data sources (Figure 6C). The x-axis shows the functions grouped by their source, and the y-axis shows the corresponding adjusted p-values in a negative log10 scale. Each circle represents a single enriched term. These circles are color-coded by data source and size-scaled according to the number of genes annotated to that term. The locations on the x-axis are fixed regardless of the user input gene list. Corresponding positions are assigned so that the functions from the same GO sub-branch are located close to each other forming peaks in the Manhattan plot because such functions share genes and therefore tend to be enriched together (see Figure 7). Fixed positions are essential for faster and more intuitive interpretation of the results for the returning users. In addition to the fixed positions on the x-axis, by default, the p-values on the y-axis are capped at $10^{-16}$ which fixates the scale of the y-axis to ensure fair visual comparison of the results from different queries. The user can turn off the value capping of the y-axis if needed.




| ID | Source | Term ID | Term Name | $p_{adj}$ (query_1) |
|---|---|---|---|---|
| 1 | GO:MF | GO:0005005 | transmembrane-ephrin receptor activity | $7.163\times10^{-48}$ |
| 2 | GO:MF | GO:0005003 | ephrin receptor activity | $6.933\times10^{-45}$ |
| 3 | GO:MF | GO:0004714 | transmembrane receptor protein tyrosine kinas... | $2.581\times10^{-33}$ |
| 4 | GO:MF | GO:0004713 | protein tyrosine kinase activity | $4.709\times10^{-27}$ |
| 5 | GO:MF | GO:0004672 | protein kinase activity | $5.167\times10^{-17}$ |
| 6 | GO:MF | GO:0005004 | GPI-linked ephrin receptor activity | $1.145\times10^{-11}$ |

**Figure 7. Illustration of the peaks in the Manhattan plot. A** The positions on the x-axis of the Manhattan plot are fixed so that the functions from the same GO sub-branch are closer to each other. The plot highlights six closely related functions from the GO molecular functions domain that form a peak in the Manhattan plot. **B** A Gene Ontology sub-graph highlighting the molecular functions selected in panel A using the corresponding color-codes and ID numbers. The relations between the functions are shown with the edges. The GO graph was composed using the AmiGO 2 web tool (version: 2.5.13) (Carbon et al., 2009).

### 2.2.2. Exploring and sharing the results

Static data visualisations, together with corresponding conclusions, are usually included in the scientific publication of any study that performs enrichment analysis. However, coming up with the story to tell, deciding on the specific results to highlight, and finally combining everything into a publishable figure can be a tedious task. To alleviate this process, g:Profiler delivers results through an interactive and responsive web design that enables the user to explore the data effortlessly and then compose a summary visualisation to convey the discoveries. For example, hovering over a circle in the Manhattan plot reveals a tooltip with the most relevant information about the term. The user can filter all the data in the detailed results tab to highlight or limit the results according to a specific keyword or term size. The user can also sort the columns in the results table and change their order by dragging the column headers. After identifying the core functions and pathways for interpretation, the user can highlight the most important terms on the Manhattan plot by clicking on the corresponding circles or selecting the functions from the results table. This action attributes the circles with identifiers and adds a descriptive table below the image showing the IDs, names, and p-values for the selected terms (Figure 6C). Finally, the user can download the composed combination of image and table as a high-quality image for publication. Similarly, the user can export the filtered tables from the detailed results tab as separate images (Figure 6D).

Besides attractive visualisations, another important aspect in publishing enrichment results is reproducibility. While the scientific journals require a full description of used methods and selected parameters, these requirements are often not fully met or are difficult to follow from the vague descriptions. At best, the authors attach the full results as Excel files that are hard to grasp by a human eye. Instead of providing long supplementary tables, we encourage to prepare and share a dedicated short-link that refers to the specific enrichment results in the g:Profiler web tool. Disclosing such a link in a publication enables both the reviewers and readers to explore the full results in the g:Profiler web page to evaluate the conclusions. This option is also useful for sharing the intermediate results easily with colleagues. Notably, the short-link contains information about the input and parameters used for this particular analysis. For this reason, g:Profiler stores the input gene list in a database in case of short-link generation. The links are permanent and available indefinitely. Otherwise, g:Profiler is committed to respects users' privacy and, by default, does not record input gene lists from the web form.

### 2.2.3. Analysing multiple gene lists

Enrichment analysis tools also start to focus on analysing multiple gene lists together. Namely, expression profiling experiments often measure gene expression in different conditions, states, or time points. Analysing these data usually re-

sults in multiple gene lists that can be associated with a specific condition. For example, the lists can be condition-specific clusters from a co-expression clustering or up- and down-regulated genes from differential gene expression analysis. Functional enrichment analysis is then performed to describe and compare the condition-specific gene lists at the level of biological processes. However, most of the available tools perform enrichment analysis for a single gene list at a time, making it difficult to combine and compare the results. In g:Profiler, the input can consist of multiple gene lists separated by a '>' symbol in front of every list. As a result, a separate Manhattan plot for each input list is presented to facilitate comparison (see Figure 8). The fixed positions on the x-axis simplify spotting the differences in the enriched functions across several queries. Capturing p-values to a fixed level serves the same purpose. In addition, the structure of the detailed results table now focuses on comparing the enrichments across the queries. That is, for each term that is significant for at least one of the input lists, the respective p-values for every query are shown side by side. The interactive functionality remains the same as in the case of a single query. Furthermore, hovering over a term in one Manhattan plot highlights the same term on other Manhattan plots. The adjusted p-values from other queries are then indicated in red next to the y-axis for easier comparison. Just as with a single query, the user can combine and export a suitable visualisation and generate a short-link to share the results of multiple queries.

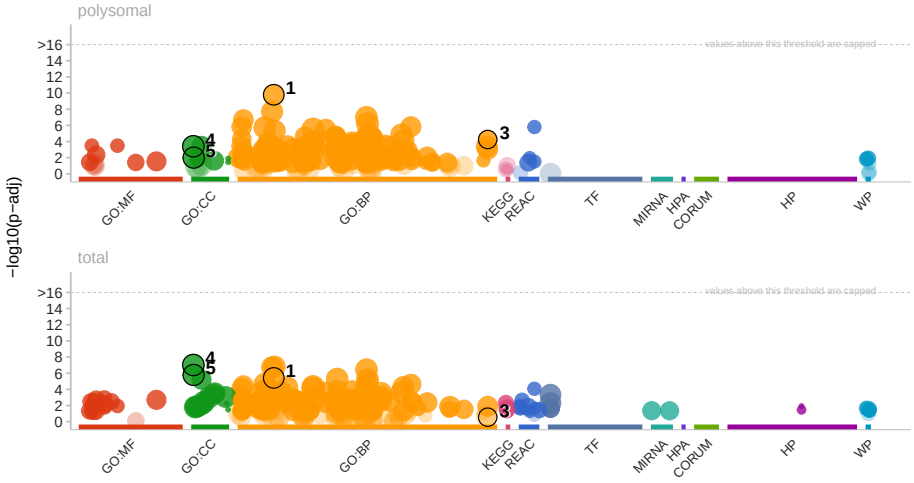
## 2.3. Automated enrichment analysis with gprofiler2 R package (Publication II)

Web tools are convenient for obtaining results quickly and with little effort. These tools are often irreplaceable for biologists who do not have extensive programming skills and do not have bioinformaticians in their lab to perform the analyses. However, with the abundance of data and analytical resources, the analysis workflows inevitably get longer and more complex, including numerous tools and programming languages (Cohen-Boulakia et al., 2017; Kanwal et al., 2017; Leipzig, 2017). Such workflows are easier to maintain and reuse if they consist of executable code snippets that are flexible and interoperable. These qualities are difficult to achieve by using various graphical interface-based tools.

R is a popular programming language for data analysis and graphics where different units of code, data, documentation, and tests are bundled together into packages that are easy to share with others (R Core Team, 2019). Numerous such packages are developed explicitly for retrieving and analysing different biological data (Amezquita et al., 2020; Drost and Paszkowski, 2017; Huber et al., 2015; ImmunoMind Team, 2019). One can easily conduct an end to end analysis of gene expression data in R starting from data retrieval and preprocessing steps to co-expression analysis and interpretation (Klaus and Reisenauer, 2016; Love, 2019; Lun et al., 2016). Most useful packages are available from one of the ded-

icated R package repositories, Bioconductor (Gentleman et al., 2004) or CRAN. These repositories have a rigorous review procedure making the packages more standardised and reliable.

The updated technical infrastructure of g:Profiler, introduced in Publication I, led to implementing a new compatible R package *gprofiler2* to provide programmatic access to g:Profiler computations and databases (Publication II). The essence of the package is simple; it wraps requests to the g:Profiler web server using corresponding application programming interface (API) endpoints. A significant added value lies in the accompanying graphics (see Figure 8) and enhanced functionality for seamless connection with the web tool. For example, *gprofiler2* enables to programmatically obtain web links to the g:Profiler results page for easy sharing with colleagues. The package is freely available from the CRAN repository.




| id | source | term_id | term_name | term_size | p_value polysomal | p_value total |
|---|---|---|---|---|---|---|
| 1 | GO:BP | GO:0009888 | tissue development | 2135 | 1.6e−10 | 3.8e−06 |
| 3 | GO:BP | GO:2000146 | negative regulation of cell motility | 369 | 5.6e−05 | 2.8e−01 |
| 4 | GO:CC | GO:0005576 | extracellular region | 4588 | 3.9e−04 | 9.6e−08 |
| 5 | GO:CC | GO:0005615 | extracellular space | 3551 | 9.7e−03 | 1.6e−06 |

g:Profiler (biit.cs.ut.ee/gprofiler)

**Figure 8. Example of functional enrichment results from *gprofiler2*.** This figure was generated with *gprofiler2* (v. 0.2.1) to reproduce Figure 2 in Publication I that was used to dissect a use case. The example uses data published by (Robert et al., 2018). Shortlink to these results in g:Profiler web interface is `https://biit.cs.ut.ee/gplink/l/_ROOQSudRO` (obtained with *gprofiler2*; g:Profiler v. e101_eg48_p14_baf17f0).

### 2.3.1. *gprofiler2* as an upgrade from previous package *gProfileR*

However, as the name *gprofiler2* suggests, this is not the first R package accompanying the g:Profiler web tool. There already was a popular package called *gPro-*

*fileR* that enabled to fetch results from the g:Profiler web server, but it was not compatible with the comprehensive technological update. We considered upgrading the *gProfileR* package. Still, we foresaw that it could cause some issues for the existing analysis pipelines due to the changes in query parameters and output structure. Instead, we decided to implement a new package and gradually redirect old users to it by including corresponding deprecation warning to the *gProfileR* package. The monthly download statistics from the CRAN repository indicates that the transition between the packages has been successful (Figure 9).

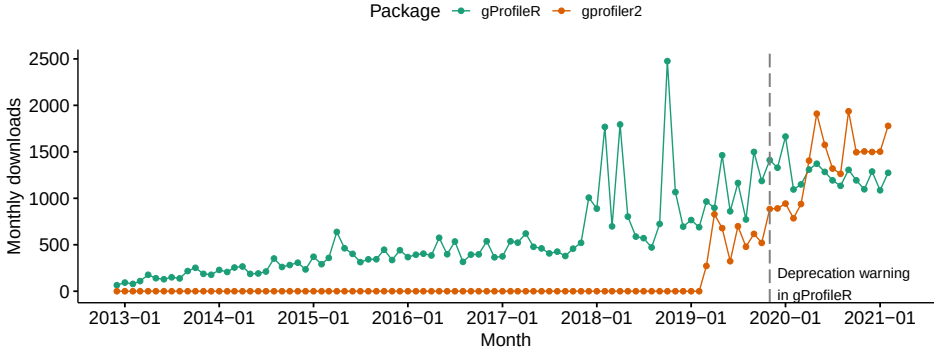



**Figure 9. Deprecating previous R package *gProfileR*.** Comparison of monthly CRAN downloads of *gProfileR* and *gprofiler2* packages from December 2012 to February 2021. The monthly downloads of *gprofiler2* (shown in orange) started to exceed the *gProfileR* downloads (shown in green) shortly after including a deprecation warning to the *gProfileR* package (shown with a vertical dashed line) and publishing the manuscript of Publication II.

Furthermore, implementing a new package enables R users to access the latest features and data in the updated g:Profiler while keeping the pre-update data versions available with the *gProfileR* package for reproducing previous results. Another important aspect is that starting with a new package provided the flexibility to implement a much broader functionality than in the initial package (see Table 2 for comparison). We also extended the output structure to provide a list of relevant metadata along with the enriched functions and related statistics to ensure reproducibility and transparency of enrichment analysis from R.

**Table 2.** Overview of feature updates of *gprofiler2* in comparison to *gProfileR*.

| | Feature | gProfileR (2012) | gprofiler2 (2020) |
|---|---|---|---|
| **Tools** | g:Orth | *gprofiler()* | *gost()* |
| | g:Convert | *gconvert()* | *gconvert()* |
| | g:Orth | *gorth()* | *gorth()* |
| | g:SNPense | - | *gsnpense()* |
| **Data** | Data versions | Up to version e94_eg41_p11 | Starting from g:Profiler version e94_eg41_p11 and newer |
| | Custom annotation sources | - | Uploading custom GMT file with *upload_GMT_file()* |
| | | - | Using previously uploaded data source by setting a key value to the '*organism*' variable in *gost()* |
| | Metadata | - | Returned with the results in the 'meta' object |
| **Presentation** | Visualisations | - | Interactive Manhattan plot |
| | | PNG image of results table obtained from the API | Results table for selected functions |
| | | - | Combined image of Manhattan plot with highlighted terms and corresponding table |
| | | - | Customisable Manhattan plot as a *ggplot2* object |
| | Shareability | - | Short link to web tool with parameter '*as_short_link = TRUE*' |

The most significant improvement was displaying an interactive Manhattan plot of the enrichment results that can be followed by combining and exporting a publication-ready image. While Shiny is the most popular package for building interactive web apps in R, it was not suitable for our use-case because we already have a flexible and interactive web interface for this purpose. However, we wanted to enable R users to mimic the functionality of our web tool programmatically. For this reason, we implemented a series of visualisation functions corresponding to the typical path a user follows through our website interface. First, provided with the enrichment results from the function `gost()`, the function `gostplot()` uses Plotly to return an interactive graph that enables to explore the results with the help of tooltips. Although the graph has the term pinning functionality, it will not automatically generate a publishable image in a way the web tool does. Instead, the user can insert the interesting terms to highlight into a separate function called `publish_gostplot()` that combines the high-quality image. The described functionality also extends to multiple queries (see Figure 8 for an example).

Sometimes it may be beneficial to identify missing features or shortcomings of a package by taking on the role of a user. We encountered a common use-case of *gprofiler2* while conducting the analyses presented in Publication IV. Namely, we detected hundreds of gene clusters using five different clustering methods. We aggregated the clusters into different groups based on specific shared properties and performed functional enrichment analysis of these cluster groups for combined interpretation. Manually copying these lists to the g:Profiler web tool was not feasible, and therefore we used *gprofiler2* that enabled us to perform the enrichment analyses automatically. However, such a high number of queries introduced the

challenge of browsing and summarising the long data tables of all the enriched functions. At the same time, the immediate results needed to be shared with colleagues to discuss potential conclusions. Reporting a vast amount of tables would have made the interpretation insufferable. These issues initiated the development of a short-link generation function in the *gprofiler2* R package. As a result, we had a functionality that conveniently sent numerous multi-queries to g:Profiler and returned the corresponding web links for sharing with the co-authors. In addition, we used *gprofiler2* to prepare images of enrichment results for the publication. In doing so, we identified and implemented some minor design changes that increase the readability of the tables in print. We dissect the particular results of Publication IV in detail in Chapter 4.

### 2.3.2. R packages accompanying web tools

There are other similar solutions where an R package is developed to interface with an enrichment web tool, mostly implemented by the community, e.g., *enrichR* (Jawaid, 2019), *RDAVIDWebService* (Fresno and Fernández, 2013), and *PANTHER.db* (Muller, 2019), and very few by the corresponding core development teams, e.g., WebGestalt (Liao et al., 2019). A high degree of community activity indicates that such R packages are sought-after. Furthermore, g:Profiler's initial package *gProfileR* already was very popular among users. Our team anticipated the continuing demand and developed *gprofiler2* in-house. Besides, in-house development ensures coherence between different g:Profiler interfaces. For example, in addition to using all the existing data sources such as the GO, KEGG, Reactome, etc., the user can upload a custom data source GMT file with the dedicated function in R and analyse the same data from the web page, and vice versa. To obtain full functionality, implementing such features requires in-depth knowledge about the infrastructure that only the core development team can have.

To my best knowledge, this is the first R package that is in sync with the corresponding web tool, provides similar graphics, and enables to display and share the results from R to the web tool. This kind of compatibility facilitates the communication between bioinformaticians and other researchers involved in the respective project. By developing the *gprofiler2* R package, we have created a resource for gene list handling that can be easily incorporated into automated analysis pipelines in R. On average, around 155,000 queries per month are performed using the *gprofiler2* package. Furthermore, the package is already included in various other packages and analysis pipelines (Lemoine, 2020; Secherre, 2020; Uyar et al., 2017), and being taught in different computational biology workshops.

## 2.4. Contribution

Providing a flexible and user-friendly application software, such as g:Profiler, that serves various life science communities takes a team effort. I have been a member of the g:Profiler team since 2015. I have developed g:Profiler from different

aspects during this time, starting from interface design to prioritising future development activities. I have led the prototyping of various functionalities, validating methods, developing the R package, and providing user support. I also documented the updated toolset, prepared several example use cases and files. In addition, I have carried out workshops on enrichment analysis and g:Profiler for interpreting gene lists as part of an annual course organised by the EMBL's European Bioinformatics Institute.

As for the articles, I had a major role in writing the manuscript of Publication I. In addition, I conducted the use case and prepared all the figures for this article. In Publication II, I had a lead role in implementing and publishing the accompanying R package *gprofiler2*. I designed, developed, and documented the package. Finally, I composed and wrote the article.

# CHAPTER 3

# GENE CO-EXPRESSION ANALYSIS TOOL FUNCEXPLORER

In the previous chapter, we addressed the importance of easy-to-use and versatile functional enrichment analysis tools for biological interpretation in the example of the updated g:Profiler toolset (Publication I) and its accompanying R package *gprofiler2* (Publication II). However, the tools strongly depend on a meaningful list of input genes. As described in Chapter 1, the genes to study are most often derived from gene expression analyses such as differential expression analysis or clustering. Here, we take a step back from functional enrichment analysis and review a standard gene expression analysis workflow that uses clustering. We describe a method that combines hierarchical clustering with functional information to reveal biologically relevant gene groups from the data. Finally, we present a web tool funcExplorer that automatically performs a revised version of this method and shows the results in a visually compact and interactive format (Publication III).

## 3.1. Gene co-expression clustering for interpretation

High-throughput experiments have opened avenues to simultaneously investigate the expression and function of many genes across several conditions or time points. Often, this involves dissecting the gene expression data into smaller groups of genes that respond similarly to a specific experimental condition and are therefore considered co-regulated and functionally related. Gene expression clustering is a routinely used approach for this task. Since clustering algorithms always find clusters, even if there are no biologically relevant clusters in the data, the gene clusters are then validated and described using functional enrichment analysis. Finally, the results are gathered, summarised, and represented visually to convey the biological interpretation.

However, this typical analysis pipeline introduces several pitfalls. First, most clustering algorithms rely on a set of parameters such as the number of clusters to detect in the K-means algorithm or the height to cut the dendrogram from hierarchical clustering. Choosing the parameter values can be highly subjective and biased towards the gut feeling of the researcher (Quackenbush, 2001; Slonim, 2002). Such selection, in turn, limits the reproducibility of the results. Next, since clustering algorithms usually identify many clusters, performing functional enrichment analysis for each cluster can be a very time-consuming and tedious task. On the other hand, selecting only a subset of clusters to analyse can overlook a potential biological discovery. Finally, the full process from clustering to

interpretation and visualisation usually involves several tools to be applied successively to obtain the desired outcome. As there is a lack of ready-made solutions, conducting such a pipeline can be challenging for a biologist without extensive programming skills. Furthermore, gathering the results and representing them visually to convey the information at a glance is a complicated task.

## 3.2. Predecessor – VisHiC

One approach to tackle the challenges mentioned above was introduced in our research group by Krushevskaya *et al.*, who proposed an enrichment-driven pruning method for hierarchical clustering of gene expression data (Krushevskaya et al., 2009). The core idea was to identify gene clusters from a dendrogram based on significantly enriched functions instead of the standard fixed height cut-off. Furthermore, they developed a web tool called VisHiC (Visualisation of Hierarchical Clustering) that performed the analysis and visualised the results. The outcome was a combination of a heatmap and a dendrogram that highlighted the enriched clusters and concealed biologically less relevant gene profiles to provide a compact overview of the gene expression data (see Figure 10).

More specifically, the proposed method starts by clustering the gene expression matrix using an approximate agglomerative hierarchical clustering algorithm HappieClust (Kull and Vilo, 2008). Instead of comparing each pair of genes, HappieClust runs hierarchical clustering on a carefully chosen subset of pairwise distances. This approach enables to reduce the computation time while still resulting in biologically comparable results to full standard hierarchical clustering (Kull and Vilo, 2008). Next, all the detected gene clusters with sizes ranging from 5 to 1,000 genes are functionally characterised using the g:Profiler toolset (Reimand et al., 2007). That is, enrichment analysis is performed for all the sub-branches of the corresponding size in the dendrogram. The enrichment p-values are then used to calculate a pre-specified score that evaluates the total enrichment in each gene cluster. There are two different strategies to choose from: the size-weighted annotation score, which summarises the negative logarithms of enrichment p-values, or the best annotation score, which assigns the maximum negative log p-value as a score to each cluster.

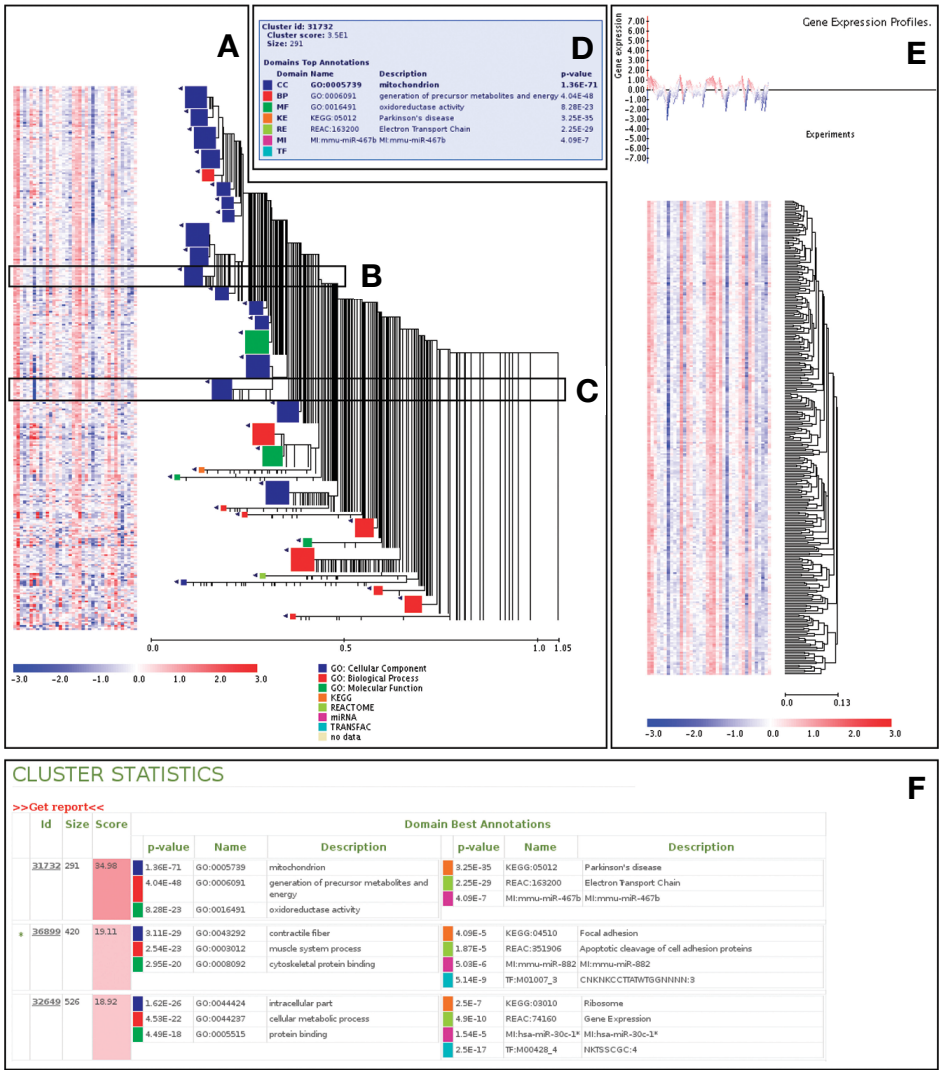



**Figure 10. Overview of VisHiC output obtained from the original paper (Krushevskaya et al., 2009). A** Gene expression matrix together with dendrogram highlighting significant gene clusters such as clusters **B** (ID:31732) and **C** (ID:36899), for example. **D** Tooltip with top functional enrichment results that appeared on mouseover of the dendrogram. **E** Detailed view page of the selected gene cluster with heatmap, dendrogram and lineplot. **F** Summary table with functional enrichment results per gene cluster.

After calculating the enrichment scores, an enrichment-driven pruning method searches for gene clusters with the highest enrichment scores. The algorithm starts from the cluster with the best score, marks it as a significant cluster, and excludes all its child and parent clusters from the further searches. This process is continued in decreasing order of the scores until there is no branch left with a non-zero enrichment score. Finally, the output shows a dendrogram together with the corresponding heatmap. Color-coded rectangles on top of the dendrogram highlight the

sub-branches with the most significant enrichment scores (see Figure 10A). Such a visualisation enables to emphasise different gene groups in the experiment that are both co-expressed and functionally related. Importantly, unlike with the standard fixed-height approach, the clusters in VisHiC are often detected at varying heights of the dendrogram. At the same time, the output hides poorly annotated expression profiles by clipping out the highest sub-branches without any enrichment results. The latter significantly reduces the size of the output compared to when fitting the whole gene expression matrix into an image.

## 3.3. Large-scale exploratory interpretation tool funcExplorer (Publication III)

When I joined the research group, the lead developer of VisHiC had left the group, as is often the case with academic software. The web tool itself was working but was out of date, mostly because of the discontinued development. Besides, the technologies, methods, and standards in software development had improved. However, functional characterisation of patterns emerging from the experimental gene expression data remained relevant. Furthermore, exploratory tools became more appealing by providing data-driven and hypothesis-free analyses with the potential to reveal novel characteristics and trends. While researchers were addressing increasingly complex data and problems, there were no other similar web tools available that would perform a full gene expression analysis pipeline from clustering to enrichment analysis all at once. Most similar gene clustering tools had key limitations (see Figure S1 in Publication III for an overview). To keep up with the advances, especially in genome-wide profiling technologies and software development, we implemented a new web tool called funcExplorer (`https://biit.cs.ut.ee/funcexplorer`). Taking inspiration from VisHiC, our main aim was to provide an improved web tool that enables users to perform hypothesis-free analysis both at the global and local levels of the data.

### 3.3.1. Overview of funcExplorer workflow

Similar to VisHiC, the analysis pipeline in funcExplorer consists of four main parts. First, the user uploads their gene expression matrix and chooses suitable pre-processing parameters such as standardisation, missing value imputation, etc. In principle, the analysis is not restricted to gene expression values only. Any numerical activity measure of genes, proteins, or probes across a number of conditions is admissible. Second, back-end scripts process the received data, perform hierarchical clustering followed by functional enrichment analysis with g:Profiler, and store the results in a database. We keep the results for 365 days by default, but on request we can keep the data for an extended period of time or completely remove the analysis traces. Third, the user gets access to the analysis results and can choose the enrichment score strategy for detecting informative gene clusters

based on the functional annotations. Finally, funcExplorer composes the clustering results into a compressed view with interactive features for exploration (see Figure 11).

While biologists usually look into few hundred specific genes related to their hypothesis and leave everything else in the data unnoticed, the comprehensive output of funcExplorer describes the whole dataset and highlights the unique functional characterisations. Therefore, new knowledge and hypotheses can arise that otherwise would be disregarded. Moreover, contrary to classical clustering techniques that find clusters under any conditions, funcExplorer will not determine gene groups if there is no biological relevance.

Although the core idea of enrichment based cut-off detection from a hierarchical dendrogram remained, the following features were implemented in funcExplorer:

- Gene expression matrix upload form enables to impute missing values, normalise RNA-seq data, perform data transformations, include sample annotations, and choose a background set for the enrichment analysis.
- Analysis pipeline applies clustering algorithm called Hybrid hierarchical clustering (Tanaseichuk et al., 2015), and g:Profiler for functional enrichment analysis.
- Selection of enrichment scores for cluster detection: first annotation strategy, F1 score, and best annotation strategy.
- Option to choose specific annotation sources to use in the analyses.
- Interactive and searchable global overview of the clustering results (Figure 11A-D).
- Detailed cluster descriptions showing eigengene profiles and functional enrichment word clouds (Figure 11E-F).
- Exporting and sharing the clustering results as PNG images, text files, or persistent web links.

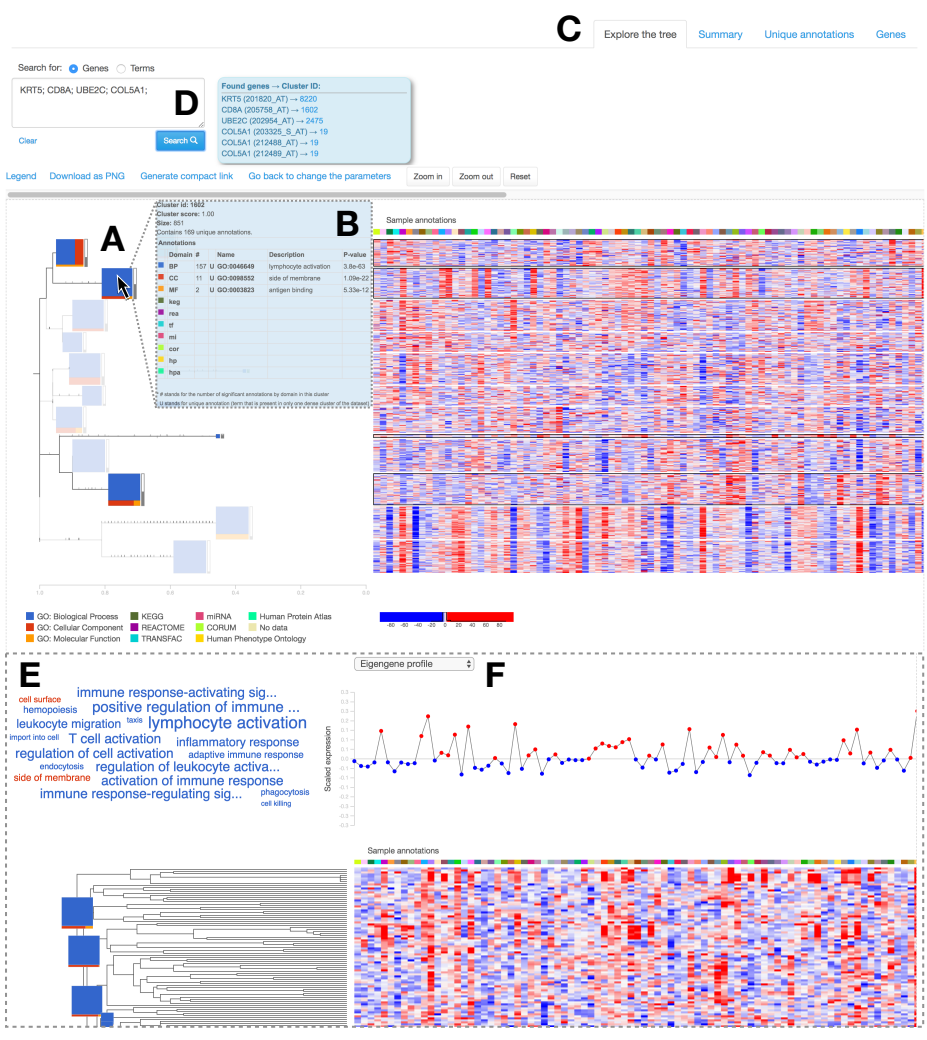



**Figure 11. Overview of funcExplorer output.** The shown example is based on a breast cancer dataset with the GEO accession no. GSE11121 first analysed by (Schmidt et al., 2008). **A** The main output shows gene expression matrix together with dendrogram highlighting significant gene clusters such as the one with ID 1602. The clusters are indicated by rectangles that are size-scaled and color-coded according to significant functions from enrichment analysis. **B** A tooltip with top functional enrichment results appears on mouseover of the cluster rectangles. **C** More detailed information tables are shown in tabs 'Summary', 'Unique annotations' and 'Genes'. **D** Search field allows searching for interesting genes or functions. The results are reported next to the text field, and corresponding modules are highlighted in the main view by dimming out unrelated clusters. **E** Clicking on a specific cluster redirects the user to a detailed view page of the selected cluster showing related functional topics in a word cloud, clustering dendrogram, and gene expression heatmap. **F** Eigengene profile is shown on top of the heatmap to summarize the cluster's behavior across samples. The results used for this figure are available for more detailed exploration at `https://biit.cs.ut.ee/funcexplorer/link/22ebb`. This figure is adapted from the Figure 1 in Publication III.

### 3.3.2. Hierarchical clustering

The size of gene expression data poses several computational challenges for hierarchical clustering, mostly in terms of resource usage and speed. The algorithms used in a web tool should be faster than usual to cluster the data and present the results. Approximate clustering algorithms are sometimes used for this purpose, such as the HappieClust algorithm (Kull and Vilo, 2008) used in VisHiC. For funcExplorer, we decided to use a different approximate agglomerative hierarchical clustering algorithm. Instead of HappieClust, we switched to a Hybrid hierarchical clustering algorithm (Tanaseichuk et al., 2015) designed to handle large datasets. The hybrid approach first partitions the data using the K-means clustering algorithm to avoid calculating all the pairwise distances. Next, the algorithm applies exact average-linkage hierarchical clustering to the $K$ clusters and the objects within them. As a result, the algorithm assembles the smaller subtrees into a complete hierarchical tree that is very similar to what exact algorithms achieve. The difference is that the method is applicable to much larger datasets such as those encountered in functional genomics. Therefore, the hybrid hierarchical clustering seemed promising for applying in funcExplorer. Furthermore, method demonstrated the ability to cluster large gene expression matrices such as the tissue-specific RNA-seq data from the Genotype-Tissue Expression (GTEx) project (Lonsdale et al., 2013; The GTEx Consortium et al., 2015) consisting of activity levels of 42,548 genes (including pseudogenes and long non-coding RNAs) measured across 53 tissues.

### 3.3.3. Defining enrichment scores

The hybrid hierarchical clustering is followed by functional enrichment analysis performed for each node in the dendrogram with a size between 5 to 1,000 genes. Next, the user can choose from three different approaches to assign an enrichment-based score to each node. This predefined score is then maximised within every dendrogram branch leading to various cutting heights instead of cutting the dendrogram at a fixed level of similarity. We provide three different scoring strategies instead of providing a single list of clusters because there is no one-size-fits-all solution to gene expression clustering (Way et al., 2020). After all, there is rarely a gold standard indicating which genes should cluster together and which genes should be in different clusters. Depending on the research question, the 'good' gene clusters can come from different levels of biological granularity. To deeply understand the properties of different enrichment scores, we reviewed the strategies in VisHiC and proposed new measures for funcExplorer. Namely, the new strategies are the *first annotation strategy* and the *F1 score strategy*.

The *first annotation strategy* traverses the dendrogram in level order starting from the root and retrieves the first and largest gene clusters of every branch that has any significant enrichment. Therefore, the approach relies on the clustering hierarchy rather than calculating enrichment scores. Putting this into the con-

text of the Gene Ontology, the *first annotation strategy* delivers gene groups described by more general functions and processes in the GO (Figure 12). This result is due to the nested annotation structure of the Gene Ontology tree that resembles a clustering dendrogram where the lower levels consist of small and specific gene groups, and higher branches include all the genes from the descendant sub-branches. When we cut the dendrogram from close to the root, the functional enrichment analysis captures the higher levels of the GO tree. Thus, gene clusters from the *first annotation strategy* represent a broad overview of functional features in the data.

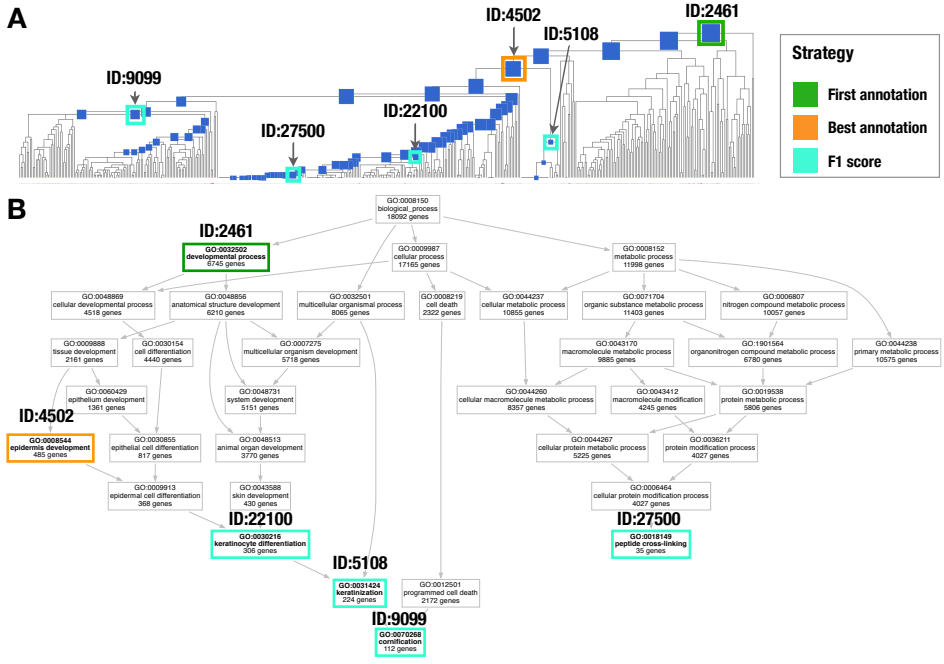



**Figure 12. Characterisation of cutting strategies.** The characterisation is based on the skin-specific clusters detected from the GTEx data (Lonsdale et al., 2013; The GTEx Consortium et al., 2015) using only the enriched functions from the GO biological processes (see `https://biit.cs.ut.ee/funcexplorer/link/24867` for all the clusters detected by the first annotation strategy). **A** The skin-specific branch from the GTEx dendrogram where the clusters detected by different cutting strategies are highlighted with colors and cluster IDs. Dark blue rectangles indicate that corresponding gene groups are enriched with at least one function from the biological process domain. **B** The corresponding sub-branch from the Gene Ontology biological process tree where each cutting strategy's leading functions are highlighted with colors and cluster IDs.

We observed similar behavior in the case of the *best annotation strategy* taken from VisHiC. Namely, the approach evaluates enrichment using the maximum value of negative log p-values across all the significant terms enriched in the cluster. However, due to the discrete nature of the hypergeometric distribution used for the enrichment analysis, the size of a gene cluster impacts the absolute enrich-

ment p-values. That is, a small gene cluster cannot achieve as high of a score as a much larger cluster can, even if it entirely coincides with a specific functional class. Consequently, the *best annotation strategy* score prefers relatively large clusters usually related to more general functional terms (Figure 12).

To complement these two strategies, we proposed an F1 score based enrichment measure that calculates the F1 scores of a cluster with respect to each of its significantly enriched functions. F1 score is the harmonic mean of the precision and recall with values ranging from 0 to 1 (perfect precision and recall) (Fung et al., 2003; Van Rijsbergen, 1979). Precision is the fraction of correct predictions out of the total number of predictions, and recall is the fraction of correct predictions from the total number of true positives. Although the F1 score is generally used to assess the quality of the overall clustering result, we adapted the measure to our setting to achieve enrichment scores from a fixed range of values (from 0 to 1). In our case, precision is the fraction of annotated genes among the genes in the cluster. Recall is the fraction of annotated cluster genes among the total genes annotated to the function. Therefore, we calculate the F1 score for a cluster with respect to a fixed function from the values given in Table 1:

$$F1 = 2 \cdot \frac{\frac{k}{n} \cdot \frac{k}{M}}{\left(\frac{k}{n} + \frac{k}{M}\right)} \tag{3.1}$$

For every node in the dendrogram, the *F1 score strategy* uses the maximum value of F1 scores across all the significantly enriched functions in a cluster as the cluster's enrichment score. Intuitively, this approach looks for clusters that are 'complete', i.e., have most genes from a functional category in a cluster, and a large proportion of cluster genes belong to that category. The perfect F1 score of 1 is achievable if a cluster coincides with a functional term. Our experiments revealed that this method results in small clusters characterised by specific functions from lower GO hierarchy levels (Figure 12). This was an expected result as most terms in annotation databases are relatively small, and therefore the occasion of finding a complete overlap is more probable for smaller clusters.

In summary, funcExplorer provides three different cutting strategies that present complementary biological representations (see Figure 12). Namely, the *first annotation strategy* leads to large general clusters, the *best annotation strategy* also results in larger clusters but with the most significant enrichments. The *F1 score strategy* is for cases where small and specific clusters are of interest. For example, when the researcher is looking for strong signals such as tumor types, the *best annotation strategy* is a suitable approach to apply. Alternatively, when searching for more subtle signals such as the activity of a specific pathway, the *F1 score strategy* is more appropriate. In the case of extensive exploratory research, it might be beneficial to apply all the strategies and integrate the results to enhance capturing a wide range of biological representations to study further. If the goal is to achieve visual compression of the gene expression matrix, then the *first annotation strat-*

*egy* is the most suitable approach. Nevertheless, the final choice of the method depends on the specific experiment and research question.

### 3.3.4. Presenting clustering results

After detecting all the biologically relevant gene groups from the experimental data, the biologist faces the challenge of summarising and interpreting the study. The clustering results are usually gathered and represented visually to convey the obtained information. This is another step in the common analysis pipeline that we have automated in funcExplorer. Namely, after funcExplorer has identified the enriched clusters based on the user-selected cutting strategy, the results are visualised in a compressed dendrogram-heatmap plot highlighting the most significant gene groups in the data (see Figure 11). VisHiC also presented a similar output. The main differences are that in funcExplorer, the clusters are highlighted with color-coded rectangles representing all the annotation sources of significant functions in the cluster (Figure 11A). In addition, the proportion of annotated genes in a cluster is shown next to each cluster with a gray bar. For example, this can be used as an indicator for the clusters where the guilt by association principle might be applicable. Color-coding is also used in annotation tracks above the heatmap to characterise the samples. For example, colored annotations enable the identification of the conditions where the cluster genes are differentially expressed. Besides the visualisations, a summary table is presented in funcExplorer to provide concise enrichment statistics of every cluster (Figure 11C). In addition to significant functions, the table includes word clouds representing a snapshot of the clusters' functional topics (see Figure 11E as an example) and eigengene profiles that characterise the expression levels prevalent in the cluster genes (see Figure 11F as an example). Furthermore, funcExplorer identifies the unique annotations in every cluster and presents the results in a separate 'Unique annotations' tab (Figure 11C) to indicate the functional difference between different gene groups in the data.

Although the visualisations in funcExplorer already present a lot of information compressed into a compact view, we implemented the web interface so that all the visualisation components and tables are highly interactive to enable convenient data exploration and retrieval. This means that different page elements reveal informative tooltips on mouseover (Figure 11B), and the summary tables are sortable and filterable. Another great extension for exploration is the searching functionality that enables to search for both specific genes and functional terms (Figure 11D). As a result, funcExplorer highlights corresponding clusters in the output and next to the search box. For example, the user can look up all the gene clusters enriched with immune response-related functions by typing in the keywords. To provide more in-depth exploration, we added the possibility to recursively zoom-in on the data up to the gene level through clickable elements on the dendrogram and summary tables. Zooming in to a single cluster view, a word cloud of enriched functions (Figure 11E) and the eigengene profile are shown

(Figure 11F) in addition to the dendrogram-heatmap view. This output is also interactive and searchable.

While the interactive exploration of the whole experiment is useful for interpretation, the results often need to be shared with colleagues. There are also potential follow-up analyses to perform, and finally, the results should accompany a scientific publication. When developing funcExplorer, we paid particular attention to these aspects. As a result, the web interface of funcExplorer incorporates extensive shareability and data extraction functionalities to facilitate these tasks. First, the user can generate a short-link to share the obtained results with peers or provide it as a reference in a publication. This functionality simplifies the reproducibility because the short-link embeds all the parameters and data required to reach the same results. Second, the user can easily export all the visualisations and summary tables as PNG images. Finally, the user can export summary tables, unique annotation information, cluster gene lists, and eigengene profiles into comma-separated text files (CSV). These data are useful for further analyses. For example, a specific use-case for exporting cluster eigengene profiles emerged in Publication IV.

## 3.4. Contribution

The main goal of funcExplorer is to empower biologists to perform standard analysis on their data and retrieve an initial overview of the experiment's functional essence without any specialised programming knowledge. Thereby, the tool enables conducting discovery-driven research.

My contribution in the development of funcExplorer is ubiquitous and covers most work with reviewing and extending the methods, implementing the analysis pipeline and the web interface to carry it out. The latter includes all the aspects, from the technological choices and back end infrastructure to the user interface design and documentation. Finally, I led writing the manuscript, which includes describing the properties of different parameters, conducting the case study, and comparisons with other methods.

# CHAPTER 4

# COMBINING CO-EXPRESSION AND FUNCTIONAL ENRICHMENT ANALYSIS TOOLS TO INTERPRET GENETIC ASSOCIATIONS

We have thus far talked about gene list functionality analysis provided by g:Profiler and looked into gene expression data exploration using funcExplorer to detect co-expressed gene groups. In the chapter at hand, we will focus on applying these tools to discover and dissect genetic variants associated with gene expression. Namely, we introduce Publication IV, a large-scale study where we applied different co-expression analysis methods, including funcExplorer, to gene expression data from various blood cell types and stimulated conditions. We detected genetic variants associated with gene modules, and used g:Profiler to characterise the modules and gain biological insights into the associations.

## 4.1. Background

Before introducing the study itself, the following sections give some background for the approaches we used in this study. We start with a brief introduction to genetic association studies with an emphasis on detecting genetic variants that affect gene expression levels. Finally, we present the objectives of the study.

### 4.1.1. Genome-wide association study

Uncovering the molecular mechanisms altered in common complex diseases is essential to identify novel and reliable drug targets or develop other suitable interventions. Genome-wide association studies (GWASs) have contributed to identifying tens of thousands of specific genetic variants associated with various human traits and diseases (Visscher et al., 2017). As of September 2018, the GWAS Catalog (Buniello et al., 2019) contains more than 71,000 variant-trait associations from more than 3,500 publications, including variants associated with cardiometabolic diseases (Nelson et al., 2017), autoimmune diseases (Dubois et al., 2010; Scott et al., 2017), and multiple cancers (Michailidou et al., 2015; Phelan et al., 2017). The idea of GWAS is that if one type of variant (one allele) is more frequent in people with the disease compared to people without the disease, the variant is likely to be associated with that disease. The associated variants are then considered to mark a region of the human genome that may influence the risk of developing the disease.

However, the underlying biological mechanisms often remain unclear in GWASs because most disease-associated genetic variants are in the non-coding regions of

the genome, indicating that these variants influence disease risk by regulating gene expression levels (Maurano et al., 2012). This means that the associated variant may not affect the nearest gene but rather some distant genes. In addition, it is sometimes hard to pinpoint the exact disease-causal variant because neighboring genetic variants are often correlated with one another due to the nonrandom association of alleles at different loci, also known as the linkage disequilibrium (LD), thus leading to multiple variants in a single region being associated with the trait. Furthermore, it is hard to distinguish in which cell types the disease-relevant GWAS variants act (Maurano et al., 2012). Therefore, interpreting GWAS results requires follow-up studies to infer specific causal variants, identify the genes they regulate, and the cell types or physiological contexts in which this regulation takes place. One way to fit GWAS findings into a broader biological context is to look for associations between variants and intermediate phenotypes, such as gene expression levels measured in different cell types and conditions (Rockman and Kruglyak, 2006). Multitude of approaches used to interpret GWAS findings are covered in (Cano-Gamez and Trynka, 2020).

### 4.1.2. Expression quantitative trait locus (eQTL) analysis

By the central dogma of molecular biology, gene expression is an intermediate link between DNA sequence and organismal phenotype. As reviewed by (Umans et al., 2020), profiling molecular traits, such as gene expression and DNA methylation, and integrating them with the GWAS results can be useful in linking non-coding variants to putative target genes and unveiling the underlying regulatory events. Therefore, studying expression quantitative trait loci (eQTLs) which are genetic variants that influence expression levels of one or more genes, has emerged (Rockman and Kruglyak, 2006).

A standard eQTL analysis performs a direct association test between SNPs and gene expression levels typically measured in tens or hundreds of individuals. Data required from every individual for eQTL analysis are genotype data, gene expression data, and covariates such as sex, age, smoking habits, environmental effect sources, etc. In a simplistic view, single eQTL mapping takes the expression levels of a gene and the genotype alleles at the position of a fixed SNP and then measures the correlation between the two and estimates the significance of the association as compared to random variants (Figure 13). The analysis identifies if the gene expression levels vary between genotypes encoded as the number of copies of the alternative allele carried by each individual, i.e., 0, 1 and 2 (Figure 13A).

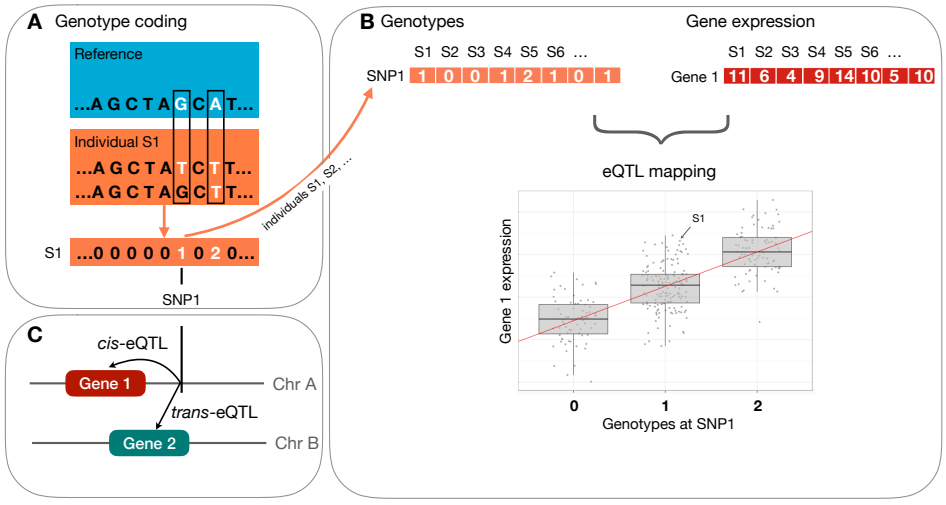



**Figure 13. Illustration of eQTL analysis.** **A** Genotypes are encoded as the number of copies of the alternative allele carried by each individual. Each individual can have 0, 1, or 2 times the alternative allele. **B** For each SNP, the vector of genotypes across all individuals is correlated with the vector of gene expression values for a specific gene across the same individuals. Box plots are often used to visualise the relationship between gene expression and genotypes. Points for each individual are usually added to these plots to convey a sense of the frequency of each genotype in the sample. The red line illustrates the linear regression model for eQTL analysis. **C** An eQTL is classified to *cis*- or *trans*-eQTL according to its location relative to the associated gene.

There are various statistical approaches used to obtain an estimate of the genotype effect on gene expression, such as linear (mixed) models (Ongen et al., 2016; Shabalin, 2012), Bayesian approaches (Imprialou et al., 2017), or even random forests (Stephan et al., 2015). A common example is fitting the following simple linear regression model for each gene–SNP pair of interest:

$$E = \beta_0 + \beta_1 G + \varepsilon,$$

where $E$ is the vector of gene expression values, $G$ is the genotype vector of the given SNP, and $\varepsilon$ is the normally distributed noise term. In the analysis of linear regression, the intercept $\beta_0$ and slope coefficient $\beta_1$ (also known as effect size) are estimated, followed by the calculation of a test statistic such as t-statistic or F-statistic (Shabalin, 2012). Finally, the $p-value$ corresponding to this test statistic is calculated to test the significance of regression. This process is repeated for all gene–SNP pairs of interest to identify all pairs with significant association at a given level after multiple testing correction.

Two types of eQTLs can be differentiated, those with local effects and those with distant effects. Local eQTLs or *cis*-eQTLs are located near the genomic location of the gene which produces the transcript or protein, and distant eQTLs or *trans*-eQTLs are located further away from the affected gene (Figure 13C) (Rockman and Kruglyak, 2006). The latter are often on different chromosomes or more

than five megabases away from the gene. Therefore, in the case of *cis*-eQTLs, the analysis is performed for all the variants in a limited region around each gene. For example, in a +/- 1 megabase window centered around the gene. Identification of *trans*-eQTLs, however, involves testing all other SNPs in a genome. In total, this can mean testing $> 10^6$ independent variants with $> 10^4$ genes.

### 4.1.3. Combining GWAS and eQTL analysis results

It has been shown that most of the GWAS variants overlap with expression QTLs (Nicolae et al., 2010), indicating that many disease-associated variants act through regulation of gene expression. Intuitively, if the same genetic variant is associated with both the risk of developing a disease A and the expression level of gene B then this could indicate that the genetic variant might influence disease A via gene B. Therefore, integrating GWAS summary data and eQTL data has potential to reveal target genes of disease-risk variants which cannot be found using the GWAS approach alone (Gallagher and Chen-Plotkin, 2018; Wainberg et al., 2019). Furthermore, eQTL analysis provides the means for functional characterisation of trait-associated variants through the target genes (Westra and Franke, 2014).

One class of approaches that has been used for linking GWAS and eQTL data are colocalisation methods. Colocalisation methods compare the association patterns of GWAS and eQTLs at a given region to find if the signals correlate and thereby provide evidence that both of the signals are driven by the same causal variants (Giambartolomei et al., 2014; Hormozdiari et al., 2016). However, colocalisation can also be considered between pairs of eQTLs, or pairs of GWASs.

Initial colocalisation approaches simply assessed whether GWAS variants were also significant eQTLs. Besides, a visual comparison of overlaps of association signals shown in Manhattan plots has been used. However, these approaches have pitfalls since often it is not clear whether the same variants are responsible for the two signals or whether it is distinct causal variants close to each other (Wallace et al., 2012). A popular colocalisation method that considers this is COLOC that relies on summary statistics to estimate the odds of colocalisation compared to the null hypothesis that the locus is not associated with any of the traits (Giambartolomei et al., 2014). We used this method also in Publication IV.

To summarise, the dynamics of the above mentioned analysis methods and data are illustrated in Figure 14.

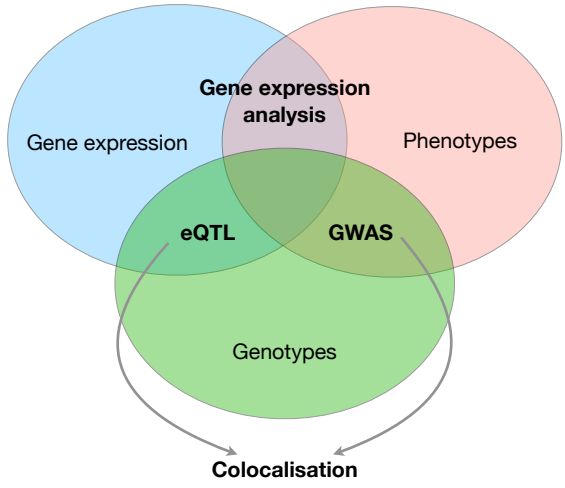



**Figure 14. Illustration of analysis methods.** The ovals represent different data types, and their intersections represent corresponding analysis methods.

### 4.1.4. Identifying credible sets of causal variants

Similar to GWASs, identifying potential causal variants from eQTL analyses is also hindered by the high correlation between neighboring SNPs. It is common to find multiple SNPs in a region that are associated with a trait. Fine mapping is sometimes used to overcome this complexity (Schaid et al., 2018). The aim of fine mapping is to define the most likely causal variants in a given region. First, significantly associated SNPs from eQTL analysis are determined. Then, the genomic region is partitioned around these SNPs to independent regions (loci) by considering the structure of LD among the variants. Statistical fine mapping is then conducted in each of the independent regions separately.

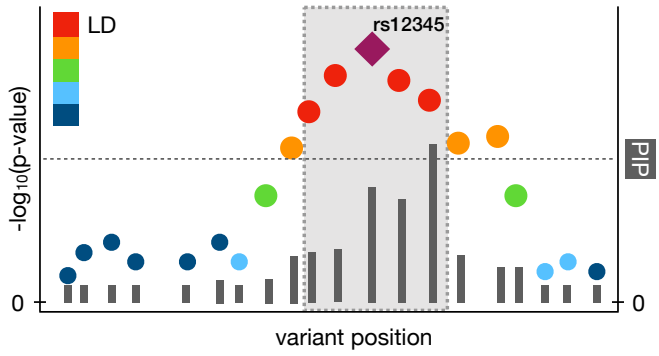



**Figure 15. Illustration of Bayesian fine mapping.** The y-axis on the left shows the -log10(p-values) from eQTL analysis, the horizontal dashed line represents the significance level. The variant chromosomal positions are on the x-axis. The purple diamond represents the lead SNP (rs12345) in the given locus. The results for other SNPs are coloured by descending degree of LD with the lead SNP (ordered red, orange, green and blue dots). The grey bars show the variant-level PIP values produced by Bayesian fine mapping methods (right axis). These PIP values can be summed to form credible sets based on a specified coverage probability threshold (for example, 95%). The light grey box represents the credible set selected by fine mapping. The figure is reproduced based on (Schaid et al., 2018).

There are several approaches used for this purpose. A heuristic approach is based on LD patterns with the lead SNP, i.e., the SNP with the highest association significance. All SNPs in that region that meet the given LD threshold are considered likely to be causally related to a trait (Schaid et al., 2018). In recent years, Bayesian methods for fine mapping have become popular (Benner et al., 2016; Maller et al., 2012; Wen et al., 2016) (Figure 15). The result is usually a minimum set of SNPs in the given region to capture the likely causal variant, also known as a credible set. These methods compute the posterior inclusion probabilities (PIP) for including each SNP as causal in a model. Then, the variants are ordered in the decreasing order of the PIP values, and the credible set is defined based on a pre-specified coverage probability threshold, often 95%. Namely, the ranked PIP values are summed until the cumulative probability exceeds the given threshold. The corresponding top $k$ variants are then considered to form a credible set (Maller et al., 2012). For example, one such fine mapping model is the Sum of Single Effects Model (SuSiE) (Wang et al., 2020).

### 4.1.5. Motivation for Publication IV

So far, detected *cis*-eQTLs largely outnumber the *trans*-eQTLs (Rotival et al., 2011), mostly because *cis*-eQTLs often have larger effect size and direct influence on gene expression (Westra and Franke, 2014). While most of the eQTL studies focus on *cis*-eQTLs, *trans*-eQTLs have shown promising results in identifying putative key driver genes that contribute to disease (Westra and Franke, 2014). Also, many *trans* associations are significantly mediated by the expression of *cis* genes (Ramdhani et al., 2020).

In order to provide more detailed molecular context, eQTL mapping studies have been performed in various cell types (Kasela et al., 2017; Momozawa et al., 2018), tissues (The GTEx Consortium et al., 2017) and with the effect of different environmental factors (Fairfax et al., 2014; Idaghdour et al., 2012). It has been shown that *trans*-eQTLs are more tissue and cell type-specific than *cis*-eQTLs and enriched for disease associations (Umans et al., 2020; Westra and Franke, 2014).

Although *trans*-eQTLs can provide valuable insights, detecting *trans*-eQTLs remains challenging due to limited sample sizes in existing experimental studies, small effect sizes of *trans*-eQTLs, and the burden of multiple-testing correction resulting from testing $> 10^6$ independent variants with $> 10^4$ genes (Grundberg et al., 2012; Umans et al., 2020; Võsa et al., 2018). Several studies have used co-expression analysis methods to reduce the number of tests by aggregating individual genes to co-expression modules and using a module's summary expression profile, eigengene, as a quantitative trait in the eQTL analysis (Hore et al., 2016; Kompass and Witte, 2011; Mao et al., 2019; Nath et al., 2017; Parts et al., 2011; Ramdhani et al., 2020; Rotival et al., 2011; Yang et al., 2017). The methods used in these *trans*-eQTL studies include sparse decomposition of arrays (SDA), ICA, WGCNA, and PLIER, but only a single method at a time. Potential effects of co-expression method choice have not been explored.

Another challenge is deciphering the regulatory mechanisms underlying the *trans*-eQTL associations. Since *trans*-eQTLs are often associated with several genes, functional enrichment analysis is sometimes used to identify significantly enriched pathways, biological processes, and transcription factors (Nath et al., 2017; Rotival et al., 2011). In addition to increasing the statistical power, co-expression modules are also beneficial because they can be directly interpreted as signatures of higher-level cellular phenotypes, such as activation of specific molecular pathways or transcription factors (Parts et al., 2011; Way et al., 2020). Incorporating such additional information enables the interpretation of the eQTL associations through the biological processes involved. However, as discussed in the previous chapters, it can be computationally complex to conduct enrichment analyses to such an extent systematically. Although this is a straightforward use-case for g:Profiler, the tool had not been fully exploited in this context before Publication IV.

## 4.2. Co-expression analysis reveals interpretable gene modules controlled by *trans*-acting genetic variants (Publication IV)

Most eQTL studies have been performed on whole tissue datasets, such as the brain or whole blood (The GTEx Consortium et al., 2017; Võsa et al., 2018), as opposed to specific cell types. In this study, we performed an extensive gene

module *trans*-eQTL analysis across six blood cell types, and three stimulated conditions from five published datasets (Figure 16A-B). Our goal was to identify potential cell type and condition-specific *trans*-eQTLs by employing co-expression analysis on a large combined dataset for increased statistical power. We ended up with a gene expression matrix that included expression values of 18,383 protein-coding genes with unique Ensembl identifiers across 3,938 samples after preprocessing and quality control.

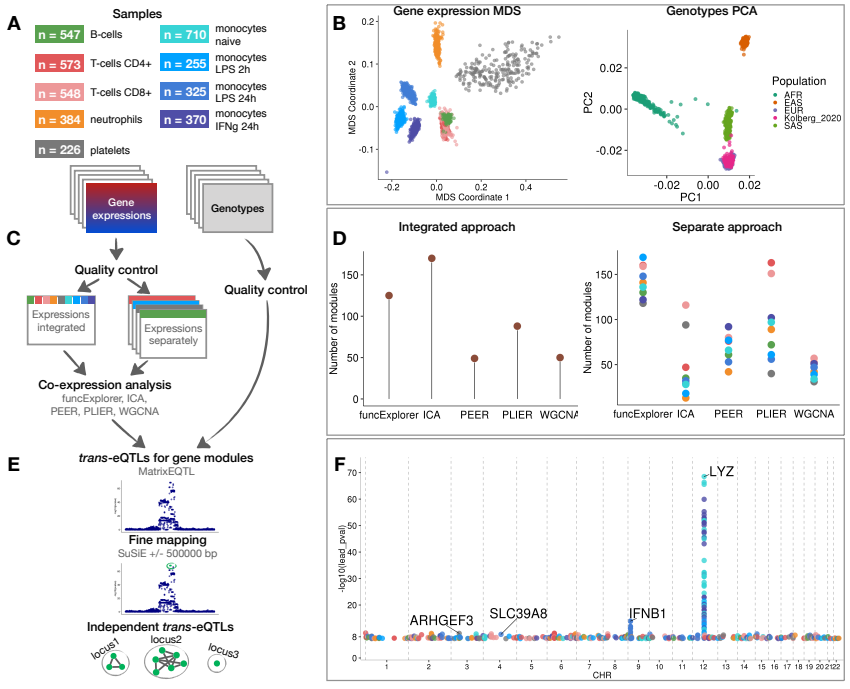



**Figure 16. Analysis workflow and results of Publication IV. A** Sample sizes of cell types and conditions included in the analysis. **B** Representation of gene expression data (shown in a multidimensional scaling (MDS) plot) and genotype data (shown in a PCA plot after projecting samples from this study to the 1000 Genomes Project reference populations) used for the eQTL analysis after quality control and normalisation. Cell types and conditions are color-coded according to panel A. **C** Analysis pipeline from quality control to applying five different co-expression methods on two partitions of gene expression data: (1) gene expression profiles across all cell types and conditions (integrated approach), (2) gene expression profiles from each cell type and condition separately (separate approach). **D** The number of gene modules detected from integrated and separate analyses. **E** Eigengenes of each gene module were used as phenotype in *trans*-eQTL analysis. To identify independent *trans*-eQTLs, we performed statistical fine mapping for all nominally significant ($p-value < 5 \times 10^{-8}$) associations and grouped together all associations with overlapping credible sets. **F** Manhattan plot of nominally significant ($p-value < 5 \times 10^{-8}$) *trans*-eQTLs. Each point corresponds to a gene module that was associated with the corresponding locus and is color-coded by the cell type from panel A. Adapted from Figure 1 in Publication IV.

### 4.2.1. Co-expression analysis

First, we figured that funcExplorer (Publication III) could be advantageous for performing *trans*-eQTL analysis because funcExplorer already incorporates prior information about biological pathways and gene sets and therefore disregards gene groups that are not biologically relevant. Furthermore, funcExplorer already retrieved eigengene profiles that can be used as a quantitative trait in the eQTL analysis. However, gene co-expression modules can be detected with various methods (see section 1.2.3). Since different methods solve distinct optimisation problems, they can identify complementary sets of gene modules (Stein-O'Brien et al., 2018). This is supported by a thorough benchmarking of multiple co-expression analysis methods demonstrating that there is no single best method or set of parameters for co-expression analysis (Way et al., 2020). Different methods and parameter settings discover biological aspects at varying levels of detail. Thus, to maximise gene module discovery, we decided to apply five co-expression analysis methods: ICA, PEER, PLIER, WGCNA, and funcExplorer. Out of these, ICA, PLIER, and PEER are matrix factorisation methods, and WGCNA and funcExplorer are clustering methods. In addition, funcExplorer and PLIER include prior biological knowledge.

Another aspect we considered in this study is data partitioning prior to co-expression analysis. This approach is particularly relevant when data from multiple cell types or conditions are analysed together. When co-expression analysis is performed across multiple cell types and conditions, then most detected gene co-expression modules are guided by differential expression between cell types (Quach et al., 2016; Van Dam et al., 2018). Consequently, cell-type-specific co-expression modules can remain undetected due to weak correlation in other cell types (Van Dam et al., 2018). Therefore, we applied the five methods to the full expression dataset integrating all the samples as well as individual cell types and conditions separately (Figure 16C). In total, we obtained 482 gene modules from the integrated approach and 3,509 from separate cell types (Figure 16D). The number of detected modules and their sizes varied due to the properties and the default parameters of each method (Figure 16D).

### 4.2.2. Detecting *trans*-eQTLs regulating gene modules

For every module, we extracted the eigengene profile representing a summary expression pattern of all the genes in the module. These expression profiles were used as quantitative traits in the eQTL analysis that included 6,861,056 common (minor allele frequency $> 5\%$) genetic variants passing strict quality control criteria. We performed *trans*-eQTL analysis in each cell type and condition separately using the MatrixEQTL (Shabalin, 2012) R package (v2.2) to fit a linear model adjusted for sex, batch (where available), and the first three principal components of the genotype data. The eigenvectors from the integrated approach were split into cell-type-specific sub-eigenvectors before the analysis. The results from every an-

alytical setting (data partitioning approach (n = 2), co-expression method (n = 5), cell type (n = 9), 90 *trans*-eQTL analyses in total) were then individually filtered to keep nominally significant variant-module associations ($p-value < 5 \times 10^{-8}$).

Next, we used the SuSiE model (Wang et al., 2020) to fine map all nominally significant associations to 864 independent credible sets of candidate causal variants (Figure 16E). For every gene module, we started fine mapping from the lead variant (variant with the smallest association $p-value$ for this module) and used a +/- 500,000 base pair (bp) window around the variant to detect the credible sets. We continued fine mapping iteratively with the next best nominally significant variant outside the previous window to account for LD. We repeated this process until no variants remained for the gene module.

Since we applied five co-expression methods to both integrated and cell-type-specific datasets, we found a large number of overlapping genetic associations. To group the individual *trans*-eQTLs to a set of non-overlapping loci and simplify the interpretation, we devised a novel aggregation approach based on the statistical fine mapping. First, we combined all credible sets into an undirected graph where every node represents a credible set of a module from a triplet (data partitioning approach, co-expression method, cell type). We defined an edge between two nodes if the corresponding credible sets shared at least one overlapping variant (Figure 17). We constructed the graph using the *igraph* R package. After obtaining the graph, we searched for connected components: subgraphs where every credible set is connected by a path. As a result, we combined the vast number of results into a list of 601 non-overlapping loci. However, these merged sets can no longer be interpreted as credible sets. We used this only as a heuristic approach to identify independent loci. Next, we defined the lead variant for every component by choosing the intersecting variant with the largest average PIP value across all the credible sets in the component. In the case of equal PIP values, we selected the first variant. Therefore, we considered the selected lead variant more as an indicator for the association's genomic region.

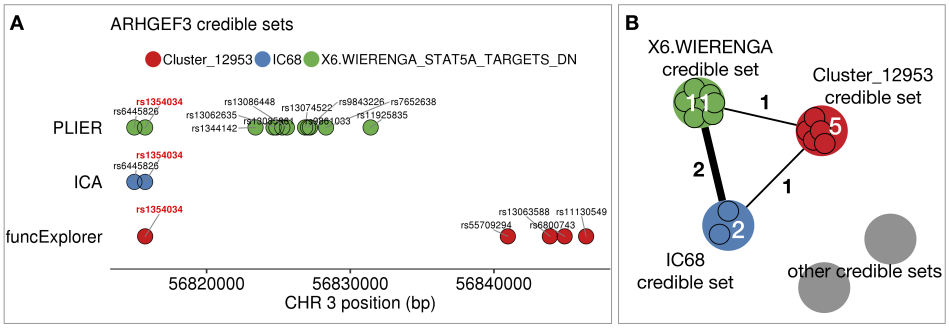



**Figure 17. Aggregating fine mapping credible sets in the example of *ARHGEF3* *trans*-eQTL locus. A** The figure shows an example of three credible sets on chromosome three for three gene modules: IC68 from applying ICA to integrated expression data; Cluster_12953 from funcExplorer and X6.WIERENGA_STAT5A_TARGETS_DN from PLIER, both applied to gene expression data from platelets. The associations shown here were detected in platelets. Each circle represents a variant belonging to the corresponding credible set, and colors distinguish the different credible sets. **B** We combined the credible sets into a graph structure where every node is a credible set of variants. We defined an edge between two nodes if they shared at least one variant. Grey circles represent all the credible sets from other genomic regions. Here, the number of shared variants is shown on the edges, and the number of variants belonging to the credible set is shown on the nodes. We searched for connected components, grouped these credible sets, and defined a lead variant for every group based on the largest average PIP value within the group. In this example, as all the credible sets share a single variant, rs1354034, then they are grouped, and the overlapping variant with the largest PIP is rs1354034. Adapted from Figure 1–figure supplement 2 in Publication IV.

### 4.2.3. Filtering *trans*-eQTLs

Genes in physical proximity often have correlated expression levels and could thus appear as co-expression modules in our analysis. As a result, if one or more genes in such modules have *cis*-eQTLs, then these *cis* variant–module associations would also be detected by our approach. To exclude co-expression modules that are driven by strong *cis*-effects, we performed gene-level eQTL analysis for 18,383 protein-coding genes and the 601 lead variants using the same approach as for the co-expression eQTL analysis. Next, we excluded the variant-module pairs from every credible set component together with corresponding credible sets where no *trans* associations (variant-level Benjamini-Hochberg FDR 5%) were included in the module. As *trans*-eQTLs, we considered variants that act on distant genes (>5 Mb away from the lead variant) and genes residing on different chromosomes. Furthermore, we performed one-sided Fisher's exact tests to assess the overlap between the modules and gene-level *trans* analysis associations. We excluded the variant-module pairs that did not significantly overlap with individual *trans* genes (Bonferroni-adjusted $p - value < 0.05$). After applying such filters, we repeated the process of aggregating credible sets, retaining 247 non-

overlapping loci that each were nominally associated with several gene modules (Figure 16F).

### 4.2.4. Characterising *trans*-eQTLs

To prioritise *trans*-eQTLs for further analyses and interpret the functional role of detected associations, we performed functional enrichment analysis for all the modules associated with the 247 loci using the g:Profiler toolset (Publication I). We had to analyse 343 significantly associated gene modules and 401 gene lists from the gene-level eQTL analysis for comparison. Therefore, we used the R package *gprofiler2* (Publication II) for this task to programmatically send the gene lists to the g:Profiler web server. Since the goal was to interpret independent loci, we aggregated the gene modules by the associated locus. We used the multiple query feature of g:Profiler to perform enrichment analyses for every list of locus-specific gene modules at once. This feature enabled better comparison of different gene modules and loci. At the same time, the enrichment results needed to be communicated with the co-authors. Thus, we generated g:Profiler web page short-links for instant sharing. Together with corresponding loci and other information, these links supplemented the study with the means for structured interpretation (see the summary results table in Supplementary file 1 in Publication IV). An example of the g:Profiler results for locus near the *SLC39A8* gene is available here: `https://biit.cs.ut.ee/gplink/l/aohV4uKeT1`. We discuss these results later in this chapter.

Besides functional characterisation, we looked for additional sources to further confirm and describe the *trans*-eQTLs. To identify if the *trans*-eQTLs might be associated with any higher-level phenotypes, we performed colocalisation analysis using GWAS summary statistics for 36 blood cell traits (Astle et al., 2016) and searched the lead variants from the GWAS Catalog database (Buniello et al., 2019). To evaluate the co-expression analysis approach, we performed a literature-based replication analysis. Although it is known that *trans*-eQTLs are more difficult to replicate between studies, additional challenges appeared due to the availability of published results such as the summary statistics or *trans*-associated genes. Nevertheless, we were able to extract *trans* genes from the supplementary materials of a selection of independent studies and compare these with our gene modules (see Table 1 in Publication IV). In addition, we compared associated modules in unstimulated monocytes, neutrophils, and T-cells to matched cell types from three independent studies for which we had access to individual-level data (Chen et al., 2016; Quach et al., 2016; Raj et al., 2014). We performed gene-level *trans*-eQTL analysis in these data and estimated the significance of the overlap between the list of significant genes and corresponding gene modules from our study. As a result, we were able to replicate three established associations. Namely, monocyte-specific *trans*-eQTLs at the *IFNB1* (Fairfax et al., 2014; Quach et al., 2016; Ramdhani et al., 2020; Ruffieux et al., 2020), and *LYZ* loci

(Fairfax et al., 2012; Rakitsch and Stegle, 2016; Rotival et al., 2011), and platelet-specific *trans*-eQTL at the *ARHGEF3* locus (Mao et al., 2019; Nath et al., 2017; Rotival et al., 2011; Wheeler et al., 2019).

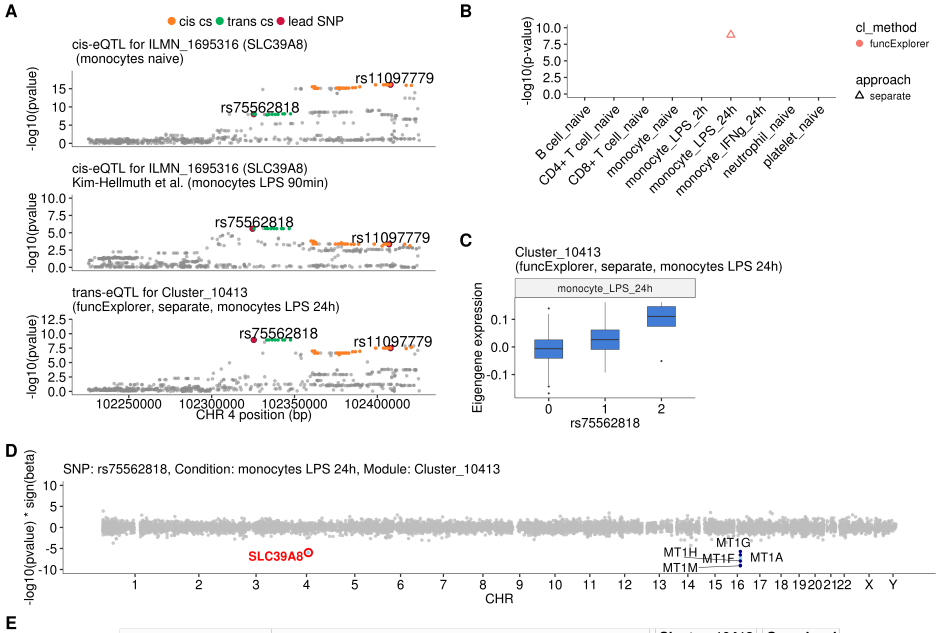
### 4.2.5. Platelet-specific *trans*-eQTL at the *ARHGEF3* locus

For the *ARHGEF3* locus, we conducted a more in-depth characterisation of the association. We were able to show that the *cis* and *trans*-eQTLs colocalise with a GWAS hit for mean platelet volume, platelet count, and plateletcrit (Astle et al., 2016). Although the *ARGEF3 trans*-eQTL has been previously detected in the whole blood, our analyses showed that the *cis* and *trans* effects were only present in platelets and not detected in other major blood cell types. We compared the *trans*-eQTL effect sizes in our platelet sample with the effects from the largest whole blood *trans*-eQTL meta-analysis (Võsa et al., 2018) and found a positive correlation (Pearson's $r = 0.68$, $p - value = 5.1 \times 10^{-12}$). The platelet specificity of the association was further supported by functional enrichment analysis with g:Profiler. Altogether, these results demonstrated how a *trans*-eQTL detected in whole blood can be driven by a strong signal present in only one cell type.

### 4.2.6. Novel association at the *SLC39A8* locus

Besides replicating known associations, we discovered a novel *trans*-eQTL at the *SLC39A8* locus regulating a module of metallothionein genes in monocytes stimulated for 24 hours with lipopolysaccharide (LPS), which is known to activate an inflammatory response (see Figure 18). The associated small module of five genes was detected only by funcExplorer because this was the only method that by default enabled to retrieve also smaller modules if supported by functional enrichment (Cluster_10413; `https://biit.cs.ut.ee/funcexplorer/link/c42d9`). Moreover, this module was identified from cell-type-specific clustering of the gene expression across samples from monocytes after 24 hours of LPS stimulation. All these five genes matched with the ones from the gene-level analysis that identified seven *trans*-associated genes in total (see Figure 18D). We also noticed that *SLC39A8* is strongly upregulated in response to LPS already at 2 hours. Still, we could not detect a colocalising *cis*-eQTL signal from naive and stimulated monocytes in our dataset (see Figure 18A). We speculated that there might be a transient eQTL earlier in the LPS response. To test this, we downloaded the *cis*-eQTL summary statistics from the (Kim-Hellmuth et al., 2017) study that had mapped eQTLs in monocytes stimulated with LPS for 90 min and 6 hours. Indeed, we found that the *cis*-eQTL 90 min after LPS stimulation colocalised with our *trans*-eQTL (see Figure 18A). This signal disappeared by 6 hours after stimulation, i.e., before the *trans* effect appeared.

The functional enrichment analysis results from g:Profiler guided the biological interpretation of this association. Namely, multiple GO terms and pathways related to zinc ion homeostasis are enriched in this module (see Figure 18E).




| Term ID | Term name | Cluster_10413 5 genes funcExplorer | Gene-level 8 genes NA |
|---|---|---|---|
| GO:0010273 | detoxification of copper ion | 1.1e-13 | 2.9e-19 |
| GO:0006882 | cellular zinc ion homeostasis | 1.9e-11 | 1.1e-19 |
| GO:0008270 | zinc ion binding | 3.6e-05 | 4.9e-07 |
| REAC:R-HSA-5661231 | Metallothioneins bind metals | 6.9e-14 | 1.9e-19 |
| TF:M00650_0 | Factor: MTF-1; motif: TBTGCACHCGGCCC; match class: 0 | 1.8e-02 | 2.1e-04 |
| WP:WP3529 | Zinc homeostasis | 2.3e-10 | 2.6e-17 |

**Figure 18. Transient eQTL near the *SLC39A8* locus.** **A** Regional plots comparing association signals between naive (rs11097779) and transiently induced *cis*-eQTLs (rs75562818) for *SLC39A8* and *trans*-eQTL (rs75562818) for a module of five co-expressed metallothionein genes. LPS-induced *cis*-eQTL summary statistics 90 min post stimulation (n = 134) were obtained from (Kim-Hellmuth et al., 2017). **B** Graph showing that the association between the module and *SLC39A8* locus is stimulation specific. As this module was detected by a cell-type-specific clustering, only a single value from the corresponding cell type is available. **C** Association between *trans*-eQTL (rs75562818) and eigengene of funcExplorer module Cluster_10413 in monocytes after 24 hr of LPS stimulation. **D** Manhattan plot of gene-level eQTL analysis for rs75562818. Dark blue points highlight the genes in module Cluster_10413. Light blue points show significantly associated genes (variant-level Benjamini-Hochberg FDR 5%) not included in the module. **E** Functional enrichment analysis of the *SLC39A8* associated module (see `https://biit.cs.ut.ee/gplink/l/aohV4uKeT1` for full results). The last column combines the FDR 5% significant genes from the gene-level analysis. The table shows adjusted enrichment p-values. Adapted from Figure 3 in Publication IV.

Furthermore, the promoter regions of the target genes are enriched for the binding motif of the metal transcription factor 1 (MTF1) transcription factor (see `https://biit.cs.ut.ee/gplink/l/aohV4uKeT1` for full results). These results suggested that a transient eQTL of the *SLC39A8* gene 90 min after stimulation regulates the expression of seven zinc-binding proteins 24 hours later, five of which were in the gene module. Multiple lines of literature evidence supported

this model (Kim et al., 2014; Liu et al., 2013; Nebert and Liu, 2019). Although we could not link this association to any diseases, it was used as an example for a relatively new field of dynamic eQTLs in the feature review article 'Where are the Disease-Associated eQTLs?' (Umans et al., 2020) to indicate the possibility of such dynamics in disease-associated eQTLs.

#### 4.2.7. Summary

While the strong *trans*-eQTL signals at the *IFNB1* and *LYZ* loci were detected by all co-expression methods in both integrated and separate analyses, most associations were found by only a subset of the analytical approaches. Out of the 247 nominally significant loci, 86 were found only by funcExplorer and 23 were found by funcExplorer and at least one of the other methods. For example, the *ARHGEF3* association was detected by three of the five methods, and *SLC39A8* only by funcExplorer. In addition, 138 loci were missed by funcExplorer. Therefore, combining multiple co-expression methods served its purpose and returned complementary results.

In summary, we demonstrated that co-expression module detection combined with gene set enrichment analysis could help identify interpretable *trans*-eQTLs to prioritise detailed experimental or computational characterisation. Still, these results depend on which co-expression method is chosen for the analysis and how the input data are partitioned beforehand.

### 4.3. Contribution

Conducting systematic exploration studies is essential for hypothesis-free discoveries. Integral parts of large-scale studies, such as the one presented in Publication IV, are numerous bioinformatics tools and methods designed to analyse and extract information from the experimental data. My contribution to this *trans*-eQTL study is extensive. I performed most of the analyses starting from the quality control of gene expression data through the co-expression analyses to eQTL mapping and fine mapping. Finally, I grouped the results into independent association regions and provided the accompanying functional enrichment results to facilitate interpretation. I had a significant role in writing the manuscript. Also, I prepared the figures and performed literature research for interpretation and replication.

# DISCUSSION

Bioinformatics software is essential for discovering and explaining patterns from high-throughput data. While the biologists ask the questions and experimental labs produce the data, computational skills and statistical methods are required to make sense of these data volumes that come in different forms. This situation is where the bioinformaticians and computer scientists help by developing tools that perform widely used analysis pipelines on user-provided data. Furthermore, the standard is to present the results in an easily understandable way with nice visualisations to convey the information. The work presented in this thesis covers the development and use of two bioinformatics tools for gene expression interpretation. Namely, the development of g:Profiler for gene list functional enrichment analysis (Publications I-II) and funcExplorer for co-expression clustering (Publication III).

The tools presented in this thesis can be considered academic software in computational biology characterised by interactive visualisations and features, increased shareability, and reproducibility. However, developing academic software introduces several challenges. One of which is keeping the software sustainable even though the incentives in academia favor the development and publication of new software and not the maintenance of existing tools. It is much harder to get funding for long-term software development, database maintenance, data curation, and documentation than for novel biological discovery obtained by applying different tools. On the other hand, extensive and expensive computational resources are needed for storing the data and performing the calculations of the web tools, such as the ones described in this thesis. For example, retaining g:Profiler archives takes a large amount of storage (on average, $\sim$ 300 GB per version), increasing every time a new update is released. FuncExplorer performs thousands of enrichment analyses in parallel by leveraging multiprocessing on a single machine. Fortunately, funcExplorer and g:Profiler have the opportunity to use the computational resources provided by the High Performance Computing Centre of the University of Tartu, and their maintenance is a key priority of ELIXIR Estonia. Nevertheless, since there rarely is a scientific paper that does not use any software for the novel findings, changes in funding strategies should be initiated to recognise more the software maintenance efforts.

Besides resources, active developers are essential for maintaining a widely used tool. While g:Profiler is a rare example of scientific software that has been taken from the prototype release to a production-grade software maintained by a team of developers, funcExplorer is an example of software released as a prototype. To alleviate this shortcoming, funcExplorer is an open-source project to enable community contribution and ensure full transparency. Providing open-source code is a common approach to extenuate potential discontinued development due to the financial limitations. Another challenge stems from the rapid advances in data representation and technologies. For example, although we up-

dated g:Profiler to keep up with the changes, now a major future work in funcExplorer is upgrading the tool to be compatible with the extensive technical update of g:Profiler. Maintaining R packages that make use of external packages is also affected by similar factors. For example, major updates in one dependency can lead to unwanted code changes in the other packages. In this case, however, it is more probable that the large and active R community contributes to the maintenance and improvement of a package. For example, there are already active users of the *gprofiler2* package who have developed different public add-ons to the core functionality.

More and more projects have strict data privacy requirements, especially when processing data that link to personal health. Therefore, uploading a whole dataset to external servers, such as funcExplorer or g:Profiler, can be problematic given the regulations of these projects and institutions. While g:Profiler does not record the queries unless a short-link is generated, funcExplorer stores the data in the back end server to be able to repeatedly serve the analysis results without running the entire pipeline every time a request is made. In either of the tools, at the moment, there is no systematic approach for removing all the traces of data and analysis results from our servers, but we can do it on request if needed. Nevertheless, although such deleting functionality could be helpful, from my experience, using stand-alone applications is the preferred solution in projects restricted by strict privacy and sharing rules. An alternative solution we have also considered for funcExplorer and g:Profiler is publishing the tools as separate Docker images that enable the users to build and run their own local instances of the tools. However, we foresaw that supporting such an extensive additional interface alongside a web tool can easily become an overhead because it requires further development and user support. In addition, it is harder to ensure that the underlying data used in local servers remains up-to-date. Furthermore, some of the g:Profiler data resources are restricted by licences that do not allow us to share these data. Thus, we have decided to prioritise the development of convenient and free web tools that provide additional interfaces for the most common use cases.

A well-known pitfall in gene clustering is the lack of a gold standard for cluster assessments. Biological associations have complex hierarchical structures, and thus the golden standards cannot be flat gene lists but should somehow incorporate these hierarchies. As different gene co-expression analysis reviews have stated, there is no single best method or parameter setting for co-expression analysis to reveal biological insight at all the different levels of granularity (Stein-O'Brien et al., 2018; Way et al., 2020). We introduced hypothesis-free gene co-expression analysis with the funcExplorer web tool. However, although funcExplorer is data-driven mainly in terms of clustering parameters such as the number of clusters to detect, it does not provide an entirely unambiguous solution for gene clustering. We proposed three different strategies based on functional enrichment, but which approach is the best remains unclear. A suggestion on how a researcher should address this challenge in large-scale exploratory analysis, such as the one performed

in Publication IV, is to apply different methods and strategies and then integrate the results to maximise the gene modules detection depicting various biological signals.

The essence of this work focuses on functional characterisation and providing visualisations such as interactive Manhattan plots that facilitate gene list interpretation by providing a global overview. However, the increase of produced data and the emergence of powerful new methods have also led to the rapid growth of the size and complexity of annotation databases. This growth has contributed to situations where a single gene list can be associated with hundreds of significant functional terms, most of which describe similar processes from different specificity levels and provide little if any additional information. Summarising and properly interpreting these long lists of terms can be a challenging task. Therefore, clever methods that reduce the amount of information without losing the essential details are needed. Broadly, there are two approaches used to tackle this issue. One way is to reduce the number of functions before enrichment analysis. For example, GO slims are cut-down versions of GO that usually contain a subset of more general functions while leaving out the specific fine-grained terms (The Gene Ontology Consortium, 2019). Other approaches post-process the significant functions to reduce the redundancy in the enrichment results (Alexa et al., 2006; Jantzen et al., 2011; Liao et al., 2019; Supek et al., 2011). However, each of the existing methods has key limitations, such as not considering the complex nature of annotations or requiring arbitrary thresholds. I have initiated the development of a parameter-free filtering algorithm for g:Profiler that identifies the driving biological functions from the long lists of significant but interrelated terms by utilising the known relations between these functions.

Sometimes, looking at the same thing from a different perspective can give fresh insight into the existing solution. However, the software developers often do not apply their products themselves for research purposes and thus might miss novel application routes or potential gaps in the implementation. In my work, I also took the user's role by applying these tools in the eQTL study presented in Publication IV. The study demonstrated that funcExplorer is applicable for purposes beyond its original role due to beneficial features such as the eigengene profiles and integrated functional interpretation. It also confirmed the capacity of g:Profiler for gene list interpretation and in-depth characterisation of genetic associations. Furthermore, using these tools in practice to answer research questions revealed some missing but useful features to include in these tools. Besides, funcExplorer and g:Profiler played a significant role in characterising the dynamic eQTL near the *SLC39A8* locus showing a possibility of identifying additional disease-associated eQTLs by keeping such transient dynamics in mind.

# CONCLUSION

The depth and volume of available experimental data are continuing to increase to this day. A key challenge is extracting biologically meaningful insights from this wealth of data. For this reason, the development and distribution of easy-to-use tools that enable both biologists and bioinformaticians to explore and analyse the data are of great importance.

This thesis aimed to highlight the relevance of developing flexible software for biological interpretation that continues to contribute to the scientific community and research. In this work, we introduced revised versions of two interlinked web tools for characterising gene expression experiments, g:Profiler and funcExplorer. First, as a member of the development team, I have led several visual design decisions and methodology evaluations related to the functional enrichment analysis in g:Profiler to ensure intuitive interpretation and fair comparison of results. To enable performing large-scale functional enrichment analyses programmatically, I designed and implemented the accompanying R package *gprofiler2* for the g:Profiler toolset. Second, I designed and implemented the co-expression analysis tool funcExplorer that uses g:Profiler to infer functionally related gene clusters from large-scale gene expression data. Finally, I applied the aforementioned functional interpretation tools in a large-scale eQTL analysis, where combining these tools facilitated systematic interpretation of genetic associations. Furthermore, funcExplorer and g:Profiler were the key methods that led to discovering a well-characterised example of transient eQTL in a relatively unexplored field of dynamic eQTLs.

Overall, the main conclusion of this thesis is that developing intuitive bioinformatics software is challenging, but the impact on scientific progress can be immeasurable when it succeeds.

# BIBLIOGRAPHY


A. Alexa and J. Rahnenfuhrer. *topGO: Enrichment Analysis for Gene Ontology*, 2019. R package version 2.38.1.

A. Alexa, J. Rahnenführer, and T. Lengauer. Improved scoring of functional groups from gene expression data by decorrelating GO graph structure. *Bioinformatics*, 22(13):1600–1607, 2006.

O. Alter, P. O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences*, 97(18):10101–10106, 2000.

N. Altman and M. Krzywinski. Points of Significance: Clustering. *Nature Methods*, 14(6):545–546, 2017.

R. A. Amezquita, A. T. Lun, E. Becht, V. J. Carey, L. N. Carpp, L. Geistlinger, F. Marini, K. Rue-Albrecht, D. Risso, C. Soneson, et al. Orchestrating single-cell analysis with Bioconductor. *Nature Methods*, 17(2):137–145, 2020.

W. J. Astle, H. Elding, T. Jiang, D. Allen, D. Ruklisa, A. L. Mann, D. Mead, H. Bouman, F. Riveros-Mckay, M. A. Kostadima, et al. The allelic landscape of human blood cell trait variation and links to common complex disease. *Cell*, 167(5):1415–1429, 2016.

A. Athar, A. Füllgrabe, N. George, H. Iqbal, L. Huerta, A. Ali, C. Snow, N. A. Fonseca, R. Petryszak, I. Papatheodorou, et al. ArrayExpress update–from bulk to single-cell expression data. *Nucleic Acids Research*, 47(D1):D711–D715, 2019.

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

C. Benner, C. C. Spencer, A. S. Havulinna, V. Salomaa, S. Ripatti, and M. Pirinen. FINEMAP: efficient variable selection using summary data from genome-wide association studies. *Bioinformatics*, 32(10):1493–1501, 2016.

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

J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Sciences*, 101(12):4164–4169, 2004.

A. Buniello, J. A. L. MacArthur, M. Cerezo, L. W. Harris, J. Hayhurst, C. Malangone, A. McMahon, J. Morales, E. Mountjoy, E. Sollis, et al. The NHGRI-EBI GWAS Catalog of published genome-wide association studies, targeted arrays and summary statistics 2019. *Nucleic Acids Research*, 47(D1):D1005–D1012, 2019.

E. Cano-Gamez and G. Trynka. From GWAS to Function: Using Functional

Genomics to Identify the Mechanisms Underlying Complex Diseases. *Frontiers in Genetics*, 11, 2020.

S. Carbon, A. Ireland, C. J. Mungall, S. Shu, B. Marshall, S. Lewis, A. Hub, and W. P. W. Group. AmiGO: online access to ontology and annotation data. *Bioinformatics*, 25(2):288–289, 2009.

L. Chen, B. Ge, F. P. Casale, L. Vasquez, T. Kwan, D. Garrido-Martín, S. Watt, Y. Yan, K. Kundu, S. Ecker, et al. Genetic drivers of epigenetic and transcriptional variation in human immune cells. *Cell*, 167(5):1398–1414, 2016.

C.-H. Chou, S. Shrestha, C.-D. Yang, N.-W. Chang, Y.-L. Lin, K.-W. Liao, W.-C. Huang, T.-H. Sun, S.-J. Tu, W.-H. Lee, et al. miRTarBase update 2018: a resource for experimentally validated microRNA-target interactions. *Nucleic Acids Research*, 46(D1):D296–D302, 2018.

D. J. Clarke, M. Jeon, D. J. Stein, N. Moiseyev, E. Kropiwnicki, C. Dai, Z. Xie, M. L. Wojciechowicz, S. Litz, J. Hom, J. E. Evangelista, L. Goldman, S. Zhang, C. Yoon, T. Ahamed, S. Bhuiyan, M. Cheng, J. Karam, K. M. Jagodnik, I. Shu, A. Lachmann, S. Ayling, S. L. Jenkins, and A. Ma'ayan. Appyters: Turning Jupyter Notebooks into data-driven web apps. *Patterns*, 2(3):100213, 2021. ISSN 2666-3899.

S. Cohen-Boulakia, K. Belhajjame, O. Collin, J. Chopard, C. Froidevaux, A. Gaignard, K. Hinsen, P. Larmande, Y. Le Bras, F. Lemoine, et al. Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities. *Future Generation Computer Systems*, 75:284–298, 2017.

A. Cruz, J. P. Arrais, and P. Machado. Interactive and coordinated visualization approaches for biological data analysis. *Briefings in Bioinformatics*, 20(4): 1513–1523, 2019.

D. Dembele and P. Kastner. Fuzzy C-means method for clustering microarray data. *Bioinformatics*, 19(8):973–980, 2003.

P. D'haeseleer. How does gene expression clustering work? *Nature Biotechnology*, 23(12):1499–1501, 2005.

H.-G. Drost and J. Paszkowski. Biomartr: genomic data retrieval with R. *Bioinformatics*, 33(8):1216–1217, 2017.

P. C. Dubois, G. Trynka, L. Franke, K. A. Hunt, J. Romanos, A. Curtotti, A. Zhernakova, G. A. Heap, R. Ádány, A. Aromaa, et al. Multiple common variants for celiac disease influencing immune gene expression. *Nature Genetics*, 42(4): 295–302, 2010.

E. Eden, R. Navon, I. Steinfeld, D. Lipson, and Z. Yakhini. GOrilla: a tool for discovery and visualization of enriched GO terms in ranked gene lists. *BMC Bioinformatics*, 10(1):1–7, 2009.

R. Edgar, M. Domrachev, and A. E. Lash. Gene Expression Omnibus: NCBI gene

expression and hybridization array data repository. *Nucleic Acids Research*, 30 (1):207–210, 2002.

M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, 1998.

B. P. Fairfax, S. Makino, J. Radhakrishnan, K. Plant, S. Leslie, A. Dilthey, P. Ellis, C. Langford, F. O. Vannberg, and J. C. Knight. Genetics of gene expression in primary immune cells identifies cell type–specific master regulators and roles of HLA alleles. *Nature Genetics*, 44(5):502–510, 2012.

B. P. Fairfax, P. Humburg, S. Makino, V. Naranbhai, D. Wong, E. Lau, L. Jostins, K. Plant, R. Andrews, C. McGee, et al. Innate immune activity conditions the effect of regulatory variants upon monocyte gene expression. *Science*, 343 (6175), 2014.

N. Fortelny, C. M. Overall, P. Pavlidis, and G. V. C. Freue. Can we predict protein from mRNA levels? *Nature*, 547(7664):E19–E20, 2017.

C. Fresno and E. A. Fernández. RDAVIDWebService: a versatile R interface to DAVID. *Bioinformatics*, 29(21):2810–2811, 2013.

B. C. Fung, K. Wang, and M. Ester. Hierarchical document clustering using frequent itemsets. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 59–70. SIAM, 2003.

M. D. Gallagher and A. S. Chen-Plotkin. The post-GWAS era: from association to function. *The American Journal of Human Genetics*, 102(5):717–730, 2018.

R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, et al. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biology*, 5(10): R80, 2004.

A. Ghazalpour, S. Doss, B. Zhang, S. Wang, C. Plaisier, R. Castellanos, A. Brozell, E. E. Schadt, T. A. Drake, A. J. Lusis, et al. Integrating genetic and network analysis to characterize genes related to mouse weight. *PLoS Genetics*, 2(8):e130, 2006.

C. Giambartolomei, D. Vukcevic, E. E. Schadt, L. Franke, A. D. Hingorani, C. Wallace, and V. Plagnol. Bayesian test for colocalisation between pairs of genetic association studies using summary statistics. *PLoS Genetics*, 10(5): e1004383, 2014.

M. Giurgiu, J. Reinhard, B. Brauner, I. Dunger-Kaltenbach, G. Fobo, G. Frishman, C. Montrone, and A. Ruepp. CORUM: the comprehensive resource of mammalian protein complexes—2019. *Nucleic Acids Research*, 47(D1):D559–D563, 2019.

E. Grundberg, K. S. Small, Å. K. Hedman, A. C. Nica, A. Buil, S. Keildson, J. T. Bell, T.-P. Yang, E. Meduri, A. Barrett, et al. Mapping cis-and trans-regulatory

effects across multiple tissues in twins. *Nature Genetics*, 44(10):1084–1089, 2012.

V. Hore, A. Viñuela, A. Buil, J. Knight, M. I. McCarthy, K. Small, and J. Marchini. Tensor decomposition for multiple-tissue gene expression experiments. *Nature Genetics*, 48(9):1094–1100, 2016.

F. Hormozdiari, M. Van De Bunt, A. V. Segre, X. Li, J. W. J. Joo, M. Bilow, J. H. Sul, S. Sankararaman, B. Pasaniuc, and E. Eskin. Colocalization of GWAS and eQTL signals detects target genes. *The American Journal of Human Genetics*, 99(6):1245–1260, 2016.

W. Huber, V. J. Carey, R. Gentleman, S. Anders, M. Carlson, B. S. Carvalho, H. C. Bravo, S. Davis, L. Gatto, T. Girke, et al. Orchestrating high-throughput genomic analysis with Bioconductor. *Nature Methods*, 12(2):115–121, 2015.

A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13(4-5):411–430, 2000.

Y. Idaghdour, J. Quinlan, J.-P. Goulet, J. Berghout, E. Gbeha, V. Bruat, T. De Malliard, J.-C. Grenier, S. Gomez, P. Gros, et al. Evidence for additive and interaction effects of host genotype and infection in malaria. *Proceedings of the National Academy of Sciences*, 109(42):16786–16793, 2012.

ImmunoMind Team. immunarch: An R Package for Painless Bioinformatics Analysis of T-Cell and B-Cell Immune Repertoires, Aug. 2019.

M. Imprialou, E. Petretto, and L. Bottolo. Expression QTLs mapping and analysis: a Bayesian perspective. In *Systems Genetics*, pages 189–215. Springer, 2017.

S. G. Jantzen, B. J. Sutherland, D. R. Minkley, and B. F. Koop. GO Trimming: Systematically reducing redundancy in large Gene Ontology datasets. *BMC Research Notes*, 4(1):267, 2011.

B. Jassal, L. Matthews, G. Viteri, C. Gong, P. Lorente, A. Fabregat, K. Sidiropoulos, J. Cook, M. Gillespie, R. Haw, et al. The reactome pathway knowledgebase. *Nucleic Acids Research*, 48(D1):D498–D503, 2020.

W. Jawaid. *enrichR: Provides an R Interface to 'Enrichr'*, 2019. R package version 2.1.

D. Jiang, C. Tang, and A. Zhang. Cluster analysis for gene expression data: a survey. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1370–1386, 2004.

X. Jiao, B. T. Sherman, D. W. Huang, R. Stephens, M. W. Baseler, H. C. Lane, and R. A. Lempicki. DAVID-WS: a stateful web service to facilitate gene/protein list analysis. *Bioinformatics*, 28(13):1805–1806, 2012.

M. Kanehisa, M. Furumichi, Y. Sato, M. Ishiguro-Watanabe, and M. Tanabe. KEGG: integrating viruses and cellular organisms. *Nucleic Acids Research*, 49(D1):D545–D551, 2021.

S. Kanwal, F. Z. Khan, A. Lonie, and R. O. Sinnott. Investigating reproducibility

and tracking provenance–A genomic workflow case study. *BMC Bioinformatics*, 18(1):1–14, 2017.

S. Kasela, K. Kisand, L. Tserel, E. Kaleviste, A. Remm, K. Fischer, T. Esko, H.-J. Westra, B. P. Fairfax, S. Makino, et al. Pathogenic implications for autoimmune mechanisms derived by comparative eQTL analysis of CD4+ versus CD8+ T cells. *PLoS Genetics*, 13(3):e1006643, 2017.

J.-H. Kim, J. Jeon, M. Shin, Y. Won, M. Lee, J.-S. Kwak, G. Lee, J. Rhee, J.-H. Ryu, C.-H. Chun, et al. Regulation of the catabolic cascade in osteoarthritis by the zinc-ZIP8-MTF1 axis. *Cell*, 156(4):730–743, 2014.

S. Kim-Hellmuth, M. Bechheim, B. Pütz, P. Mohammadi, Y. Nédélec, N. Giangreco, J. Becker, V. Kaiser, N. Fricker, E. Beier, et al. Genetic regulatory effects modified by immune activation contribute to autoimmune disease associations. *Nature Communications*, 8(1):1–10, 2017.

R. J. Kinsella, A. Kähäri, S. Haider, J. Zamora, G. Proctor, G. Spudich, J. Almeida-King, D. Staines, P. Derwent, A. Kerhornou, et al. Ensembl BioMarts: a hub for data retrieval across taxonomic space. *Database*, 2011, 2011.

B. Klaus and S. Reisenauer. An end to end workflow for differential gene expression using Affymetrix microarrays. *F1000Research*, 5, 2016.

S. Köhler, M. Gargano, N. Matentzoglu, L. C. Carmody, D. Lewis-Smith, N. A. Vasilevsky, D. Danis, G. Balagura, G. Baynam, A. M. Brower, et al. The Human Phenotype Ontology in 2021. *Nucleic acids research*, 49(D1):D1207–D1217, 2021.

L. Kolberg, U. Raudvere, I. Kuzmin, J. Vilo, and H. Peterson. gprofiler2–an R package for gene list functional enrichment analysis and namespace conversion toolset g: Profiler. *F1000Research*, 9(709):709, 2020.

R. Kolde and J. Vilo. GOsummaries: an R package for visual functional annotation of experimental data. *F1000Research*, 4, 2015.

K. S. Kompass and J. S. Witte. Co-regulatory expression quantitative trait loci mapping: method and application to endometrial cancer. *BMC Medical Genomics*, 4(1):6, 2011.

D. Krushevskaya, H. Peterson, J. Reimand, M. Kull, and J. Vilo. VisHiC—hierarchical functional enrichment analysis of microarray data. *Nucleic Acids Research*, 37(suppl_2):W587–W592, 2009.

M. V. Kuleshov, M. R. Jones, A. D. Rouillard, N. F. Fernandez, Q. Duan, Z. Wang, S. Koplev, S. L. Jenkins, K. M. Jagodnik, A. Lachmann, et al. Enrichr: a comprehensive gene set enrichment analysis web server 2016 update. *Nucleic Acids Research*, 44(W1):W90–W97, 2016.

M. Kull and J. Vilo. Fast approximate hierarchical clustering using similarity heuristics. *BioData Mining*, 1(1):9, 2008.

P. Langfelder and S. Horvath. Eigengene networks for studying the relationships between co-expression modules. *BMC Systems Biology*, 1(1):1–17, 2007.

P. Langfelder and S. Horvath. WGCNA: an R package for weighted correlation network analysis. *BMC Bioinformatics*, 9(1):1–13, 2008.

P. Langfelder, B. Zhang, and S. Horvath. Defining clusters from a hierarchical cluster tree: the Dynamic Tree Cut package for R. *Bioinformatics*, 24(5):719–720, 2008.

J. Leipzig. A review of bioinformatic pipeline frameworks. *Briefings in Bioinformatics*, 18(3):530–536, 2017.

G. Lemoine. *GWENA: Pipeline for augmented co-expression analysis.*, 2020. R package version 1.0.0.

Y. Liao, J. Wang, E. J. Jaehnig, Z. Shi, and B. Zhang. WebGestalt 2019: gene set analysis toolkit with revamped UIs and APIs. *Nucleic Acids Research*, 47(W1): W199–W205, 2019.

A. Liberzon, A. Subramanian, R. Pinchback, H. Thorvaldsdóttir, P. Tamayo, and J. P. Mesirov. Molecular signatures database (MSigDB) 3.0. *Bioinformatics*, 27(12):1739–1740, 2011.

W. Liebermeister. Linear modes of gene expression determined by independent component analysis. *Bioinformatics*, 18(1):51–60, 2002.

M.-J. Liu, S. Bao, M. Gálvez-Peralta, C. J. Pyle, A. C. Rudawsky, R. E. Pavlovicz, D. W. Killilea, C. Li, D. W. Nebert, M. D. Wewers, et al. ZIP8 regulates host defense through zinc-mediated inhibition of NF-κB. *Cell Reports*, 3(2):386–400, 2013.

J. Lonsdale, J. Thomas, M. Salvatore, R. Phillips, E. Lo, S. Shad, R. Hasz, G. Walters, F. Garcia, N. Young, et al. The genotype-tissue expression (GTEx) project. *Nature Genetics*, 45(6):580–585, 2013.

M. Love. *rnaseqGene: RNA-seq workflow: gene-level exploratory analysis and differential expression*, 2019. R package version 1.10.0.

M. I. Love, W. Huber, and S. Anders. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 15(12):550, 2014.

A. T. L. Lun, D. J. McCarthy, and J. C. Marioni. A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. *F1000Research*, 5:2122, 2016.

J. B. Maller, G. McVean, J. Byrnes, D. Vukcevic, K. Palin, Z. Su, J. M. Howson, A. Auton, S. Myers, A. Morris, et al. Bayesian refinement of association signals for 14 loci in 3 common diseases. *Nature Genetics*, 44(12):1294, 2012.

W. Mao, E. Zaslavsky, B. M. Hartmann, S. C. Sealfon, and M. Chikina. Pathway-level information extractor (PLIER) for gene expression data. *Nature Methods*, 16(7):607–610, 2019.

M. Martens, A. Ammar, A. Riutta, A. Waagmeester, D. N. Slenter, K. Hanspers,

R. A. Miller, D. Digles, E. N. Lopes, F. Ehrhart, et al. WikiPathways: connecting communities. *Nucleic Acids Research*, 49(D1):D613–D621, 2021.

M. T. Maurano, R. Humbert, E. Rynes, R. E. Thurman, E. Haugen, H. Wang, A. P. Reynolds, R. Sandstrom, H. Qu, J. Brody, et al. Systematic localization of common disease-associated variation in regulatory DNA. *Science*, 337(6099): 1190–1195, 2012.

D. J. McCarthy, Y. Chen, and G. K. Smyth. Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. *Nucleic Acids Research*, 40(10):4288–4297, 2012.

C. Meng, O. A. Zeleznik, G. G. Thallinger, B. Kuster, A. M. Gholami, and A. C. Culhane. Dimension reduction techniques for the integrative analysis of multi-omics data. *Briefings in Bioinformatics*, 17(4):628–641, 2016.

D. Merico, D. Gfeller, and G. D. Bader. How to visually interpret biological data using networks. *Nature Biotechnology*, 27(10):921–924, 2009.

H. Mi, D. Ebert, A. Muruganujan, C. Mills, L.-P. Albou, T. Mushayamaha, and P. D. Thomas. PANTHER version 16: a revised family classification, tree-based classification tool, enhancer regions and extensive API. *Nucleic Acids Research*, 49(D1):D394–D403, 2021.

K. Michailidou, J. Beesley, S. Lindstrom, S. Canisius, J. Dennis, M. J. Lush, M. J. Maranian, M. K. Bolla, Q. Wang, M. Shah, et al. Genome-wide association analysis of more than 120,000 individuals identifies 15 new susceptibility loci for breast cancer. *Nature genetics*, 47(4):373–380, 2015.

Y. Momozawa, J. Dmitrieva, E. Théâtre, V. Deffontaine, S. Rahmouni, B. Charloteaux, F. Crins, E. Docampo, M. Elansary, A.-S. Gori, et al. IBD risk loci are enriched in multigenic regulatory modules encompassing putative causative genes. *Nature Communications*, 9(1):2427, 2018.

J. Muller. *PANTHER.db: A set of annotation maps describing the entire PANTHER Gene Ontology*, 2019. R package version 1.0.10.

A. P. Nath, S. C. Ritchie, S. G. Byars, L. G. Fearnley, A. S. Havulinna, A. Joensuu, A. J. Kangas, P. Soininen, A. Wennerström, L. Milani, et al. An interaction map of circulating metabolites, immune gene networks, and their genetic regulation. *Genome Biology*, 18(1):1–15, 2017.

D. W. Nebert and Z. Liu. SLC39A8 gene encoding a metal ion transporter: discovery and bench to bedside. *Human Genomics*, 13(1):1–21, 2019.

C. P. Nelson, A. Goel, A. S. Butterworth, S. Kanoni, T. R. Webb, E. Marouli, L. Zeng, I. Ntalla, F. Y. Lai, J. C. Hopewell, et al. Association analyses based on false discovery rate implicate new loci for coronary artery disease. *Nature Genetics*, 49(9):1385, 2017.

D. L. Nicolae, E. Gamazon, W. Zhang, S. Duan, M. E. Dolan, and N. J. Cox. Trait-associated SNPs are more likely to be eQTLs: annotation to enhance discovery from GWAS. *PLoS Genetics*, 6(4):e1000888, 2010.

W. Ning, S. Lin, J. Zhou, Y. Guo, Y. Zhang, D. Peng, W. Deng, and Y. Xue. WocEA: The visualization of functional enrichment results in word clouds. *Journal of Genetics and Genomics= Yi chuan xue bao*, 45(7):415–417, 2018.

S. Oliver. Guilt-by-association goes global. *Nature*, 403(6770):601–602, 2000.

H. Ongen, A. Buil, A. A. Brown, E. T. Dermitzakis, and O. Delaneau. Fast and efficient QTL mapper for thousands of molecular phenotypes. *Bioinformatics*, 32(10):1479–1485, 2016.

J. D. Osborne, J. Flatow, M. Holko, S. M. Lin, W. A. Kibbe, L. J. Zhu, M. I. Danila, G. Feng, and R. L. Chisholm. Annotating the human genome with Disease Ontology. *BMC Genomics*, 10(1):1–8, 2009.

L. Parts, O. Stegle, J. Winn, and R. Durbin. Joint genetic analysis of gene expression data with inferred cellular phenotypes. *PLoS Genetics*, 7(1):e1001276, 2011.

G. A. Pavlopoulos, D. Malliarakis, N. Papanikolaou, T. Theodosiou, A. J. Enright, and I. Iliopoulos. Visualizing genome and systems biology: technologies, tools, implementation techniques and trends, past, present and future. *GigaScience*, 4 (1):s13742–015, 2015.

K. Peng, W. Xu, J. Zheng, K. Huang, H. Wang, J. Tong, Z. Lin, J. Liu, W. Cheng, D. Fu, et al. The Disease and Gene Annotations (DGA): an annotation resource for human disease. *Nucleic Acids Research*, 41(D1):D553–D560, 2012.

C. M. Phelan, K. B. Kuchenbaecker, J. P. Tyrer, S. P. Kar, K. Lawrenson, S. J. Winham, J. Dennis, A. Pirie, M. J. Riggan, G. Chornokur, et al. Identification of 12 new susceptibility loci for different histotypes of epithelial ovarian cancer. *Nature genetics*, 49(5):680–691, 2017.

J. Piñero, J. M. Ramírez-Anguita, J. Saüch-Pitarch, F. Ronzano, E. Centeno, F. Sanz, and L. I. Furlong. The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic Acids Research*, 48(D1):D845–D855, 2020.

H. Quach, M. Rotival, J. Pothlichet, Y.-H. E. Loh, M. Dannemann, N. Zidane, G. Laval, E. Patin, C. Harmant, M. Lopez, et al. Genetic adaptation and Neandertal admixture shaped the immune system of human populations. *Cell*, 167 (3):643–656, 2016.

J. Quackenbush. Computational analysis of microarray data. *Nature Reviews Genetics*, 2(6):418–427, 2001.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019.

T. Raj, K. Rothamel, S. Mostafavi, C. Ye, M. N. Lee, J. M. Replogle, T. Feng, M. Lee, N. Asinovski, I. Frohlich, et al. Polarization of the effects of autoimmune and neurodegenerative risk alleles in leukocytes. *Science*, 344(6183): 519–523, 2014.

B. Rakitsch and O. Stegle. Modelling local gene networks increases power to

detect trans-acting genetic effects on gene expression. *Genome Biology*, 17(1): 33, 2016.

S. Ramdhani, E. Navarro, E. Udine, A. G. Efthymiou, B. M. Schilder, M. Parks, A. Goate, and T. Raj. Tensor decomposition of stimulated monocyte and macrophage gene expression profiles identifies neurodegenerative disease-specific trans-eQTLs. *PLoS Genetics*, 16(2):e1008549, 2020.

U. Raudvere, L. Kolberg, I. Kuzmin, T. Arak, P. Adler, H. Peterson, and J. Vilo. g: Profiler: a web server for functional enrichment analysis and conversions of gene lists (2019 update). *Nucleic Acids Research*, 47(W1):W191–W198, 2019.

J. Reimand, M. Kull, H. Peterson, J. Hansen, and J. Vilo. g: Profiler—a web-based toolset for functional profiling of gene lists from large-scale experiments. *Nucleic Acids Research*, 35(suppl_2):W193–W200, 2007.

J. Reimand, T. Arak, and J. Vilo. g: Profiler—a web server for functional interpretation of gene lists (2011 update). *Nucleic Acids Research*, 39(suppl_2): W307–W315, 2011.

J. Reimand, T. Arak, P. Adler, L. Kolberg, S. Reisberg, H. Peterson, and J. Vilo. g: Profiler—a web server for functional interpretation of gene lists (2016 update). *Nucleic Acids Research*, 44(W1):W83–W89, 2016.

M. E. Ritchie, B. Phipson, D. Wu, Y. Hu, C. W. Law, W. Shi, and G. K. Smyth. limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Research*, 43(7):e47–e47, 2015.

A. W. Robert, A. B. B. Angulski, L. Spangenberg, P. Shigunov, I. T. Pereira, P. S. L. Bettes, H. Naya, A. Correa, B. Dallagiovanna, and M. A. Stimamiglio. Gene expression analysis of human adipose tissue-derived stem cells during the initial steps of in vitro osteogenesis. *Scientific Reports*, 8(1):1–11, 2018.

M. V. Rockman and L. Kruglyak. Genetics of global gene expression. *Nature Reviews Genetics*, 7(11):862–872, 2006.

M. Rotival, T. Zeller, P. S. Wild, S. Maouche, S. Szymczak, A. Schillert, R. Castagné, A. Deiseroth, C. Proust, J. Brocheton, et al. Integrating genome-wide genetic variations and monocyte expression data reveals trans-regulated gene modules in humans. *PLoS Genetics*, 7(12):e1002367, 2011.

H. Ruffieux, A. C. Davison, J. Hager, J. Inshaw, B. P. Fairfax, S. Richardson, L. Bottolo, et al. A global-local approach for detecting hotspots in multiple-response regression. *Annals of Applied Statistics*, 14(2):905–928, 2020.

W. Saelens, R. Cannoodt, and Y. Saeys. A comprehensive evaluation of module detection methods for gene expression data. *Nature Communications*, 9(1): 1–12, 2018.

D. J. Schaid, W. Chen, and N. B. Larson. From genome-wide associations to candidate causal variants by statistical fine-mapping. *Nature Reviews Genetics*, 19(8):491–504, 2018.

M. Schena, D. Shalon, R. W. Davis, and P. O. Brown. Quantitative monitoring of

gene expression patterns with a complementary DNA microarray. *Science*, 270 (5235):467–470, 1995.

M. Schmidt, D. Böhm, C. Von Törne, E. Steiner, A. Puhl, H. Pilch, H.-A. Lehr, J. G. Hengstler, H. Kölbl, and M. Gehrmann. The humoral immune system has a key prognostic impact in node-negative breast cancer. *Cancer Research*, 68 (13):5405–5413, 2008.

L. M. Schriml, E. Mitraka, J. Munro, B. Tauber, M. Schor, L. Nickle, V. Felix, L. Jeng, C. Bearer, R. Lichenstein, et al. Human Disease Ontology 2018 update: classification, content and workflow expansion. *Nucleic Acids Research*, 47 (D1):D955–D962, 2019.

R. A. Scott, L. J. Scott, R. Mägi, L. Marullo, K. J. Gaulton, M. Kaakinen, N. Pervjakova, T. H. Pers, A. D. Johnson, J. D. Eicher, et al. An expanded genome-wide association study of type 2 diabetes in Europeans. *Diabetes*, 66(11):2888–2902, 2017.

E. Secherre. *famat: Functional analysis of metabolic and transcriptomic data*, 2020. R package version 1.0.0.

A. Shabalin. Matrix eQTL: ultra fast eQTL analysis via large matrix operations. *Bioinformatics*, 28(10):1353–1358, 2012.

M. J. Silk, X. A. Harrison, and D. J. Hodgson. Perils and pitfalls of mixed-effects regression models in biology. *PeerJ*, 8:e9522, 2020.

D. K. Slonim. From patterns to pathways: gene expression data analysis comes of age. *Nature Genetics*, 32(4):502–508, 2002.

O. Stegle, L. Parts, M. Piipari, J. Winn, and R. Durbin. Using probabilistic estimation of expression residuals (PEER) to obtain increased power and interpretability of gene expression analyses. *Nature Protocols*, 7(3):500, 2012.

G. L. Stein-O'Brien, R. Arora, A. C. Culhane, A. V. Favorov, L. X. Garmire, C. S. Greene, L. A. Goff, Y. Li, A. Ngom, M. F. Ochs, et al. Enter the matrix: factorization uncovers knowledge from omics. *Trends in Genetics*, 34(10):790–805, 2018.

J. Stephan, O. Stegle, and A. Beyer. A random forest approach to capture genetic effects in the presence of population structure. *Nature Communications*, 6(1): 1–10, 2015.

A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

F. Supek and N. Škunca. *Visualizing GO Annotations*, pages 207–220. Springer New York, 2017.

F. Supek, M. Bošnjak, N. Škunca, and T. Šmuc. REVIGO summarizes and visualizes long lists of gene ontology terms. *PloS One*, 6(7):e21800, 2011.

O. Tanaseichuk, A. Hadj Khodabakhshi, D. Petrov, J. Che, T. Jiang, B. Zhou, A. Santrosyan, and Y. Zhou. An Efficient Hierarchical Clustering Algorithm for Large Datasets. *Austin Journal of Proteomics, Bioinformatics*, 2(1):1–6, 2015.

The Gene Ontology Consortium. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Research*, 47(D1):D330–D338, 2019.

The Gene Ontology Consortium. The Gene Ontology resource: enriching a GOld mine. *Nucleic Acids Research*, 49(D1):D325–D334, 2021.

The GTEx Consortium et al. The Genotype-Tissue Expression (GTEx) pilot analysis: Multitissue gene regulation in humans. *Science*, 348(6235):648–660, 2015.

The GTEx Consortium et al. Genetic effects on gene expression across human tissues. *Nature*, 550(7675):204–213, 2017.

The UniProt Consortium. UniProt: The universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, 2021.

T. Tian, Y. Liu, H. Yan, Q. You, X. Yi, Z. Du, W. Xu, and Z. Su. agriGO v2. 0: a GO analysis toolkit for the agricultural community, 2017 update. *Nucleic Acids Research*, 45(W1):W122–W129, 2017.

Z. Tian, W. He, J. Tang, X. Liao, Q. Yang, Y. Wu, and G. Wu. Identification of important modules and biomarkers in breast cancer based on WGCNA. *OncoTargets and Therapy*, 13:6805, 2020.

M. Uhlén, L. Fagerberg, B. M. Hallström, C. Lindskog, P. Oksvold, A. Mardinoglu, Å. Sivertsson, C. Kampf, E. Sjöstedt, A. Asplund, et al. Tissue-based map of the human proteome. *Science*, 347(6220), 2015.

B. D. Umans, A. Battle, and Y. Gilad. Where Are the Disease-Associated eQTLs? *Trends in Genetics*, 2020.

B. Uyar, D. Yusuf, R. Wurmus, N. Rajewsky, U. Ohler, and A. Akalin. RCAS: an RNA centric annotation system for transcriptome-wide regions of interest. *Nucleic Acids Research*, 45(10):e91–e91, 2017.

S. Van Dam, U. Vosa, A. van der Graaf, L. Franke, and J. P. de Magalhaes. Gene co-expression analysis for functional classification and gene–disease predictions. *Briefings in Bioinformatics*, 19(4):575–592, 2018.

C. J. Van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

P. M. Visscher, N. R. Wray, Q. Zhang, P. Sklar, M. I. McCarthy, M. A. Brown, and J. Yang. 10 years of GWAS discovery: biology, function, and translation. *The American Journal of Human Genetics*, 101(1):5–22, 2017.

U. Võsa, A. Claringbould, H.-J. Westra, M. J. Bonder, P. Deelen, B. Zeng, H. Kirsten, A. Saha, R. Kreuzhuber, S. Kasela, et al. Unraveling the polygenic architecture of complex traits using blood eqtl metaanalysis. *BioRxiv*, page 447367, 2018.

L. Wadi, M. Meyer, J. Weiser, L. D. Stein, and J. Reimand. Impact of outdated gene annotations on pathway enrichment analysis. *Nature Methods*, 13(9):705–706, 2016.

M. Wainberg, N. Sinnott-Armstrong, N. Mancuso, A. N. Barbeira, D. A. Knowles, D. Golan, R. Ermel, A. Ruusalepp, T. Quertermous, K. Hao, et al. Opportunities and challenges for transcriptome-wide association studies. *Nature Genetics*, 51 (4):592–599, 2019.

C. Wallace, M. Rotival, J. D. Cooper, C. M. Rice, J. H. Yang, M. McNeill, D. J. Smyth, D. Niblett, F. Cambien, C. Consortium, et al. Statistical colocalization of monocyte gene expression and genetic risk variants for type 1 diabetes. *Human Molecular Genetics*, 21(12):2815–2824, 2012.

G. Wang, A. Sarkar, P. Carbonetto, and M. Stephens. A simple new approach to variable selection in regression, with application to genetic fine mapping. *Journal of the Royal Statistical Society Series B*, 82(5):1273–1300, December 2020.

Z. Wang, M. Gerstein, and M. Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, 2009.

G. P. Way, M. Zietz, V. Rubinetti, D. S. Himmelstein, and C. S. Greene. Compressing gene expression data using multiple latent space dimensionalities learns complementary biological representations. *Genome Biology*, 21(1):1–27, 2020.

X. Wen, Y. Lee, F. Luca, and R. Pique-Regi. Efficient integrative multi-SNP association analysis via deterministic approximation of posteriors. *The American Journal of Human Genetics*, 98(6):1114–1129, 2016.

H.-J. Westra and L. Franke. From genome to function by studying eQTLs. *Biochimica et Biophysica Acta (BBA)-Molecular Basis of Disease*, 1842(10): 1896–1902, 2014.

H. E. Wheeler, S. Ploch, A. N. Barbeira, R. Bonazzola, A. Andaleon, A. Fotuhi Siahpirani, A. Saha, A. Battle, S. Roy, and H. K. Im. Imputed gene associations identify replicable trans-acting genes enriched in transcription pathways and complex traits. *Genetic Epidemiology*, 43(6):596–608, 2019.

E. Wingender. The TRANSFAC project as an example of framework technology that supports the analysis of genomic regulation. *Briefings in Bioinformatics*, 9 (4):326–332, 2008.

M. Q. Yang, D. Li, W. Yang, Y. Zhang, J. Liu, and W. Tong. A Gene Module-Based eQTL Analysis Prioritizing Disease Genes and Pathways in Kidney Cancer. *Computational and Structural Biotechnology Journal*, 15:463–470, 2017.

G. Yu. Enrichplot: Visualization of functional enrichment result. *R package version*, 1(2), 2018.

B. Zhang and S. Horvath. A general framework for weighted gene co-expression

network analysis. *Statistical Applications in Genetics and Molecular Biology*, 4(1), 2005.

Y. Zhou, B. Zhou, L. Pache, M. Chang, A. H. Khodabakhshi, O. Tanaseichuk, C. Benner, and S. K. Chanda. Metascape provides a biologist-oriented resource for the analysis of systems-level datasets. *Nature Communications*, 10(1):1–10, 2019.

# ACKNOWLEDGEMENTS

I am very grateful to my supervisor **Hedi Peterson**. About seven years ago, I contacted Hedi as I was looking for a summer job. She took me in even though I did not have any prior knowledge about bioinformatics nor software development. It has been a long summer. I thank her for introducing me to the academic world, creating all the conditions to get the work done, providing me various opportunities to travel, and constantly reminding the importance of mental health and personal life in all of this.

I thank professor **Jaak Vilo** for leading such a great bioinformatics research group BIIT and allowing me to be a proud member of this group. In addition, I would like to thank **Kaur Alasoo** for an inspiring collaboration, guiding me in my research, and always finding time to give constructive feedback. I'm grateful to **Leopold Parts** for proofreading this thesis and providing useful notes.

My special thanks go to the 'gP–boys', **Uku** and **Ivan**. You are the best teammates I could ever wish for. You have shown me that a great team can work well together in stressful situations without ever losing a sense of humor. I am very grateful for everything you have taught me.

I have met so many great people thanks to being a member of an academic family called **BIIT research group**. I appreciate all the fruitful discussions and brainstorming, but also the fun extracurricular activities and extended snacking breaks at the banana time. Especially I thank **Mari-Liis** and **Ahto** for starting this journey together with me and sharing the burden. Without you I would not have been able to do it. I am passing on the baton to you now. Also, I am deeply grateful to **Elena** for all the motivation, support, and advice she has given me in many aspects of doctoral studies and life in general. I have no words on how much positivity you have given me. Additionally, I want to thank **Sulev** and **Kaido** for being my predecessors and always kindly sharing the lessons learned, and all the other BIIT members too for the ever friendly environment.

The computational work in this thesis was done using different resources from the High Performance Computing Center of the University of Tartu. I am very grateful to the whole **HPC team**, especially to **Sander** who has been very patient with me over these years and always found a solution to my problems very quickly.

Ma olen väga tänulik oma **perekonnale**, et nad ei ole kunagi kahelnud minu otsustes ja on toetanud mind alati kõigis minu tegemistes. Viimaseks, kuid mitte vähem tähtsaks, on mul väga vedanud, et **Mart** on olnud minu kõrval kogu selle konarliku teekonna vältel. Aitäh, et oled alati minusse uskunud, minu kurtmist kuulanud ja mind tagant tõuganud just nendel hetkedel, mil olen seda kõige rohkem vajanud.

# SISUKOKKUVÕTE

## Bioinformaatika tööriistade arendamine ja rakendamine geenieksпressiooni andmete interpreteerimiseks

Laborid üle maailma toodavad suurel hulgal eksperimentaalseid bioloogilisi andmeid, mida tuleb analüüsida ja tõlgendada, et saada neist andmetest väärtuslikke ja kasutatavaid teadmisi. Muuhulgas on tehnoloogiate arengu tõttu nende andmekogumite mahud oluliselt suurenenud. Selliste andmete analüüsiks kasutatakse erinevaid arvutuslikke meetodeid, mille rakendamiseks on tarvis head programmeerimisoskust või bioinformaatikute abi. Andmeid koguvate ja nende kohta teaduslikke küsimusi esitavate teadlaste arv ületab aga bioinformaatikute arvu. Seetõttu loovad bioinformaatikud lihtsasti kasutatavaid tööriistu, mis võimaldavad bioloogidel erinevaid arvutuslikke analüüse ise läbi viia. Sellised tööriistad annavad teadlastele võimaluse otsida suuremahulistest andmetest, nagu geenieksпressiooni andmed, erinevaid mustreid ja koondada leitud teadmised, et paremini kirjeldada uuritavat süsteemi.

Käesolevas doktoritöös arendame kahte bioinformaatika tööriista, mis on mõeldud geenieksпressiooni andmete interpreteerimiseks. Esimesena tutvustame uuendatud veebitööriista g:Profiler, mille põhiliseks rakenduseks on geeninimekirjade funktsionaalse rikastamise analüüsi läbi viimine (Publikatsioon I). Täpsemalt keskendume töös g:Profileri uute funktsionaalsuste kirjeldamisele ja analüüsi tulemuste visuaalsele esitamisele, kasutades selleks uuenduslikku Manhattani joonist. Lisaks arendasime ka kaasneva R-i paketi nimega *gprofiler2*, mis võimaldab g:Profilerit hõlpsasti integreerida ka automatiseeritud analüüsi töövoogudesse (Publikatsioon II). Sarnaselt veebitööriistale võimaldab *gprofiler2* pakett ka interaktiivse Manhattani joonise kuvamist.

Eelmainitud tööriistade kasutamiseks on ennekõike tarvis leida huvitavad geeninimekirjad, mida soovitakse bioloogiliselt kirjeldada. Tavaliselt leitakse sellised geenid pika ja keerulise geenieksпressiooni andmete analüüsi tulemusena. Selle protsessi lihtsustamiseks arendasime käesolevas töös veebitööriista funcExplorer, mis integreerib geenieksпressiooni andmete klasterdamise ja g:Profileri abil tehtud funktsionaalse rikastatuse analüüsi (Publikatsioon III). Tööriista eesmärgiks on tuvastada koos-ekspresseeritud geenigrupid ning anda eksperimentaalsetest andmetest kirjeldav ülevaade. Selleks ühendab funcExplorer erinevad andmeanalüüsi osad üheks tervikuks ja esitab saadud tulemused kasutajale selliselt, et neid saab lihtsasti uurida, tõlgendada ning teistega jagada.

Töö viimases osas demonstreerime nende tööriistade kasulikkust, rakendades neid geenieksпressiooni tasemetega seotud geneetiliste variantide süstemaatilises analüüsis (Publikatsioon IV). Täpsemalt, viisime läbi geneetiliste variantide ja geenigruppide vahelise assotsiatsioonianalüüsi, et tuvastada need variandid, mis mõjutavad korraga mitme geeni ekspressioonitasemeid. Teiste meetodite hulgas kasutasime ka funcExplorerit, et leida koos-ekspresseerunud geenigruppe.

Seejärel kasutasime g:Profilerit, et seotud geenigruppe kirjeldada vastavalt nende ühistele bioloogilistele funktsioonidele. Meie tulemused kinnitasid mitmeid varem teadaolevaid seoseid. Lisaks, rakendades funcExplorerit andmestikule, mis on saadud monotsüütide stimuleerimisel lipopolüsahhariididega (LPS) 24 tundi, leidsime ühe uue seose. Nimelt leidis funcExplorer metallotioneiini geenide grupi, mille kohta selgus, et seda gruppi reguleerib *SLC39A8* geeni läheduses asuv variant. Huvitaval kombel leidsime, et seda seost vahendas mööduv efekt, mis esines ainult LPS'i varasemas vastuses ja mis kadus enne leitud geenigrupi efekti ilmnemist. Teisisõnu mõjutas see geneetiline variant geeni *SLC39A8* ekspressiooni tunde varem, vahetult pärast stimulatsiooni. Sellise laiaulatusliku uuringu läbiviimisel ja tulemuste tõlgendamisel oli kandev roll käesolevas doktoritöös arendatud kasutajasõbralikel bioinformaatika tööriistadel.

# PUBLICATIONS

# CURRICULUM VITAE

## Personal data

| | |
|---|---|
| Name: | Liis Kolberg |
| Date of birth: | February 5th 1990 |
| Nationality: | Estonian |
| E-mail: | liis.kolberg@gmail.com |

## Education

| | |
|---|---|
| 2016–... | PhD studies, Computer Science, Institute of Computer Science, Faculty of Science and Technology, University of Tartu, Estonia |
| 2012–2015 | Master's studies, Financial and Actuarial Mathematics, Faculty of Mathematics and Computer Science, University of Tartu, Estonia |
| 2009–2012 | Bachelor's studies, Mathematics, Faculty of Mathematics and Computer Science, University of Tartu, Estonia |
| 1997–2009 | Oskar Lutsu Palamuse Gümnaasium (Silver medal), Estonia |

## Employment

| | |
|---|---|
| 2018–... | Junior Research Fellow of Bioinformatics, Institute of Computer Science, Faculty of Science and Technology, University of Tartu |
| 2014–2018 | Programmer, Institute of Computer Science, Faculty of Science and Technology, University of Tartu |

## Scientific work

**Main fields of interest:** data science, bioinformatics, computational methods, functional enrichment analysis, gene expression analysis

# ELULOOKIRJELDUS

## Isikuandmed


| | |
|---|---|
| Nimi: | Liis Kolberg |
| Sünniaeg: | 05.02 1990 |
| Rahvus: | Eestlane |
| E-post: | liis.kolberg@gmail.com |


## Haridus

| | |
|---|---|
| 2016–... | Doktoriõpe, informaatika, loodus-ja täppisteaduste valdkond, Tartu Ülikool, Eesti |
| 2012–2015 | Magistriõpe, finants- ja kindlustusmatemaatika, Tartu Ülikool, Eesti |
| 2009–2012 | Bakalaureuseõpe, matemaatika, Tartu Ülikool, Eesti |
| 1997–2009 | Oskar Lutsu Palamuse Gümnaasium (hõbemedal), Eesti |

## Teenistuskäik

| | |
|---|---|
| 2018–... | Bioinformaatika nooremteadur, arvutiteaduse instituut, loodus- ja täppisteaduste valdkond, Tartu Ülikool |
| 2014–2018 | Programmeerija, arvutiteaduse instituut, loodus- ja täppisteaduste valdkond, Tartu Ülikool |

## Teadustegevus

**Peamised uurimisvaldkonnad:** andmeteadus, bioinformaatika, arvutuslikud meetodid, geenide funktsionaalsuse analüüs, geenieksprессiooni analüüs

# DISSERTATIONES INFORMATICAE PREVIOUSLY PUBLISHED IN DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS

19. **Helger Lipmaa.** Secure and efficient time-stamping systems. Tartu, 1999, 56 p.
22. **Kaili Müürisep.** Eesti keele arvutigrammatika: süntaks. Tartu, 2000, 107 lk.
23. **Varmo Vene.** Categorical programming with inductive and coinductive types. Tartu, 2000, 116 p.
24. **Olga Sokratova.** Ω-rings, their flat and projective acts with some applications. Tartu, 2000, 120 p.
27. **Tiina Puolakainen.** Eesti keele arvutigrammatika: morfoloogiline ühestamine. Tartu, 2001, 138 lk.
29. **Jan Villemson.** Size-efficient interval time stamps. Tartu, 2002, 82 p.
45. **Kristo Heero.** Path planning and learning strategies for mobile robots in dynamic partially unknown environments. Tartu 2006, 123 p.
49. **Härmel Nestra.** Iteratively defined transfinite trace semantics and program slicing with respect to them. Tartu 2006, 116 p.
53. **Marina Issakova.** Solving of linear equations, linear inequalities and systems of linear equations in interactive learning environment. Tartu 2007, 170 p.
55. **Kaarel Kaljurand.** Attempto controlled English as a Semantic Web language. Tartu 2007, 162 p.
56. **Mart Anton.** Mechanical modeling of IPMC actuators at large deformations. Tartu 2008, 123 p.
59. **Reimo Palm.** Numerical Comparison of Regularization Algorithms for Solving Ill-Posed Problems. Tartu 2010, 105 p.
61. **Jüri Reimand.** Functional analysis of gene lists, networks and regulatory systems. Tartu 2010, 153 p.
62. **Ahti Peder.** Superpositional Graphs and Finding the Description of Structure by Counting Method. Tartu 2010, 87 p.
64. **Vesal Vojdani.** Static Data Race Analysis of Heap-Manipulating C Programs. Tartu 2010, 137 p.
66. **Mark Fišel.** Optimizing Statistical Machine Translation via Input Modification. Tartu 2011, 104 p.
67. **Margus Niitsoo**. Black-box Oracle Separation Techniques with Applications in Time-stamping. Tartu 2011, 174 p.
71. **Siim Karus.** Maintainability of XML Transformations. Tartu 2011, 142 p.
72. **Margus Treumuth.** A Framework for Asynchronous Dialogue Systems: Concepts, Issues and Design Aspects. Tartu 2011, 95 p.
73. **Dmitri Lepp.** Solving simplification problems in the domain of exponents, monomials and polynomials in interactive learning environment T-algebra. Tartu 2011, 202 p.

74. **Meelis Kull.** Statistical enrichment analysis in algorithms for studying gene regulation. Tartu 2011, 151 p.
77. **Bingsheng Zhang.** Efficient cryptographic protocols for secure and private remote databases. Tartu 2011, 206 p.
78. **Reina Uba.** Merging business process models. Tartu 2011, 166 p.
79. **Uuno Puus.** Structural performance as a success factor in software development projects – Estonian experience. Tartu 2012, 106 p.
81. **Georg Singer.** Web search engines and complex information needs. Tartu 2012, 218 p.
83. **Dan Bogdanov.** Sharemind: programmable secure computations with practical applications. Tartu 2013, 191 p.
84. **Jevgeni Kabanov.** Towards a more productive Java EE ecosystem. Tartu 2013, 151 p.
87. **Margus Freudenthal.** Simpl: A toolkit for Domain-Specific Language development in enterprise information systems. Tartu, 2013, 151 p.
90. **Raivo Kolde.** Methods for re-using public gene expression data. Tartu, 2014, 121 p.
91. **Vladimir Šor.** Statistical Approach for Memory Leak Detection in Java Applications. Tartu, 2014, 155 p.
92. **Naved Ahmed.** Deriving Security Requirements from Business Process Models. Tartu, 2014, 171 p.
94. **Liina Kamm.** Privacy-preserving statistical analysis using secure multi-party computation. Tartu, 2015, 201 p.
100. **Abel Armas Cervantes.** Diagnosing Behavioral Differences between Business Process Models. Tartu, 2015, 193 p.
101. **Fredrik Milani.** On Sub-Processes, Process Variation and their Interplay: An Integrated Divide-and-Conquer Method for Modeling Business Processes with Variation. Tartu, 2015, 164 p.
102. **Huber Raul Flores Macario.** Service-Oriented and Evidence-aware Mobile Cloud Computing. Tartu, 2015, 163 p.
103. **Tauno Metsalu.** Statistical analysis of multivariate data in bioinformatics. Tartu, 2016, 197 p.
104. **Riivo Talviste.** Applying Secure Multi-party Computation in Practice. Tartu, 2016, 144 p.
108. **Siim Orasmaa.** Explorations of the Problem of Broad-coverage and General Domain Event Analysis: The Estonian Experience. Tartu, 2016, 186 p.
109. **Prastudy Mungkas Fauzi.** Efficient Non-interactive Zero-knowledge Protocols in the CRS Model. Tartu, 2017, 193 p.
110. **Pelle Jakovits.** Adapting Scientific Computing Algorithms to Distributed Computing Frameworks. Tartu, 2017, 168 p.
111. **Anna Leontjeva.** Using Generative Models to Combine Static and Sequential Features for Classification. Tartu, 2017, 167 p.
112. **Mozhgan Pourmoradnasseri.** Some Problems Related to Extensions of Polytopes. Tartu, 2017, 168 p.

113. **Jaak Randmets.** Programming Languages for Secure Multi-party Computation Application Development. Tartu, 2017, 172 p.
114. **Alisa Pankova.** Efficient Multiparty Computation Secure against Covert and Active Adversaries. Tartu, 2017, 316 p.
116. **Toomas Saarsen.** On the Structure and Use of Process Models and Their Interplay. Tartu, 2017, 123 p.
121. **Kristjan Korjus.** Analyzing EEG Data and Improving Data Partitioning for Machine Learning Algorithms. Tartu, 2017, 106 p.
122. **Eno Tõnisson.** Differences between Expected Answers and the Answers Offered by Computer Algebra Systems to School Mathematics Equations. Tartu, 2017, 195 p.

# DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS

1. **Abdullah Makkeh**. Applications of Optimization in Some Complex Systems. Tartu 2018, 179 p.
2. **Riivo Kikas**. Analysis of Issue and Dependency Management in Open-Source Software Projects. Tartu 2018, 115 p.
3. **Ehsan Ebrahimi**. Post-Quantum Security in the Presence of Superposition Queries. Tartu 2018, 200 p.
4. **Ilya Verenich**. Explainable Predictive Monitoring of Temporal Measures of Business Processes. Tartu 2019, 151 p.
5. **Yauhen Yakimenka**. Failure Structures of Message-Passing Algorithms in Erasure Decoding and Compressed Sensing. Tartu 2019, 134 p.
6. **Irene Teinemaa**. Predictive and Prescriptive Monitoring of Business Process Outcomes. Tartu 2019, 196 p.
7. **Mohan Liyanage.** A Framework for Mobile Web of Things. Tartu 2019, 131 p.
8. **Toomas Krips.** Improving performance of secure real-number operations. Tartu 2019, 146 p.
9. **Vijayachitra Modhukur.** Profiling of DNA methylation patterns as biomarkers of human disease. Tartu 2019, 134 p.
10. **Elena Sügis.** Integration Methods for Heterogeneous Biological Data. Tartu 2019, 250 p.
11. **Tõnis Tasa.** Bioinformatics Approaches in Personalised Pharmacotherapy. Tartu 2019, 150 p.
12. **Sulev Reisberg.** Developing Computational Solutions for Personalized Medicine. Tartu 2019, 126 p.
13. **Huishi Yin.** Using a Kano-like Model to Facilitate Open Innovation in Requirements Engineering. Tartu 2019, 129 p.
14. **Faiz Ali Shah.** Extracting Information from App Reviews to Facilitate Software Development Activities. Tartu 2020, 149 p.
15. **Adriano Augusto**. Accurate and Efficient Discovery of Process Models from Event Logs. Tartu 2020, 194 p.
16. **Karim Baghery.** Reducing Trust and Improving Security in zk-SNARKs and Commitments. Tartu 2020, 245 p.
17. **Behzad Abdolmaleki.** On Succinct Non-Interactive Zero-Knowledge Protocols Under Weaker Trust Assumptions. Tartu 2020, 209 p.
18. **Janno Siim.** Non-Interactive Shuffle Arguments. Tartu 2020, 154 p.
19. **Ilya Kuzovkin.** Understanding Information Processing in Human Brain by Interpreting Machine Learning Models. Tartu 2020, 149 p.
20. **Orlenys López Pintado.** Collaborative Business Process Execution on the Blockchain: The Caterpillar System. Tartu 2020, 170 p.
21. **Ardi Tampuu.** Neural Networks for Analyzing Biological Data. Tartu 2020, 152 p.

22. **Madis Vasser.** Testing a Computational Theory of Brain Functioning with Virtual Reality. Tartu 2020, 106 p.
23. **Ljubov Jaanuska.** Haar Wavelet Method for Vibration Analysis of Beams and Parameter Quantification. Tartu 2021, 192 p.
24. **Arnis Parsovs.** Estonian Electronic Identity Card and its Security Challenges. Tartu 2021, 214 p.
25. **Kaido Lepik.** Inferring causality between transcriptome and complex traits. Tartu 2021, 224 p.
26. **Tauno Palts.** A Model for Assessing Computational Thinking Skills. Tartu 2021, 134 p.