Gene regulatory network modelling with evolutionary algorithms - an integrative approach

Alina Sîrbu

B.Sc. Computer Science

A Dissertation submitted in part fulfilment of the

requirements for the award of

Doctor of Philosophy (Ph.D.)

to the

Dublin City University

Faculty of Engineering and Computing, School of Computing

Supervisors: Prof. Heather J. Ruskin, Dr. Martin Crane

September, 2011

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: Alina Sîrbu

_____

ID No: 58119850

Date: 21.09.2011

Contents

Abstract

Building models for gene regulation has been an important aim of Systems Biology over the past years, driven by the large amount of gene expression data that has become available. Models represent regulatory interactions between genes and transcription factors and can provide better understanding of biological processes, and means of simulating both natural and perturbed systems (e.g. those associated with disease). Gene regulatory network (GRN) quantitative modelling is still limited, however, due to data issues such as noise and restricted length of time series, typically used for GRN reverse engineering. These issues create an under-determination problem, with many models possibly fitting the data. However, large amounts of other types of biological data and knowledge are available, such as cross-platform measurements, knockout experiments, annotations, binding site affinities for transcription factors and so on. It has been postulated that integration of these can improve model quality obtained, by facilitating further filtering of possible models. However, integration is not straightforward, as the different types of data can provide contradictory information, and are intrinsically noisy, hence large scale integration has not been fully explored, to date.

Here, we present an integrative parallel framework for GRN modelling, which employs evolutionary computation and different types of data to enhance model inference. Integration is performed at different levels. (i) An analysis of cross-platform integration of time series microarray data, discussing the effects on the resulting models and exploring cross-platform normalisation techniques, is presented. This shows that time-course data integration is possible, and results in models more robust to noise and parameter perturbation, as

well as reduced noise over-fitting. (ii) Other types of measurements and knowledge, such as knock-out experiments, annotated transcription factors, binding site affinities and promoter sequences are integrated within the evolutionary framework to obtain more plausible GRN models. This is performed by customising initialisation, mutation and evaluation of candidate model solutions. The different data types are investigated and both *qualitative* and *quantitative* improvements are obtained. Results suggest that caution is needed in order to obtain improved models from combined data, and the case study presented here provides an example of how this can be achieved. Furthermore, (iii), RNA-seq data is studied in comparison to microarray experiments, to identify overlapping features and possibilities of integration within the framework. The extension of the framework to this data type is straightforward and qualitative improvements are obtained when combining predicted interactions from single-channel and RNA-seq datasets.

Acknowledgments

Firstly, I would like to offer special thanks to my supervisors, Prof. Heather J. Ruskin and Dr. Martin Crane for the continuous support they have provided during my PhD studies. They have guided my work and they have helped me develop my scientific personality, both by setting examples and by expertly directing me through the project milestones. I have learned a lot from them, and I am profoundly indebted. I would like to acknowledge my collaborators, Dr. Gráinne Kerr and Dr. Markus Kirkilionis, for useful discussions on modelling and data analysis topics, which have helped me round out the last part of the thesis. Further, I am grateful to my undergraduate Professor, Dr. Liviu Ciortuz, who has been the first to encourage me to start a PhD programme and get involved in research. He has introduced me to the fields of Machine Learning, Bioinformatics and Statistical Data Analysis, topics that I went back to many times during these years. I wish to recognize the financial support from the Irish Research Council for Science Engineering and Technology and that of the Irish Centre for High End Computing, for providing technical support and computational infrastructure, which have facilitated the development of this thesis.

The work presented here would not have been possible without the unconditional support of my parents and sister, to whom I dedicate the thesis. I am extremely grateful for the patience, trust and strength they have shown to me, while away from home. Further, I am grateful to all my colleagues and friends, for the many nice moments spent together, which have been invaluable during these years. Many thanks to Pierluigi who has put up with me towards the end of my studies, never letting my confidence go down and providing advice during the last efforts for finalising this work.

List of Abbreviations

AIC Akaike's Information Criterion

ANN Artificial Neural Network

Annot GO Annotations

AUPR Area Under Precision Recall curve

AUROC Area Under Receiver Operating Characteristic curve

BHI Biological Homogeneity Index

BP Back-Propagation

BPr Biological Process (Gene Ontology annotation)

BS Binding Site

cDNA Complementary DNA

CLGA Classical GA

CLGA Classical GA

Corr Correlation

DBI Davies-Bouldin Index

DBN Dynamic Bayesian Network

DE Differential Evolution

DEx Differential Expression

DKO Double Knockout

DNA DeoxyriboNucleic Acid

EA Evolutionary Algorithm

EC Evolutionary Computation

EGIA Evolutionary optimisation of GRNs - an Integrative Approach

EP Evolutionary Programming

ES Evolution Strategy

FPR False Positive Rate

GA Genetic Algorithm

GLSDC Genetic Local Search with distance independent Diversity Control

GO Gene Ontology

GP Genetic Programming

GRN Gene Regulatory Network

KO KnockOut

MC Markov Chain

MF Molecular Function (Gene Ontology annotation

MO Multi Objective

MOGA Multi Objective GA

MOGA Multi Objective GA

mRNA Messenger RNA

MSE Mean Squared Error

NGS Next Generation Sequencing

PEACE1 Predictor by Evolutionary Algorithms and Canonical Equations 1

PSWM Position Specific Weight Matrix

RMSE Root Mean Squared Error

RNA Ribonucleic acid

RPKM Reads Per Kilo-base per Million mapped reads

RSS Resudual Sum of Squares

SKO Single KO

TF Transcription Factor

TPR True Positive Rate

WOF Wheel Of Fortune

List of Figures

List of Tables

Part I

Introduction and background

Chapter 1

Introduction

1.1 Motivation

Correct functioning of living organisms is the result of collaboration and interdependency between several agents that work together to govern different processes. This starts from the genetic and molecular level, and continues to population, environmental and global level, with each of these levels affecting the others. Analysis of the complex system of interactions at all levels, as well as between them, is key to understanding the different aspects of life. At the molecular level, in particular, this collaboration occurs between genes, gene products and the environment, resulting in pathways, (e.g. signalling, regulatory or metabolic), which are crucial for natural processes [Heath and Kavraki, 2009]. Disruptions along these pathways can result in organism malfunction, i.e. in disease. Multifactorial genetic disorders, for instance, (e.g. cancer), are caused by genetic modifications (both innate and induced by the environment). These cascade in very complex processes, interfering with multiple pathways and leading to serious illness. These diseases are very common and an important cause of death both in humans and other species. In consequence, analysis of the molecular players and their interactions is extremely important in order to underpin disease markers and develop effective treatments [Tan et al., 2008].

Gene Regulatory Networks (GRNs) are an example of pathways that govern the correct functioning of the organism, by providing a mechanism to control protein levels in cells.

The agents involved are proteins, (i.e. transcription factors), and genes, (i.e. protein coding DNA sequences), which work together to control different processes. Considering the large number of genes present in the genome, these networks can be extremely difficult to analyse by human experts, so computational resources, tools and high throughput techniques for measuring gene activity have been developed [Hecker et al., 2009]. This has resulted in a vast amount of data, with which whole genome analysis of such pathways [Liang et al., 1998], using suitable computational tools, may be attempted.

1.2 Scope and contribution

The discovery of regulatory interactions can be performed at different levels, each exploring various data aspects and providing different types of knowledge of the GRN. A first step in GRN analysis is considered to be clustering of gene expression patterns [Thieffry, 1999]. Genes that belong to the same cluster are assumed to be co-regulated, (i.e. regulated by the same protein complex), or co-regulating, (i.e. regulating each other). Once clusters are generated, binding site motifs in the precursors of the genes in each cluster can be sought and hypotheses formulated on which proteins are involved in co-regulation, based on previous knowledge on the regulatory motifs.

While clustering is valuable, simulation of the gene expression process is important also and mathematical modelling is a powerful tool. Building a GRN model requires inference, (or reverse engineering), of parameters from available data (typically using a computational method, i.e. an inferential algorithm). The models can then be used for analysis and simulation in various contexts, which are often difficult to realise in laboratory experiments. Several approaches using mathematical modelling, ranging from qualitative (e.g. Boolean Networks, Rule Sets) to quantitative (e.g. Artificial Neural Networks, Differential Equation Systems), have been applied to discovery of GRNs. Simulation models have proved very useful to analyse some aspects of these complex systems. However, the size of GRNs and the nature of the data, (which are highly dimensional, noisy, and sometimes insufficient for analysis of GRN dynamics), limit robustness when mimicking natural behaviour. This is

particularly true for *quantitative* models, which aim to simulate very detailed patterns of expression, increasing the number of parameters to be inferred. However, such models could provide extremely useful insight on the gene expression process, and their improvement is an ongoing aim of Systems Biology [Przytycka et al., 2010].

Given the challenges posed by available gene expression data and poor model robustness to date, integration of several data types is a new direction for Systems Biology [Hecker et al., 2009]. In this thesis, a novel integrative inferential framework is presented, which permits data to be analysed at different stages. Included are (i) pre-processing and combining time-series expression data, (ii) use of other data types and knowledge and (iii) extension to next generation sequencing datasets. A novel inferential algorithm, based on Evolutionary Computation, is developed. Evolutionary Algorithms have been selected as they provide increased flexibility, implicit parallelism and have proved to be suitable search methods for underdetermined problems, noisy data and large search spaces [Baeck et al., 2000].

The strength of the newly-introduced platform is based on the number of data types to be combined and flexibility of integration. The customisation of different stages of the Evolutionary Algorithm enables more knowledgeable exploration of the search space and more informative evaluation criteria. Furthermore, a general methodology for GRN inference from multiple data types is developed. This includes an error structure analysis to identify at which stage of the algorithm each data type should be integrated.

The aim of this work is to enhance GRN inference, i.e. the reverse engineering algorithm, by introducing new criteria for evaluation and solution exploration. This, however, does not include development of novel mathematical models. The EGIA framework can be applied to any model in a relatively straightforward manner, by substituting different programming modules.

1.3 Thesis structure

Part I presents an introduction to GRN modelling and the state of the art in this field. Chapter 2 provides a description of the gene regulation process, while in Chapter 3, a literature

review of GRN modelling and the role of Evolutionary Computation is presented.

Part II introduces the novel integrative framework, consisting of three integration steps. The first step, (Chapter 4), analyses the integration of cross-platform microarray time series for GRN inference, with respect to normalisation choice and model impact. The second step consists of integrating additional data types and knowledge in the inferential process. Detailed information on the reverse engineering algorithm, under the name of EGIA (Evolutionary optimisation of GRNs - an Integrative Approach), is presented in Chapter 5. The performance of different novel elements, introduced by this framework, is analysed in Chapter 6, which studies the effect of additional data types on a *Drosophila melanogaster* test case. A third step towards integration is extension of the framework to next generation sequencing data, which are rapidly becoming available. A high-level analysis of microarray and RNA-seq data, aiming to identify overlapping features, is presented in Chapter 7, together with application of the EGIA framework to these data. Finally, Chapter 8 concludes the thesis and outlines future research directions.

Additional details on topics discussed are provided in the Appendices in Part III. Existing Evolutionary Algorithms for quantitative modelling are described in Appendix A, involving implementation and analysis of seven selected methods and discussion of their advantages and disadvantages. Information on the datasets used is provided in Appendix B and definitions of standard evaluation criteria in Appendix C. Finally, Appendix D includes a list of publications arising from this work.

Chapter 2

Biological Background

2.1 Gene Regulatory Networks

The functioning of living organisms, i.e. the coordination of the different processes involved, is governed by proteins working together. The information to create these molecules is encoded in the genetic material of the cell, in the DNA. The DNA is composed of two strands of *nucleotides* (ACGT), joined together in a double helical form by hydrogen bonds which can only appear between pairs A-T and C-G, and has both coding and non-coding regions. Genes are coding regions that contain the information for creating a protein, which is synthesised during the process of *gene expression*.

The *central dogma of molecular biology* describes the gene expression process as being composed of two stages (Figure 2.1): *transcription* and *translation* [Brown, 2002]. At the first stage, a copy of a coding region is created, resulting in messenger RNA (mRNA). RNA is a single-stranded sequence of nucleotides (ACGU), and can have different functions in the gene expression process. mRNA, in particular, is used as material to create proteins during the second stage of gene expression, translation.

The central dogma is a very simplified view of the true and complex gene expression process. Although the DNA is the same in all cells, different tissues display different behaviour, so cells clearly have other mechanisms to regulate gene expression levels. One such mechanism is *transcriptional regulation* and occurs during the initiation of transcrip-

Figure 2.1: Central dogma of molecular biology and transcriptional regulation

tion (Figure 2.1).

A particular class of proteins, namely *Transcription Factors* (TFs), act as *activators* or *repressors* for genes. This regulating activity is enabled by the binding of TFs to specific DNA regions that are close to the target gene, (typically upstream, i.e. before the gene in the sequence order). These regions are known as *promoter* regions or *cis-regulatory* modules. When an activator TF is bound to a promoter, the transcription rate of the associated gene is increased. Conversely, a bound repressor decreases the transcription level of the gene. The specific location where the protein binds is the *binding site*. The process of TF binding depends on the DNA sequence in the promoter region. Each TF has preference for specific DNA sequences: a *binding site affinity*.

Given that each TF is encoded by a corresponding gene, transcriptional regulation results in a complex network of interactions between genes and gene products. This is known as the *Gene Regulatory Network*, and is very important in controlling most natural processes [Lee and Tzou, 2009]. Figure 2.2 displays an example of such a network, with three genes and their corresponding gene products forming a regulatory circuit. Network visualisation can be simplified by removing the gene products, resulting in a graph with genes as vertices and regulatory effects as edges.

2.2 Measurement technologies

Several types of measurement can be performed at the level of transcriptional regulation, and advances in technology have enabled vast amounts of data to be generated, most of which are available in public databases. These data include gene expression measurements, either at the mRNA or protein level, assessment of binding site affinities for different tran-

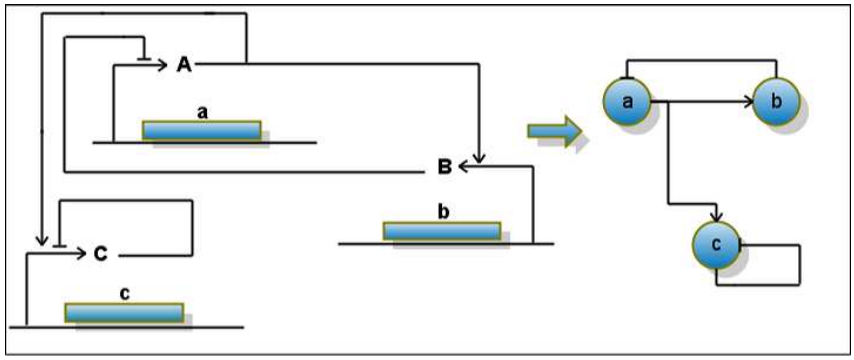


Figure 2.2: Gene regulatory network - an example

scription factors and identification of promoter regions for genes. In the following, we outline the different technologies and data types available.

2.2.1 Gene expression data

Gene expression measurements can be performed both at the mRNA and protein level, by measuring concentration of these molecules in the cell. Several different technologies are available, and these range from qPCR (quantitative polymerase chain reaction) [Logan et al., 2009] or in-situ hybridisation [Jin and Lloyd, 1997] to microarrays and RNA-seq. The former class allows for high-quality quantitative information to be extracted, but for a limited number of genes, while the latter yield more noisy measurement but at the genome level, i.e. including thousands of genes at the same time. In this work, we concentrate on high-throughput technologies, as these provide a data base for inference of large GRNs.

Microarrays [Quackenbush, 2001] are a mature technology based on *hybridisation* of cDNA, cRNA or ssDNA molecules onto a predefined array of complementary probes, where each probe corresponds to a specific transcript. The sample is labelled with a fluorescent dye and quantification of gene expression is performed by measuring the intensity of the dye on each probe. Two types of classical microarray platforms exist, *single-channel* (oligonucleotide, e.g. Affymetrix), where one sample is hybridised to one array [Lockhart et al., 1997], and *dual-channel* (cDNA) where two samples with different dyes are hybridised on the same array and relative expression levels can be retrieved [Schena et al., 1995]. Initially, the arrays contained probes for a set of known genes or transcripts; how-

ever, more recently, tiling arrays have appeared [Kapranov et al., 2002], which span the whole genome. These have the advantage of sampling DNA sequences not known to contain any genes at all, so that novel transcript discovery can be achieved.

Consequent upon microarray technology features, the resulting data are characterised by noise, which can be introduced at different stages [Baldi and Hatfield, 2002]. Firstly, unspecific hybridisation, i.e. hybridisation to probes that are not a perfect match, can affect the expression levels obtained. This, together with fluorescence from chemicals other than labelling dyes, forms the background noise. This can be estimated and subtracted from the expression levels. Secondly, different probes on the array, as well as different arrays, can have different specificity. Also, the various dyes used can introduce bias. Further, an image processing stage is required to obtain expression levels from dye intensity, which is also prone to errors. Additional noise can be introduced during the experiment, by variations in the sample preparation.

A different type of noise is natural noise. The gene expression process is stochastic in nature, and variation could have harmful effects. In consequence, regulatory networks and pathways have evolved to be resilient to natural variance and mechanisms such as autoregulation or existence of low transcript genes have been shown to be key in GRN robustness [Ozbudak et al., 2002]. However, natural variance also has beneficial effects, as it results in phenotypic variation within populations. One issue in gene expression data analysis is the fact the natural and experimental noise cannot be distinguished. Normalisation of these data can somewhat help, and has stimulated considerable research efforts Do and Choi [2006]; Lim et al. [2007]. Nevertheless, problems still exist with experiment reproducibility and integration of these data [Hurd and Nelson, 2009].

Good statistical analysis of gene expression data typically requires experimental replicates. These can be *biological*, when samples from different populations, but from the same environment and describing the same process, are measured. This allows for assessment of natural variability of the expression process, which needs to be taken into account when measuring variability between samples. Additionally, *technical* replicates can be obtained, which measure gene expression repeatedly for the same sample. This enables estimation

of variability due to experimental settings and other technology-specific biases. One type of technical replicate is the *dye-swap* replicate for dual-channel microarrays, which repeats an experiment by cross-labelling the two samples, to enable analysis of dye-based biases. Other technical replicates are used to control for probe- or array-specific bias. Typically, technical replicates display less variability than biological replicates.

Recent advances in high throughput sequencing technologies (*Next Generation Sequencing* - NGS) have introduced a new alternative to microarrays, namely RNA-seq [Mortazavi et al., 2008]. This quantifies gene expression by sequencing short strands of cDNA, aligning the sequences obtained back to the genome or transcriptome[1], and counting the aligned reads for each gene. This technology is expected to overcome some of the disadvantages of microarrays. For instance, it is able to identify transcripts that have not been previously annotated [Hurd and Nelson, 2009] and it can quantify both very low transcripts, (unlike microarrays where there is background noise interference) [Mortazavi et al., 2008], and very high ones (where microarrays suffer from *hybridisation saturation*, i.e. only a limited amount of cDNA can hybridise to a microarray spot) [Hurd and Nelson, 2009]. At the moment, although significant efforts have been made to modify algorithms and technologies, problems still exist with obtaining quantified transcription data. Some of these relate to read errors, short read mapping, SNPs, RNA splicing and sequencing depth, which particularly affect analysis of more complex transcriptomes [Mortazavi et al., 2008]. Additionally, the experimental cost for these technologies is still very high, compared to microarrays [Hurd and Nelson, 2009], while data handling is not straightforward, as large amounts of data result from each experiment, and this needs to be stored for further processing. Improvements are expected as the length of reads is increased and new algorithms and methods are developed [Hurd and Nelson, 2009]. Despite these initial issues, this technology is becoming increasingly used for gene expression quantification [Bullard et al., 2010] and datasets are becoming available. However, these are still scarce, in terms of number of replicates and time-course data.

Gene expression can be measured at different stages during a process. The most com-

[1]Ensemble of all RNA molecules produced through transcription in a particular organism.

Position	1	2	3	4	5	6	7	8
A	9	8	45	47	1	1	2	5
C	18	3	1	0	0	44	26	12
G	3	1	1	1	16	0	3	18
T	18	36	1	0	31	3	17	13

Table 2.1: PSWM example.

mon are *steady-state* measurements, i.e. sampled at equilibrium state. For the purpose of quantitative modelling of the expression process, however, *time-series* measurements are very important. These follow the change of gene expression with time, e.g. during a certain process or after perturbation (such as a mutation or treatment). Samples that are not perturbed in any way are known as *wild-type*. One type of applied perturbation can silence a gene or set of genes, and the resulting *knockout* (KO) expression data measure the changes caused by the perturbation. Typically, both knockout and wild-type experiments are performed, and the effects on the other genes can be expressed as log-ratios between the two, (which can be analysed using differential expression analysis).

2.2.2 Binding site affinities

Binding site affinities are important in assessing which TFs can bind to a specific DNA promoter sequence. For some organisms, the genome is known, so discovery of particular areas where a TF binds is possible. Chromatin immunoprecipitation (ChIP, [Collas, 2010]) has been widely used to study *in vivo* binding of proteins to DNA sequences. This enables isolation of the DNA regions where the specific protein binds, and has been combined with genome-wide technologies such as microarrays (ChIP-chip), [Buck and Lieb, 2004], or Next Generation Sequencing (ChIP-seq), [Jothi et al., 2008], to enable genome-wide identification of binding sites. *DNA footprinting* [Hampshire et al., 2007] can be also used to measure *in vitro* interactions between proteins and DNA.

One disadvantage of these technologies is that only analysis of one TF at a time is possible. This limits the availability of binding site affinity data, although this is expected to change as the technology becomes more advanced and sophisticated.

Footprinting and ChIP analyses identify a large number of positions in the DNA where

the TF binds, and indicate locations on the genome where promoter regions exist. These bound sequences can be combined to define a general pattern of the binding sites for the TF under analysis. This results in a Position-Specific Weight Matrix (PSWM) [Bergman et al., 2005], which can be used subsequently to compute an affinity measure to different sequences in the genome. The PSWM is composed of 4 rows, (nucleotides), and a number of columns representing the length of the pattern. On each column, a score is given to each nucleotide, showing how often that nucleotide has appeared at that specific position during the experiment, e.g. Table 2.1. Positions 3,4 and 6 in this example can be seen to be very specific, with one nucleotide having a very large score, while positions 1 and 8 allow more nucleotide possibilities, so are more flexible.

Given a new DNA sequence, represented as an array of characters of length n, and the PSWM, a $4 \times n$ matrix with rows indexed by the four nucleotides, the affinity score is computed as:

$$BS = \sum_{i=1}^{n} PSWM[DNA[i], i] \tag{2.1}$$

For instance, for the PSWM in Table 2.1, the DNA sequence GGTCAGGA would achieve a score of 14, while the sequence CTAATCCG would achieve the maximum possible score of 265. In Chapters 5 and 6, this method is used to compute affinities of TFs to regions upstream of genes or to known cis-regulatory modules.

Chapter 3

Modelling Background

3.1 Introduction

Uncovering interactions between genes and their products has been a major aim of Systems Biology over recent years, [Przytycka et al., 2010]. The objective is to gain a better understanding of the functioning of different organisms, as well as discovering disease markers and new treatments, [Bar-Joseph, 2004; Tan et al., 2008]. Gene regulatory network (GRN) analysis has been facilitated by the advent of technologies for measuring gene expression such as *microarrays* or, more recently, *RNA-Seq*. Characterised as they are by high dimensionality and noise levels, analysis of these data is far from trivial. The class of computational methods known as Evolutionary Algorithms, (EAs), has demonstrated relevance for different investigative targets, [Sîrbu et al., 2010a; Pal et al., 2006]. This chapter, in consequence, presents an overview of approaches and issues in GRN modelling and inference, and discusses the role of EAs in this regard[1].

Three different analysis stages can be identified for GRN inference: (i) expression pattern analysis, (ii) mathematical modelling from expression data and (iii) integrative modelling. At each of these, and most particularly at the last stage, EAs have an important role to play, due to the strength and flexibility of these search methods.

[1]This work has been published in [Sîrbu et al., 2011b].

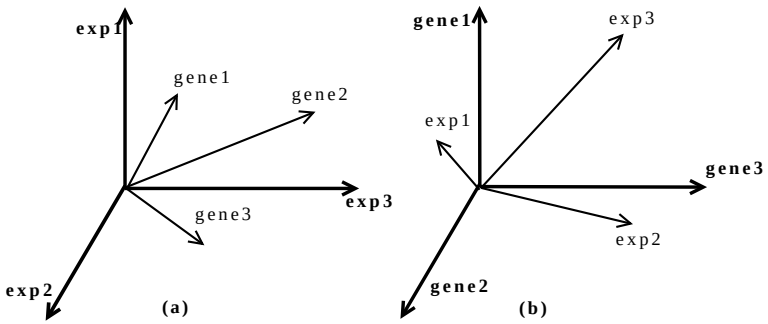


Figure 3.1: Gene expression data interpretations: (a) a set of vectors representing expression values for one gene under different experiments and (b) a set of vectors representing expression values for multiple genes under a single experiment

Expression pattern analysis is largely concerned with the application of classification and clustering methods to gene expression data. Gene expression data consists of the expression levels of many genes under multiple conditions, [Stekel, 2003]. Hence, for each gene, a vector of values shows the *gene expression pattern* for a number of different experiments, (Figure 3.1a). At the same time, the data can be viewed as a set of vectors describing the behaviour of the organism under certain conditions, (experiments), i.e. the *experimental patterns*, which represent expression values for many genes in a single experiment, (Figure 3.1b). By analysing both pattern types, (separately or together), useful knowledge related to the connections between genes or the similarity between conditions can be found. *Clustering* of gene patterns, as a first step towards GRN modelling, [Thieffry, 1999; Lee and Yang, 2008], together with sample *classification*[2], give valuable insight on gene involvement in different processes. EAs are typically employed at this stage, with some success, for *feature selection* as well as clustering.

A *second stage* in GRN inference is mathematical modelling using time series gene expression data. In these data, gene expression levels are measured over time, with each experiment in the data describing a different time point. These series patterns can be modelled using mathematical tools, of which a large number have been applied to GRNs, ([He et al., 2009; Lee and Tzou, 2009] and references therein). Models obtained can be used for

[2]usually for diagnostic purposes, to distinguish between tissue types, e.g. control/treatment or healthy/infected

in silico simulation and process analysis under various criteria. Generally, the process of modelling GRNs consists of a few main steps: choosing an appropriate model, inferring parameters from data, validating the model and conducting simulations of the GRN to predict its behaviour under different conditions. Due to the large number of genes in such datasets, clustering methods (stage one) have been applied by some authors for *dimensionality reduction* (either by considering cluster centroids as being one gene in the network, [Wahde and Hertz, 2000], or by analysing subsets of genes corresponding to selected clusters, [Lee and Yang, 2008]).

In order to model a GRN, genes are considered to be variables that change their values over time. Depending on variable type, methods can be classified as discrete or continuous, deterministic or stochastic, qualitative or quantitative, or as *hybrid* (using more than one type of variable). Approaches in the literature distinguish between *coarse-* and *fine-grained* models, [Lee and Tzou, 2009], with the former containing less detail on the interactions between genes. Usually, coarse-grained models use discrete variables, while fine grained models use continuous ones. However, a GRN can be very large and contain complicated interactions, so fine-graining carries its own penalties, such as an enormous number of parameters to deal with. Global analysis depends on the 'top-down' or coarse-grained approach to reduce complexity, [Maki et al., 2001; Repsilber et al., 2002; Linden and Bhaya, 2007]. Other authors, [Morishita et al., 2003; Wahde and Hertz, 2000; Kikuchi et al., 2003; Tominaga et al., 1999; Noman and Iba, 2006; Xu et al., 2007], have chosen to focus on detailed models, but for the analysis of sub-networks only of the entire GRN. Combining both levels of detail, by moving between the coarse and fine-grained model to highlight key biological knowledge is clearly useful [Maki et al., 2001; Kirkilionis et al., 2011a].

The ideal model for a GRN would be quantitative, accounting for all the features of the real GRN, applied to the entire genome in a cell. Achieving such a model is a non-trivial task, as most methods to date are either too coarse or can not model large systems [Lee and Tzou, 2009]. Also, existing gene expression time-series data are insufficient to infer the large number of parameters for such a detailed model, due to experimental cost and limitations. This leads to *under-determination* [Xu et al., 2004], also known as *the curse*

of dimensionality, especially when quantitative models need to be extracted (as these have an increased number of parameters). This means that multiple parameter sets are able to reproduce the behaviour seen in the data, and, in consequence, means of discriminating between these are necessary [Fomekong-Nanfack et al., 2009].

Consequently, a *third stage* in network inference, integrative analysis, [Hecker et al., 2009], aims at reconciling different data types and sources, in order to improve reliability of the inferential process, and model realism. This is not without risk, as multi-source data can contain heterogeneous noise. Further, large scale integrative analysis requires large computational resources and algorithms have to be optimised and parallelised to address this. Additional data types, which can contribute to this synthesis, include DNA-protein interactions, knock-out/knockdown experiments, binding site affinities, as well as known transcription factors (TFs) and RNA interference measures (Chapter 2). Integrative approaches have started to appear, [Hecker et al., 2009] and references therein, but are at an early stage only. Examples based on EAs are presented here. Typically, these combine only *one* additional data type with expression measurements, while, ideally, all related data should contribute to the inferential process.

3.2 GRN mathematical models

Boolean Networks Boolean networks are coarse-grained models for GRNs that use Boolean values for gene expression: the gene is on/off with values 1/0 respectively ([Liang et al., 1998]). Regulation is expressed in terms of Boolean functions attached to each gene :

$$Y_i = F_i(X_{i_1}, .., X_{i_k})$$

where $X_{i_1}, .., X_{i_k}$ are the binary expression levels of regulators of gene i and Y_i is the predicted expression value for gene i. This model is very well suited to modelling large networks, as it does not require a large number of parameters. Hence, Boolean networks have been employed in the analysis of steady state and general behaviour of GRNs. However,

disadvantages include inability to simulate continuous behaviour and complex nonlinear interactions, which characterise GRNs [Lee and Tzou, 2009]. Additionally, discretisation of expression values may lead to information loss.

A generalisation of the Boolean network is the multistate discrete network [Repsilber et al., 2002]. In this model, gene expression levels can take more than two discrete values (in the set $S = \{0, .., n\}$) and the regulation rules are general functions $F_i : \{0, .., n\}^k \rightarrow \{0, .., n\}$, mapping between current expression values for all genes and that of gene i at the next time point.

Rule Sets Another model of regulation uses different types of rules to explain the observed patterns in the data. This approach has the advantage of being more intuitive, as relationships between genes are expressed using natural language. One such model uses fuzzy rules, [Linden and Bhaya, 2007], based on fuzzy sets. These have imprecise boundaries, defined by a membership function: applied to any element in the universe, they return a number in the interval [0,1], representing the degree to which that element is a member of the current set. A fuzzy rule is a conditional of the form *if x is in A then y is in B*, which specifies a relation between fuzzy sets *A* and *B*. Every fuzzy rule also has a membership function that specifies the degree of truth of the implication.

Bayesian Networks Bayesian networks [Friedman et al., 2000] model gene expression as a joint probability distribution over a set of variables, each of these corresponding to one gene. They are represented as directed acyclic graphs, with an associated set of conditional probability distributions. The model adopts the Markov assumption that all variables are independent of the other variables, (except for their parents), given their parents in the graph. Thus, the joint probability can be decomposed as a product of conditional probabilities:

$$P(X) = \prod_i P(X_i|Pa(X_i)) \tag{3.1}$$

where X_i is the variable associated with the expression levels of gene i and $Pa(X_i)$ is the set of parents of gene i in the Bayesian network (i.e. those nodes that have outgoing

edges to gene i). Fitting such a model to data requires both the connection graph and the probability distribution parameters. Network variables can be discrete or continuous.

If we have a set of gene expression vectors $\mathcal{D}$, we can formulate the problem of inferring a Bayesian network as finding the model M with the *maximum posterior probability*, ($P(M|\mathcal{D})$). Using Bayes' rule ([Mitchell, 1997]):

$$P(M|\mathcal{D}) = \frac{P(\mathcal{D}|M)P(M)}{P(\mathcal{D})} \tag{3.2}$$

we can express the *posterior* probability of a model M using the probability of the data under that model, ($P(\mathcal{D}|M)$), the *prior* probability of the model, ($P(M)$), and the *prior* probability of the data, ($P(\mathcal{D})$). As $P(\mathcal{D})$ is the same for all possible Bayesian network models, we can eliminate this term from the computation when looking for the most probable model. The probability of the data given the model is computed in Equation 3.1 and the *prior* probability of the model has to be given. This can be equal for all models or it can promote preferred ones, based on known biological facts.

Advantages of Bayesian networks for gene expression data are stochasticity and scalability, [Kim et al., 2003]. However, a disadvantage is that they can not contain cycles, so can not model feedback loops, known to be crucial elements of GRNs.

Dynamic Bayesian Networks Dynamic Bayesian Networks (DBN) [Friedman et al., 1999; Kim et al., 2003] are able to model stochastic evolution of complex systems over time, by treating the value of gene X_i at time t as a random variable $X_i[t]$. The variables can be discrete or continuous. Letting $X[t] = \{X_1[t], .., X_n[t]\}$, the goal is to obtain the joint distribution over all genes at all times, $P(X[0], X[1], .., X[T])$. The modelled process is assumed to be Markovian, i.e. the expression values at moment $t + 1$ depend only on the expression values at time t

$$P(X[t+1]|X[0], .., X[t]) = P(X[t+1]|X[t]) \tag{3.3}$$

and stationary, i.e. $P(X[t+1]|X[t])$ does not depend on t. To specify the model one has to specify two components: a prior Bayesian network B_0 that represents a distribution over the initial states $X[0]$ and a transition network $B_{\rightarrow}$ over the variables $X[0] \cup X[1]$, which specifies the transition probabilities $P(X[t+1]|X[t])$.

The DBN adds the ability to model feedback loops to advantages of Bayesian networks. The method has been successfully applied to microarray data to build both discrete and continuous models [Kim et al., 2003].

Ordinary Differential Equations Most models described so far are coarse-grained. These use discrete states for gene expression values, with the influences of a given set of genes on other genes described qualitatively, rather than quantitatively. However, gene interactions are very complex and, in order to model these, a fine-grained continuous model is needed, which considers interactions quantitatively. One such model is a system of differential equations.

Ordinary Differential Equation systems express the change in the expression level of each gene in time as a function of the expression levels of other genes, but make no other assumption about the mathematical form:

$$\frac{dx_i}{dt} = F_i(x_1, .., x_n) \tag{3.4}$$

where x_i represents the expression level of gene i. The inferential algorithm, therefore, is not restricted to a prescribed set of functions and can model complex behaviour. At the same time, few constraints mean that the search space is very large and more sophisticated methods are typically required to refine the analysis.

Linear Differential Equations The simplest model example, and one that has received a lot of attention [Ando and Iba, 2003; Deng et al., 2005; Akutsu et al., 2000], is the linear system of differential equations. This retains the continuous aspects inherent in differential equations but results in loss of modelling power, as gene interactions are known to be more complex than linear dependencies imply.

This model describes changes in gene expression values as:

$$\frac{dx_i}{dt} = \sum_{j=1}^{n} w_{ij} x_j \tag{3.5}$$

where x_i and x_j represent expression values of genes i and j and w_{ij} the regulation *strength* of gene j on gene i. A negative value for w_{ij} corresponds to repression of gene i by gene j, a positive value corresponds to activation and a null value to no effect of gene j on gene i. Different versions of the model exist, which add other terms to the equation, accounting for external stimuli, degradation rates or noise [Yeung et al., 2002]. The system can be described by the matrix $\mathbf{W} = (w_{ij})$, also known as the *interaction or regulation matrix*. Inferring a model means finding the values in $\mathbf{W}$.

S-Systems Although linear systems include some detail, regulatory networks are intrinsically *nonlinear* systems and more sophisticated models of gene interactions are needed. S-Systems are a special type of differential equation systems, based on power-law formalism, and are capable of capturing complex dynamics. The disadvantages are an increase in the number of parameters and reduction in choice of reverse engineering techniques, as linear regression methods do not apply. The equations in S-Systems are of the form:

$$\frac{dx_i}{dt} = \alpha_i \prod_{j=1}^{n} x_j^{g_{ij}} - \beta_i \prod_{j=1}^{n} x_j^{h_{ij}} \tag{3.6}$$

The two terms correspond, respectively, to synthesis and degradation, influenced by other genes in the network; specifically, the *rate constants* α_i and β_i represent basal synthesis and degradation rate, while g_{ij} and h_{ij}, (*kinetic orders*), indicate the influence of gene j on the synthesis and degradation of the product of gene i.

Partial Differential Equations The differential equations models, presented so far, do not take into account the spatial distribution of cells and gene products. However, in certain situations, such as cell differentiation during development, spatial information is very important, so Partial Differential Equation systems are needed [Baldi and Hatfield, 2002].

These express concentration changes in both space and time, using reaction-diffusion equations. Here, the one-dimensional version of these equations is described, but these can be extended to 2- or 3-D situations. Considering a linear sequence of L compartments or cells, the concentration of product i in cell l depends on the regulatory effects in cell l but also on the diffusion process between this cell and its neighbours. Diffusion is considered to be proportional to the concentration difference between the two cells. So, the differential equation that describes this process is:

$$\frac{dx_i^{(l)}}{dt} = F_i(x_1^{(l)}, .., x_n^{(l)}) + D_i(x_i^{(l-1)} - 2x_i^{(l)} + x_i^{(l+1)}) \tag{3.7}$$

where F_i are the regulation functions and D_i are diffusion functions. This is for the case when space is discrete (well-delimited cells and compartments). In the continuous case, the concentrations of the products are functions of both time and space, so the system can be modelled with equations of the form

$$\frac{\partial X_i}{\partial t} = F_i(x_1, .., x_n) + D_i(\frac{\partial^2 x_i}{\partial s^2}) \tag{3.8}$$

where s is the space variable.

Artificial Neural Network (ANN) models ANNs are inspired by neural activity, and consist of interconnected neural units [Mitchell, 1997]. Each unit has a set of inputs and an output, and the output value is computed by applying an *activation* function (e.g. a sigmoid or a step function) to a weighted sum of inputs. The input weights are the model parameters 'learned' from data. ANNs are well-suited to model complex behaviour as they have been proved to be able to simulate any mathematical function, by adjusting the weights and topology [Cybenko, 1989]. A type of ANN that has been consistently used to model GRNs is the Recurrent Neural Network (RNN) [Wahde and Hertz, 2000; Vohradsky, 2001; Lee and Yang, 2008], which model dependencies between genes as:

$$\frac{dx_i}{dt} = m_i S(\sum_{j=1}^{n} w_{ij} x_j + b_i) - d_i x_i \tag{3.9}$$

where d_i is the degradation rate of gene product i, b_i accounts for external input, m_i is the maximum expression rate and x_j are expression levels while S is a sigmoid function. This model is similar to that of linear systems of differential equations, but introduction of the sigmoid function allows for modelling non-linear behaviour. A variant of this model considers discrete time points, [Keedwell and Narayanan, 2005], and computes the expression value of gene i at time point $t + 1$ using the values of the regulators at time t:

$$x_i(t) = S(\sum_{j=1}^{n} w_{ij}x_j + b_i - d_i x_i) \tag{3.10}$$

This reduces the computational cost for simulation, as no differential equations are involved, an important advantage in the context of evolutionary optimisation, (which requires simulation for every *fitness* evaluation - Section 3.3).

Multi-Scale Dynamic Modelling The models mentioned above are general mathematical models applied to the GRN problem, and analyse the different molecules involved on the same level, i.e. by making the same assumptions on all. Recently, a novel hybrid model of gene regulation has been introduced, [Kirkilionis et al., 2011a], especially tailored for GRNs. The so called 'macro-molecules' such as DNA or transcriptases, which appear in low copy-numbers, are assumed to have a finite number of discrete states modelled by a Markov Chain (MC). The state of the MC is determined by which binding sites are occupied and which are not. Transcription factors, on the other hand, are smaller molecules that appear in large copy-number and are described by continuous variables. In each state of the Markov chain, certain genes are expressed and the change in the concentration of transcription factors is described by a set of differential equations. At the same time, the transition between MC states depends on transcription factors, which bind with a specific rate to the gene promoters. These two levels continuously feed into one another, subject to the assumption that the time scale at the macroscopic level is longer than at microscopic level, and that MC state transitions are instantaneous. Using the invariant measure of the MC, a set of average dynamics can be derived as a system of differential equations. This approach

has been used to model a synthetic genetic clock that was engineered in *Escherichia coli*, [Kirkilionis et al., 2011b], where the number of binding sites involved and the interactions were previously known.

This modelling approach has the advantage of taking into account structural aspects of gene regulation, such as binding sites, as well as more complex mechanisms such as DNA looping and cooperative binding. However, due to the need for a complete description of the binding sites available for each gene, inference may require optimisation of this information as well. Nevertheless, reverse engineering can be enhanced by using previous knowledge of binding sites and affinities, providing a good base for data integration.

3.3 Evolutionary algorithms

EAs are a family of population-based optimisation algorithms inspired by Darwinian evolution, sharing a set of common features (see [Baeck et al., 2000] for a general description of EAs). Included are: Genetic Algorithm, (GA), Evolution Strategy, (ES), Genetic Programming, (GP), Evolutionary Programming, (EP), Differential Evolution, (DE). They perform an iterated search in the solution space, using information from previous iterations (*generations*) to guide exploration. The algorithm starts with a fixed-size population of candidate solutions to the optimisation problem, (also called *individuals* or *chromosomes*), which evolve over a number of generations. The value of each individual, i.e. its *fitness*, is given by a function defined for the specific optimisation problem. Evolution is performed using genetic operators that depend on the specific problem and encoding, e.g (i) *mutation*, which modifies one solution from the population, to obtain a new one and (ii) *crossover*, which uses several parents to create a number of offspring. Mutation is used to *explore* new areas of the solution space, while crossover *exploits* the information already gained in previous generations. The optimisation process needs to balance these two components, exploration and exploitation, to build an intelligent search strategy, with an emergent property being the ability to construct viable solutions for the problem at hand [Mitchell, 1999]. For each generation, a new set of solutions is produced from the previous population, either by replacing

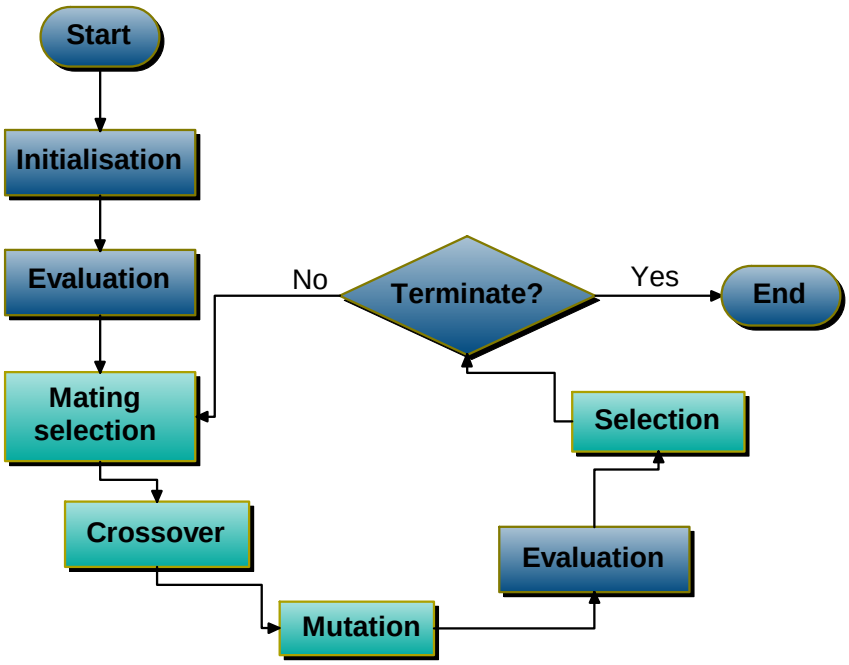


Figure 3.2: General schema of EAs.

selected parents by children, or by performing fitness-based selection on all parents and children. Figure 3.2 depicts the general algorithmic schema of EAs, with each individual component described in the rest of this section.

3.3.1 Representation

In order to be able to model candidate solutions, a computer representation needs to be derived. This defines the *genotype* of each EA individual, consisting of a set of *genes*, and a correspondence to the *phenotype*, i.e. the parameters to be optimised. In classical GAs, for instance, individuals are encoded as bit strings, and conversion to the corresponding parameters forming the solution is required [Mitchell, 1999]. For example, if the solution sought is an integer array (i.e. the phenotype), then the binary representation of these integers would be used as genotype, with each bit representing one gene. Other examples of representations are *permutations* (e.g. the Travelling Salesman Problem), or trees, (common in GP). However, over the years, new types of representations have been used, such as real-encoded individuals, which have been shown to be more suitable for multidimensional

continuous optimisation problems, [Mitchell, 1999]. In this case, the genes are the parameters to be optimised, removing the need to transform genotype into phenotype and reducing the computational power needed. Similarly, integer arrays can be used for representation of solutions, depending on the problem to be solved.

3.3.2 Initialisation

The initialisation phase defines the initial population of the algorithm, i.e. generation zero. This can be performed by assigning a value to each gene of the GA individual, chosen randomly from all possible values. However, if additional knowledge exists, the distribution for initial solution sampling can be non-uniform. Another option is to perform a heuristic search, (such as Hill Climbing), to derive a set of initial solutions.

3.3.3 Genetic operators

Mutation The exploration of the search space is performed by the EA using mutation. This is applied to each individual or gene with a certain probability and depends on the type of representation used. For instance, for *bit-string* representations, a mutation is a bit flip while for *permutations*, a mutation can swap the values on two random positions. For *real-encoded* EAs, mutations add a sample value to one gene (for a certain distribution, such as Gaussian). Mutations help to preserve diversity within the population of candidate solutions.

Crossover The crossover operator is applied to a number of individuals in the population to generate offspring. This results in different components of the parents being preserved in the offspring and aims at finding a better combination of these in the future generations. A typical crossover operator for individuals encoded as arrays, (of bits, integer or real numbers), is n-point uniform crossover, where two parents generate two children, by selecting n cutting points and randomly reconnecting the sections obtained. However, non-uniform crossover operators can be derived also, giving higher priority for example to sections corresponding to the fittest of the parents.

Selection Selection is performed to progress from one generation to another. In the general schema for EAs, selection can be performed at two stages: before and after crossover and mutation [Baeck et al., 2000]. The former, known as *mating selection*, allows only the most fit individuals in the population to generate offspring. These offspring can replace the parents, based on their fitness value or otherwise. Alternatively, the offspring can be incorporated in the population along with the parents. In this case, at the end of the generation, the second selection type is employed to reduce the population to the initial size. A popular mating selection is *tournament selection*, [Baeck et al., 2000], where a number of randomly selected individuals participate in a 'tournament', where the best wins the possibility to perform crossover. For population selection, (i.e. after crossover and mutation), an example of operator is the *wheel of fortune selection* (also known as *roulette wheel*) [Mitchell, 1999]. This assigns, to each individual in the population, a probability of selection in the next generation, proportional to its fitness. Individuals with higher fitness are assigned higher odds and the wheel is spun to select a number of individuals equal to the algorithm population size. This may result in multiple copies of the same individual being transferred to the next generation. The two types of selection, i.e. mating and population selection, can be combined, if necessary.

3.3.4 Evaluation

Evaluation of candidate solutions is very important as it influences which individuals are selected for mating and transferred to the next generation. It depends on the objective of the optimisation process (usually specified as a maximisation/minimisation problem). An *evaluation function*, applied to the phenotype of an individual, gives the quality of the solution. This results in a fitness landscape, with fitness values associated with all possible individuals in the search space. While defining an evaluation function can be straightforward in many cases (such as the travelling salesman problem, where the objective is to minimise the travel distance) this is not generally true. In most modelling problems, evaluation criteria need to be carefully derived, as, in these cases, the fitness landscape can be noisy, and can contain many local minima that could negatively influence the algorithm

performance. Selection of evaluation criteria depends on the information available for the system being modelled, so is often a challenging task.

3.3.5 Termination condition

Among the most popular termination conditions are the number of fitness evaluations performed, or the number of generations. This needs to be fixed at the start of the algorithm, and offers control over the running time. However, there is no guarantee of achieving a certain fitness value. Hence, some approaches use more advanced termination conditions, such as solutions with fitness larger than a threshold value, which controls the approximation error obtained at the end. However, the running time of the optimisation is not controlled beforehand. Of course, these criteria can be combined in order to obtain the optimal configuration.

Although these are common features of EAs (representation, genetic operators, selection procedures, etc.) they are also the elements that differentiate one type of EA from the others. For instance, individuals of GAs are typically encoded as binary arrays, DE and ES use arrays of real numbers as an encoding for the solution, while GP evolve tree-encoded expressions. At the same time, these methods use different genetic operators (applied to the different encodings) or use one *main* operator (for instance, an ES does not perform crossover but only mutation on its individuals). Even given strict differences between each individual in the EA family of methods, the distinction has become fuzzier with time [Mitchell, 1999], as new hybrid approaches have appeared, such as *real- or integer-encoded GA*.

EAs have been widely used for different optimisation problems, and examples include [Kita, 2011; Chambers, 2000; Baeck et al., 2000; Mitchell, 1999] and references therein. These range from the classical Travelling Salesman Problem or Prisoner's Dilemma, to more practical applications such as feature selection techniques for different classifiers, scheduling optimisation or modelling in environmental, social, physical or biological contexts. This wide usage has been triggered by advantages of EAs in working with underdetermined problems and noisy fitness functions, as well as flexibility and implicit parallelism.

However, from the theoretical point of view, EAs are not fully understood, although several theories have been applied in this direction (such as the Schema Theorem or Statistical Mechanics approaches [Mitchell, 1999]), and the good performance is still an empirical fact. This can be considered a disadvantage if a detailed mathematical description is required for the optimisation strategy. However, the empirically proven good performance, described as an emergent behaviour of a complex system [Mitchell, 1999], makes these algorithms a viable choice. An additional disadvantage of this algorithm type is the number of meta-parameters and operator choices that exist. Typically, the user defines values for these in an empirical manner; however, optimisation can be applied at the algorithm level to select the best meta-parameter set.

Here, the aim is to explore the power of these search methods for gene regulatory network reverse engineering. For this, EAs require a specified model type and data set. This enables parameter evolution to be monitored and performance to be evaluated. The fitness function is typically defined as the difference between the observed data and the output of the model, (squared, or averaged over the data points), as described in Equation 3.11.

$$\text{fitness} = \sum_{i=1}^{n} \sum_{t=1}^{T} (x_i(t) - y_i(t))^2 \tag{3.11}$$

where $x_i(t)$ is the expression value of gene i at time t, observed in real experiments, and $y_i(t)$ is the expression value of gene i at time t generated by the model. Since every model has its distinctive features, steps in the algorithm differ from one approach to another, but the main skeleton is usually preserved. Different EA approaches will be presented here, for inferences on *discrete qualitative* to *continuous quantitative* models. Consequently, the discussion includes *classical* to *hybrid* EAs and identification of strengths and weaknesses.

3.4 Stage 1 - Pattern Recognition

3.4.1 Clustering

Clustering is considered here as unsupervised[3] learning where a set of data entries has to be grouped into clusters, based on their attribute values [Manning and Schtze, 1999]. The clusters and cluster assignment for the training data set are typically not known beforehand, but are deduced based on dissimilarity or distance measures. Such measures may be statistical constructs, such as correlation, as well as standard spatial distance measures: Euclidean, Manhattan and others. Bi-clustering is also a variant that has been widely applied to gene expression data [Kerr et al., 2008]. This aims at grouping both genes and experiments at the same time, indicating not only clusters of co-expressed genes, but also in which experiments these appear.

GenClust [Di Gesu et al., 2005] is a novel method, using a GA-like approach. It differs from other GAs in that the population does not represent a set of possible solutions, but only one. Each individual in the population encodes one sample and a label representing its cluster (Figure 3.3). By analyzing these labels, the components of each cluster can be computed. The approach incorporates elements of EC, such as genetic operators, that are applied at each generation. Fitness evaluation is based on the sum of intra-cluster variances. The aim of the algorithm is to minimise this measure and, consequently, to obtain tight clusters. The method has been validated on five datasets (Rat Nervous System, Reduced Yeast Cell Cycle, Yeast Cell Cycle, Peripheral Blood Monocytes and Reduced Peripheral Blood Monocytes) and compared with other clustering methods like K-Means. The algorithm has been shown to converge rapidly to a local minimum; however, the resulting clusters were comparable those obtained by other techniques.

A similar objective was pursued by [Lu et al., 2004b,a], where a hybrid Genetic K-Means Algorithm (GKA) with two different versions (FGKA - Fast GKA and IGKA - Iterative GKA) was introduced. Hybridisation with K-Means consists of a *custom genetic*

[3]Recently, supervised clustering methods have emerged [Eick et al., 2004] addressing the need for more control over the meaning of the resulting clusters or the features that are considered by the unsupervised clustering technique. This section does not consider these.

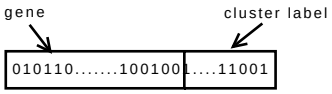


Figure 3.3: Chromosome representation in GenClust

operator, which changes cluster allocation to the closest centroid in random individuals. Neither FGKA nor IGKA uses a crossover operator, and mutation is performed based on dynamically computed probabilities depending on current cluster assignment. The difference between the two algorithms is that the latter updates cluster centroids and within-cluster variance each time a mutation is performed on *an individual*, while the former computes these for *each generation*. This makes IGKA faster when mutation probabilities are low, while FGKA is faster when these are large. In consequence, a hybridisation of the two (HGKA - hybrid GKA) is also proposed [Lu et al., 2004b]. The algorithms were applied to microarray yeast and serum data and IGKA was shown to obtain better clusters of genes from the same functional categories.

In [Chakraborty and Maka, 2005] a genetic Bi-Clustering algorithm based on K-Means and greedy local search seeding is presented. The algorithm was applied to yeast and human lymphoma data and was shown to provide better bi-clusters when validated against previous biological knowledge, compared to [Cheng and Church, 2000] (which adopts a greedy search[4] approach). A similar algorithm, that of [Mitra and Banka, 2006], employs multi-objective optimisation for Bi-Clustering. The algorithm is initialised using a greedy algorithm based on random initial solutions. Two objective functions are used, one maximising the number of genes and conditions in the bi-cluster, and another maximising homogeneity. Method evaluation was performed on the same yeast and human lymphoma datasets, and results indicate better performance compared to the single objective variant and to Simulated Annealing for Bi-Clustering [Bryan, 2005].

[4] A greedy algorithm is an optimisation algorithm that makes choices based on local optimum, with the aim of reaching the global optimum.

3.4.2 Feature selection techniques

A gene expression dataset can contain thousands of genes so that the elements to be clustered/classified are points in a high-dimensional space, hence analysis is computationally intensive [Sun et al., 2010]. Also, as data are intrinsically noisy, the high number of dimensions can bias algorithm convergence. It is possible that some features of the gene expression data are redundant [Cho and Won, 2003]; hence there is a need to develop feature selection techniques, in order to make analysis more efficient. Such methods select features (genes) that are important in the process under analysis, as they display a change in expression from one experiment to another. This aids sample classification.

Feature selection methods can be classified into two categories: Wrapper and Filter methods. *Filter* methods compute for each feature a measure of relevance for the current classification task. The features are sorted by their relevance and the top n are further used for pattern recognition. *Wrapper* methods, on the other hand, use the classifier itself to find the importance of a set of genes. They select a feature subset and train a chosen method on that set. The performance of the trained classifier can be seen as a measure of the relevance of the genes in the subset. The wrapper method iterates this operation for different subsets and chooses the best one. The difficulty is how to choose feature subsets that maximise the accuracy of the classifier, while minimising the number of selected genes and iterations. The search space for this problem is huge: if the number of initial genes is n, 2^n possible subsets exist. In this context, evolutionary techniques are known to cope well, as they benefit from mechanisms obtaining good solutions by searching a small portion only of the entire space, [Baeck et al., 2000]. Consequently, there are several approaches that use EAs as wrapper methods for feature selection, for example [Li, 2001; Ooi and Tan, 2003; Shah and Kusiak, 2004; Souza and Carvalho, 2005; Li et al., 2004]. One of these has also been applied to proteomics data, [Li, 2001; Li et al., 2004].

Most evolutionary approaches for wrapper methods are very similar. A population of gene subsets is maintained and allowed to evolve using different genetic operators. The fitness of each candidate solution is a measure based on the training error of the classifier

when using that specific set of features. After applying genetic operators, the fittest individuals remain in the next generation. While the principles are the same, existing methods differ in terms of classifier used or EA components, (e.g. size of the chromosomes, fitness function, etc). Additionally, some methods, [Li, 2001; Shah and Kusiak, 2004; Liu et al., 2009b], aggregate features obtained in multiple runs in order to improve performance. Table 3.1 summarises existing EA wrapper methods.

Method	**EA**	**Classifier**	**Multi-class**	**Feature set size**	**Combining results**	**Fitness**	**Datasets**
[Li, 2001; Li et al., 2004]	GA	K-NN	No	Fixed	Filter by appearance count	Classifier accuracy	Leukemia (microarray), Ovarian cancer (SELDI-TOF)
[Shah and Kusiak, 2004]	GA	Decision tree	No	Fixed	Reunion or intersection	Classifier accuracy	Emulated
[Ooi and Tan, 2003]	GA	Bayesian	Yes	Variable	None	Classifier error rate in cross- validation and independent test	9 cancer types, 14 cancer types
[Souza and Carvalho, 2005]	GA	SVM	Yes	Variable	None	Classifier error rate and feature set size	Leukemia, Blue-cell tumour
[Liu et al., 2009b]	GA	ICA-SVM, P-ICR	No	Variable	Intersection	LOOCV + feature set size	Colon cancer, High-grade glioma

Table 3.1: Features of EA wrapper methods. Abbreviations: ICA - Independent Component Analysis, [Liu et al., 2009b], GA- Genetic Algorithm, SVM - Support Vector Machine, K-NN - K - Nearest Neighbours, LOOCV - Leave One Out Cross Validation, P-ICR - Penalised Independent Component Regression

Recently, a new method for feature selection using genetic algorithms has been developed [Zhu et al., 2007]. This is a hybrid of the wrapper and filter methods: two operators

that add or remove features from a set, in a filter-like manner, are applied to the feature set encoded by the best individual of each generation. In this way, the individuals of the genetic algorithm, (candidate feature sets), are fine-tuned to improve the overall fitness and reduce the number of generations. The algorithm uses an SVM as a classifier. The algorithm was applied by the authors on 11 different datasets, including ones for lung or breast cancer, and compared to other filter and wrapper feature selection methods. It was shown to perform better than other methods for most datasets. Further, in [Zhu and Ong, 2007], another hybrid filter-wrapper GA-based feature selection algorithm is presented, which implements the new genetic operators using a ranking method, i.e. [Robnik-Łikonja and Kononenko, 2003]. This is shown to have similar results in terms of accuracy. However, the [Zhu et al., 2007] algorithm finds smaller gene sets, so it has an important advantage in terms of practical use: the smaller the number of features, the less expensive the diagnostic procedure. Further, [Zhu et al., 2007] has also been applied very recently to multi-class problems, [Zhu et al., 2010b,a], using multi-objective optimisation, (where each objective corresponds to the accuracy of a bi-classification task in a one-versus-all manner). Here, the notions of *full class relevant* and *partial class relevant* features are introduced. These, respectively, are features that have a role in differentiating all classes (i.e. display different expression levels in all classes) and features that differentiate only part of the classes (i.e. may have similar values is some classes). The algorithm identifies both types of features and is shown to perform better than [Zhu et al., 2007] on synthetic and microarray gene expression data.

3.5 Stage 2 - Model inference using time-series data

An overview of existing EAs for GRN model inference from time series data is presented in this section. The discussion considers methods applied to both discrete and continuous models. Due to the added complexity of the latter models, many EA approaches have been developed, from classical to advanced algorithms, and these are outlined, indicating their gradual development and their role in GRN inference.

3.5.1 Discrete models

Although applied more extensively for continuous models, EAs have been also used for qualitative model analysis. [Linden and Bhaya, 2007] introduced a method of inferring fuzzy rules from microarray data, using genetic programming. The algorithm uses the Reverse Polish Notation[5] for rules, which can be easily represented as trees, with three Boolean operators for the conditions: NOT, AND and OR. A population of this type was evolved using classical genetic operators on trees and the best individuals were selected to progress to the next generation. Fitness was defined as the percentage error observed between real data and the data generated by the rules. The algorithm was applied to finding rules in microarray data from experiments on the response to cold of the plant *Arabidopsis Thaliana*, as well as on the rat nervous system. A clustering algorithm was applied initially to reduce dimensionality, with resulting clusters considered to form one node in the network. Results were validated based on previous knowledge of the datasets, while new hypotheses for subsequent laboratory experimentation were proposed.

In [Repsilber et al., 2002] a Genetic Algorithm was used to fit a multistate discrete network, (Section 3.2), to simulated gene expression data. The aim was to rank previously known hypotheses about the structure of the network, by allowing model parameters to evolve. The approach also introduced time delays ($\delta = \{\delta_1, .., \delta_N\}$) to model the *time gap* between the transcription of one gene and the regulation effect of the resulting protein. This time gap is the time needed for the mRNA to be translated into a protein, so a node changes its state only after initiation of transcription *plus a time delay*. The algorithm thus searches for the most probable model structure for the data available.

Another method of inferring discrete GRN models, based on Genetic Programming, was developed by [Eriksson and Olsson, 2004]. Here, genes take Boolean values and the regulatory network structure for each target gene is encoded as a tree and evolved to obtain better structure. For each such tree in the population, a Boolean function is determined from data (by computing the truth table using expression levels in the data) and fitness is assigned

[5] In the Reverse Polish Notation, the symbol order in an expression is changed: the operators are in front of the operands. For instance, $a \times (b + c) - d$ becomes $- \times a + bcd$.

Method	EA	Model	Fitness	Local search	Stages	Datasets (Size)
[Sakamoto and Iba, 2001; Ando et al., 2002; Iba, 2008]	GP	ODE	Error + degree penalty	LMS	-	Synthetic (5)
[Fomekong-Nanfack et al., 2007]	ES	PDE	Error	-	-	Fly (6)
[Ando and Iba, 2003]	GA	LDE	Error	-	$\surd$	E. coli (9), Yeast (8)
[Deng et al., 2005]	GA	LDE	1 + Error	-	$\surd$	Rat 20)
[Tominaga et al., 1999]	GA	SS	Error	-	-	Synthetic (2)
[Iba and Mimura, 2002]	GA	SS	Error	-	$\surd$	Synthetic (10)
[Kikuchi et al., 2003]	GA	SS	Error + parameter penalty	Simplex Crossover	$\surd$	Synthetic (5)
[Kimura et al., 2003]	GA	SS	Error + parameter penalty	QP	$\surd$	Synthetic (30)
[Noman and Iba, 2005]	DE	SS	Error + parameter penalty	-	$\surd$	Synthetic (5)
[Noman and Iba, 2006, 2007]	DE	SS	Error + parameter penalty	HC	$\surd$	Synthetic (20), Yeast (14- qualitative)

Table 3.2: Evolutionary Algorithms for continuous model inference (1). *Error* is a measure of the difference between observed and simulated data, and different versions of this (RSS - Residual Sum of Squares, MSE - Mean Squared Error, Appendix C) have been used; however, their use is equivalent, as the number of genes and time points, (i.e. degrees of freedom), is the same for all individuals to be evaluated in a given optimisation run. Methods employing any type of iterated optimisation (Section 3.5.2.2), nested optimisation (Section 3.5.2.1) or *divide et impera* (Section 3.5.2.1) contain $\surd$ in column *Stages*. Abbreviations: ODE - Ordinary Differential Equations, EA - Evolutionary Algorithm, GP - Genetic Programming, LMS - Least Means Squares, PDE - Partial Differential Equations, LDE - Linear Differential Equations, SS - S-System, ES - Evolutionary Strategy, DE - Differential Evolution, HC - Hill Climbing, QP- Quadratic Programming.

based on ambiguities that arise. This results in choosing those structures that have fewer ambiguities, so indicate more plausible interactions. The method was tested on synthetic networks of different size, (10 to 160 genes), and shown, for networks smaller than 40 genes, to successfully locate structures with over 75% of the optimal fitness (with a value of 51% for the 160-gene network). However, to date, this method has not been validated with real data.

A similar EA optimising the *wiring of a Boolean network* is described in [Esmaeili and Jacob, 2009]. This method starts with randomly generated wirings with a limited number of regulators for each gene and evolves these structures using differential evolution. Each structure is evaluated using a multi-objective approach, which aims at optimising sensitivity, attractor cycle length and number of attractors. The method is shown to yield more stable structures for a synthetic network of size 8. However, as before, the method was not applied to real gene expression data, so further analysis is required.

3.5.2 Continuous models

Several algorithms for inference of continuous GRN models from gene expression data have been developed in recent years, and Tables 3.2 and 3.3 give an overview of methods. These include application of classical evolutionary techniques and development of novel algorithms, especially tailored for gene expression data. In this section, a general discussion on the benefits introduced by these methods is presented. A more detailed comparison of seven methods from the literature is presented in Appendix A.

One of the first approaches to GRN reverse engineering, based on Evolutionary Computation (EC), is due to [Tominaga et al., 1999]. A classic, double-encoded Genetic Algorithm is used to infer S-System models from time series data. However, this method was only applied to synthetic data for a very small network (2 genes). Another more recent application of a classic evolutionary algorithm is that of [Fomekong-Nanfack et al., 2007]. This employs an Evolutionary Strategy to optimise model parameters for a 6-gene developmental network for *Drosophila Melanogaster*, based on Partial Differential Equations (Section 3.2). Although suitable for a larger network than [Tominaga et al., 1999], the size of the

Method	EA	Model	Fitness	Local search	Stages	Datasets (Size)
[Xu et al., 2007]	DE PSO	RNN	Error	-	-	Synthetic (8), E. Coli(8)
[Koduru et al., 2004, 2005, 2007, 2008]	GA PSO	LDE, SS, RNN	Multi Objective - Error per gene	Simplex	-	Rice (2), A. Thaliana(3)
[Morishita et al., 2003; Ono et al., 2004; Imade et al., 2003, 2004]	GA	SS	Error	GA	√	Synthetic (5)
[Spieth et al., 2004, 2005c]	GA	SS	Error	ES	√	Synthetic (20)
[Keedwell and Narayanan, 2005]	GA	ANN	BP Error	BP	√	Synthetic Boolean (10), Rat (112), Yeast (2468)
[Daisuke and Horton, 2006]	GA	SS	Error	Simplex Crossover, Scale free	√	Synthetic (5), Mouse (7)
[Spieth et al., 2005b]	GA, ES	SS	Multi Objective - Error, Connectivity	-	-	Synthetic (5, 10)
[Kabir et al., 2010]	SA-DE	LTV	Error	-	-	Synthetic (5), E. Coli (6)

Table 3.3: Evolutionary Algorithms for continuous model inference (2). Abbreviations: LDE - Linear Differential Equations, SS - S-System, PSO - Particle Swarm Optimisation, RNN - Recurrent Neural Network, ES - Evolutionary Strategy, ANN - Artificial Neural Network, BP - Back-Propagation, SA-DE - Self Adaptive Differential Evolution, LTV - Linear Time-Variant.

inferred GRN is still very small compared to the total number of genes typically involved in such a system, showing that classical EAs are not powerful enough for larger networks. This will be discussed in more detail also in Appendix A.

The limitation in size and model quality for quantitative analysis derives from the nature of the data to be studied. A consequence of noise and under-determination is the *ruggedness* of the fitness landscape for this problem, [Rodrigo et al., 2010]. Hence, application of algorithms to real data is not straightforward. This is emphasised in Tables 3.2 and 3.3, where few algorithms have been applied to real data. However, EAs are known to perform well on under-determined problems and noisy fitness functions [Mitchell, 1999], so have clear benefit over other inferential methods. Evolutionary Computation approaches that address these issues can guide the optimisation towards more plausible solutions and are discussed next.

3.5.2.1 Addressing the under-determination problem

Divide et impera Given that model parameters are independent for each gene, (relying only on the expression level of the genes at previous time points), one method of addressing under-determination is to use a *divide et impera* approach. This consists of separate optimisation of parameters for each individual gene, using observed rather than simulated expression levels for the other genes. This method has been implemented in several EAs, [Ando and Iba, 2003; Iba and Mimura, 2002; Liu et al., 2008; Noman and Iba, 2006, 2007; Keedwell and Narayanan, 2005], and has the advantage of reducing the solution space by decreasing the number of parameters to be inferred at any one time. In Appendix A, a comparison study will show that algorithms using this approach scale better than those which attempt to optimise parameters for the entire network simultaneously. However, a typical disadvantage of this method is that expression levels in the data are noisy, and these are used in single gene simulations, resulting in model parameters slightly different from those that would be obtained by simulating all genes. This can be avoided by a second optimisation stage: starting from single gene models, evolutionary optimisation is employed to fine-tune the parameters for the entire network, [Ando and Iba, 2003; Noman and Iba, 2005]. Further,

a model that handles noise better, such as an Artificial Neural Network, (which employs a Sigmoid function to compute expression levels), may also decrease this effect.

Obtaining skeletal/scale-free structures To further distinguish between many possible models, known characteristics of the network structure, such as low connectivity or scale-free nature, have been considered also. Many methods apply such knowledge to the optimisation process at different levels of the algorithm. The simplest idea sets parameters to zero once these fall below a fixed threshold [Tominaga et al., 1999; Kikuchi et al., 2003]. However, more advanced approaches have also been developed. For instance, [Kikuchi et al., 2003; Kimura et al., 2003; Noman and Iba, 2005, 2006] use an additional term that penalises solutions with large parameter values. A refinement of this penalty-based idea can be seen in methods, which start by penalising all connections, and then use a connectivity threshold to reduce possibilities. This results in more advanced fitness functions, and is possible because evolutionary optimisation, unlike numerical methods, has the advantage of not restricting fitness function type. A similar method, [Spieth et al., 2005b], uses the connectivity as a second objective in multi-objective optimisation. Analogously, [Sakamoto and Iba, 2001; Ando et al., 2002; Iba, 2008] have used Genetic Programming to evolve sparse Ordinary Differential Equations, by penalising functions of large degree. [Deng et al., 2005] also employed a limit on the connectivity of a Linear Differential Equation model, evolving the connectivity parameter during optimisation, in order to find the optimal connectivity for the network, i.e. the number of regulators that achieves best data fit.

Another mechanism, used to obtain solutions with more plausible structures, is local search. For instance, [Noman and Iba, 2006] use Hill Climbing to set parameters to zero in two candidate solutions for each generation. Also, in [Daisuke and Horton, 2006], models are checked for scale-free structure, and modified if they do not comply, by adding or removing random connections, (setting the corresponding parameters to zero). All these methods result in sparser networks.

Nested Optimisation Reduction in the number of parameters to be optimised has been also performed using a Nested Optimisation approach, [Morishita et al., 2003; Spieth et al., 2004, 2005c; Keedwell and Narayanan, 2005]. These methods divide the search into two stages: structure and parameter search. During structure search, network topology is evolved using a genetic algorithm. Candidate structures are built so that the number of regulators is bounded for each gene, and these are evaluated by a second algorithmic stage, which optimises parameters for the existing connections. This reduces the number of real-valued parameters to be inferred at the second stage. The parameter search is performed using an Evolution Strategy, [Spieth et al., 2004, 2005c], a Genetic Algorithm, [Morishita et al., 2003], or Back-Propagation, ([Keedwell and Narayanan, 2005], with an ANN as the model). Again, this is facilitated by the flexibility of fitness evaluation, which is characteristic of EAs. Nested Optimisation increases parallelisation potential, (a parallelised version of [Morishita et al., 2003] was later developed by [Imade et al., 2003]). Separation of structure and parameter search is important as this allows the topology to have a larger influence in the optimisation process, rather than optimising real-valued parameters directly. This is particularly relevant in the current context as dynamical behaviour in biological networks relies mostly on *topology*, [Alvarez-Buylla et al., 2007], with parameter perturbations of lesser importance.

Parallelisation Quantitative models require optimisation of a very large number of parameters, and fitness evaluation is costly in simulation terms. These costs increase when additional time series datasets are used, so parallelisation of methods is mandatory. EAs have the advantage of being intrinsically parallel, facilitating efficient multi-threading of the optimisation process. Several examples of parallel implementations exist in evolutionary methods for GRN modelling, [Imade et al., 2004, 2003; Fomekong-Nanfack et al., 2007; Daisuke and Horton, 2006; Spieth et al., 2005a]. These correspond to both grid and cluster systems, while parallel frameworks for analysis have been implemented and are publicly available [Swain et al., 2005; Spieth et al., 2006].

3.5.2.2 Handling local minima

Combining multiple methods Due to the ruggedness of the fitness landscape, an EA can be trapped in local minima and fail to find an optimal solution. One way to avoid this is to combine different evolutionary methods. For instance, [Xu et al., 2007] alternates differential evolution and *Particle Swarm Optimisation*, (parameterisation of a Neural Network model), to obtain better overall models than from the two optimisation strategies separately. A different approach, [Kikuchi et al., 2003; Daisuke and Horton, 2006], uses *Simplex Crossover*, which efficiently balances the exploration and exploitation of the search space [Kikuchi et al., 2003] and in consequence speeds up convergence (as the local minima problem is diminished).

Iterated optimisation A second technique with the same aim is iterated optimisation, (possible due to the stochastic nature of Evolutionary Computation). Multiple runs of the same algorithm typically lead to different solutions, i.e. different local minima, which can be combined to obtain a better model: [Kikuchi et al., 2003] describe a second optimisation run, initialised with these local solutions. An alternative is to analyse local solutions by methods other than EAs. For instance, [Deng et al., 2005; Daisuke and Horton, 2006] employ a *voting procedure* for connections found in multiple runs, while [Noman and Iba, 2005] use voting to find null parameters in the model. Similarly, [Noman and Iba, 2007] apply Z-score analysis to local solutions to find plausible qualitative connections for yeast cell cycle data, (quantitative analysis being hampered by data limitations of length and noise).

3.5.2.3 Handling noise

Noise (both natural and experimental) is an intrinsic property of gene expression measurements and, unfortunately, most of the algorithms developed for model inference from these data do not specifically take it into account. This makes many methods unfit for real data, even when validated in principle for synthetic systems. Appendix A will include evaluations of method performance on noisy data [Sîrbu et al., 2010a]. While most methods

displayed good behaviour, up to 5% added noise, only two maintained this with up to 10%. One, [Keedwell and Narayanan, 2005], uses an ANN to model gene regulation, while the second employs a local search procedure, based on Quadratic Programming, that handles noisy measurements [Kimura et al., 2003]. The superior performance of these two methods is a strong indication that noise needs to be explicitly addressed in the model or evolutionary process, in order to obtain algorithms that can be applied to real-world data (Refer again to Tables 3.2 and 3.3).

3.6 Stage 3 - Integrating heterogeneous biological data

A first step towards integration is combining time-series data from different sources. Time-course data have been widely used for model reverse-engineering, and is very important in order to obtain quantitative models. However, most existing approaches use data from only one laboratory, i.e. the authors analyse their own experiments only. Multiple time series datasets from the *same* single channel platform have been used for linear model inference using singular value decomposition [Wang et al., 2006], but integrating gene expression data from *different platforms* has been formerly analysed only for *tissue (sample) classification*, [Cheng et al., 2009]. In the context of quantitative GRN modelling, however, this integration is not straightforward. Different analyses of normalisation techniques for multi-platform integration have been performed recently [Johnson et al., 2007; Shabalin et al., 2008], but again only for pattern recognition methods. Also, some normalisation techniques, specific to single and dual-channel microarrays, have been extended to be used for both types of microarrays, such as *Loess* or *dChip*, [Do and Choi, 2006]. An analysis of these in the context of GRN model inference from cross platform data is required.

A second step in data integration incorporates other types of data in the inferential process. It is widely recognised that integrative modelling approaches are required to enhance regulatory analysis, [Przytycka et al., 2010; Hecker et al., 2009; Alvarez-Buylla et al., 2007], and these have started to appear in recent years, mostly for coarse-grained analysis [Huttenhower et al., 2009; Kundaje et al., 2008] or based on Bayesian models, [Bernard and

Hartemink, 2005]. These integrate expression data with other types of measurements, such as binding affinities or protein interactions, to better discriminate between candidate models, but usually the integration is limited, (i.e. uses only one additional data type, besides time-course data, to enhance the inferential process). However, several such data-types are available, and the hypothesis is that combining all of these, i.e. large-scale integration, can further increase the modelling power. Recently, *Drosophila Melanogaster* datasets (RNA-seq, microarrays, ChIP-Seq, ChIP-chip) have been integrated, but again, for qualitative analysis only [modENCODE Consortium et al., 2010].

Evolutionary approaches for data integration are few, to date. One of the first methods attempting to incorporate previous knowledge, [Shin and Iba, 2003], used an AIC-based (Akaike's Information Criterion[6]) fitness function, (similar to that of [Noman and Iba, 2006]), modified to account for known interactions between genes in an S-System model. Thus models containing known interactions have better fitness and this leads the search towards regions in space that are more likely to contain the correct structure. The [Shin and Iba, 2003] algorithm was applied repeatedly and results were analysed using Z-scores to identify significant relationships. On synthetic data, the approach was shown to have increased sensitivity to finding correct interactions, compared to the standard method, which made no use of previous knowledge. Further, the former worked well even when previous knowledge was partially incorrect, (not unusual in a real experiment). When applied to the real microarray data of *E. Coli*, the method was also shown to identify previously known interactions.

A second data type integrated into the evolutionary optimisation process relates to knock-out experiments. [Ono et al., 2004] attempted inclusion of time series knock-out data, and demonstrated that this improved the structure search. Again, the method was only applied to synthetic systems. However, [Ferrazzi et al., 2007] integrated steady state knock-out measurements to infer parameters for a linear model of regulation in the cell cycle of *Saccharomyces cerevisiae*. The additional data are used to initialise a GA with biologi-

[6]Akaike's Information Criterion (AIC, [Noman and Iba, 2006]) is an information criterion used for model selection, which is based on the error between observed and simulated data.

cally plausible interactions, by analysing differentially-expressed genes in the knock-out experiments, and keeping these known interactions fixed in the structure. This enhanced approach was compared to a simple GA, and was shown to be more robust. Thus, feeding the optimisation with interactions from knock-out data guided the algorithm towards similar solutions in the search space during different runs, (implying that these were closer to the real network of interactions).

The increased availability of RNA-seq datasets has triggered efforts to extend analysis of gene expression to this data type. The postulated advantages of RNA-seq over microarrays (Chapter 3) may increase the power of GRN model inference, especially by integration of the two technologies. However, to date, integrative efforts are reduced to analyses of compatibility and complementarity of datasets with respect to general expression patterns, [Mortazavi et al., 2008; Fu et al., 2009], splice junctions, [Bradford et al., 2010], and differential expression, [Liu et al., 2011], and do not concentrate on model reverse engineering. Results from these studies show good correlation between microarray (including exon arrays) and RNA-seq expression levels, (reported Spearman rank and Pearson correlation values between 0.55 and 0.85, [Pickrell et al., 2010; Montgomery et al., 2010; Bradford et al., 2010]). However, RNA-seq experiments are reported to be more suitable than microarrays for quantifying absolute gene expression levels, when validated with mass spectrometry measurements, [Fu et al., 2009]. RNA-seq data have been shown to display more sensitivity to differential expression tests, compared to microarrays, with the number of identified genes generally larger, [Marioni et al., 2008; Bloom et al., 2009]. Additionally, the new platform seems to display better discrimination of differentially expressed genes with very large expression values (as expected based on technical specifications), while microarrays were reported better for very low transcript concentrations, [Bloom et al., 2009; Liu et al., 2011; Bottomly et al., 2011], (which is somewhat surprising given that NGS data have been postulated to have an advantage for low transcript quantification). For sample classification, [Cabanski et al., 2010] show that no significant difference between Agilent and Illumina technologies exists. These studies mostly concentrate on the same samples measured with the different technologies, in order to eliminate biases due to biological variability, which

allows for a more robust test of advantages and disadvantages of each platform. However, in the context of large scale integration, more heterogeneous datasets, from different sources and samples, should also be analysed and overlapping features identified in the more general setting. A detailed analysis of the gene space structure (i.e. clustering) is needed, as well as integration for GRN inference, which has not yet been performed to our knowledge.

3.7 Summary

This chapter has presented the role of Evolutionary Algorithms at different stages of gene regulatory network inference. These include (i) expression pattern analysis, (ii) model inference from time series data and (iii) data integration for model inference. For (i), methods for clustering and feature selection for gene expression data have been described. For (ii), method development from classical to more advanced hybrid algorithms has been presented. This has been motivated by issues in network modelling, such as under-determination and noisy data. These issues have been addressed to some extent by taking advantage of the flexibility and power of evolutionary approaches. For instance, the flexibility of the fitness function has been used to reward models with sparse or scale free structures. Hybridisation with local search and other optimisation algorithms has also benefited from the simple basis of the evolutionary algorithmic scheme, in order to avoid local minima traps and to handle noise. Additionally, the parallelisation potential of these methods, combined with their stochastic underpinning, has led to iterated algorithm versions, (designed to handle local solutions), and nested optimisation, (used to limit the number of real-valued parameters to be addressed). All these improvements have permitted a scale-up of quantitative modelling, from 2 to 30 genes, (Tables 3.2 and 3.3). However, this is still very modest compared to real GRN requirements.

Many of the methods presented have, to date, been applied only to synthetic data, (Tables 3.2 and 3.3), while most applications to real data can yield only qualitative results, as quantitative models obtained remain unreliable. In order to further improve inference, different data sources can be combined, and this has been presented as the third stage of GRN

inference. Advances in high-throughput technologies other than microarrays and global research efforts have created very large biological data sets containing protein-protein interactions, protein measurements, knock-out experiments, protein binding sites and gene sequence information. Although such data are currently insufficient to determine the underlying GRN, combining them could prove to be very powerful and EAs are flexible enough to enable their integration. However, such integration is not straightforward, as the different data types can negatively influence the modelling process, due to inherent noise and biases, so caution is needed. Existing methods, nevertheless, under-exploit EC potential, to some extent, by integrating only one additional data type. In consequence, in the following chapters, we will introduce a novel integrative framework, drawing on methods presented in Section 3.5.2. This aims at large-scale data integration for GRN quantitative modelling, using fitness evaluation, initialisation, mutation and parallelisation to include heterogeneous data in the optimisation process.

Part II

Integrative framework - description and results

Chapter 4

Step 1: Microarray Time Series Integration

In this chapter, we first present an analysis of expression data integration in the context of quantitative *GRN modelling*, using microarray time series datasets from different platforms (Section 4.2). The hypothesis being investigated is that using such heterogeneous datasets is possible and gives models which are more robust to data and parameter perturbations and capture essential dynamics in the data, without noise over-fitting[1].

Secondly, we assess the effect of cross-platform normalisation on the integration of the four datasets (Section 4.3). To date, cross-platform integration of microarray data has been analysed only for clustering and classification problems, using normalisation techniques to remove platform and batch effects [Johnson et al., 2007; Shabalin et al., 2008]. In the context of quantitative GRN modelling, this integration introduces new challenges, as different pre-processing techniques may impair data quality, e.g. by removing signal as well as noise. This leads to over-smoothing, resulting in significant loss of information, especially when multiple stages are involved, as for cross-platform normalisation. In consequence, correlations between interacting genes may be lost, or spurious correlations introduced dur-

[1]These results have been presented in ECCS 2010, and accepted for publication in the ECCS10 Special Issue Theory in Biosciences [Sîrbu et al., 2011a]. The final publication is available at springerlink.com. DOI 10.1007/s12064-011-0133-0.

ing pre-processing, making it very difficult for inferential algorithms to uncover the real structure of the GRN. Given that data are high-dimensional and complex, the resulting datasets are difficult to validate. Additionally, the data used for inference need to measure the same quantity, whereas the integration process may need to deal with log-ratios, log-values or other transformed quantities from the pre-processing stage. In consequence, two joint (single- and dual-channel) pre-processing approaches, based on existing normalisation techniques, are introduced to reconcile derived quantities and a comparison framework is built for assessment of results[2].

4.1 Datasets and inferential approach

Several inferential algorithms for regulatory network modelling exist in the literature, and, for this study, we have implemented and used one that has proven performance with real microarray data [Sîrbu et al., 2010a]. This is an Evolutionary Algorithm based on a hybridisation between Differential Evolution and Hill Climbing local search [Noman and Iba, 2006], described in Appendix A.1.7. The model used is the S-System (Chapter 3, Section 3.2). For the purpose of this chapter, the decoupled version is used, where model parameters for each gene are inferred separately, as opposed to determining parameter values for the whole system at once. Even though outcomes may be influenced by the inferential technique, the error between these simulations and the real signal seen in the test datasets is still a very good indication of how close the datasets are and, consequently, of how the integration strategy performs.

Integration analysis has been performed on four distinct datasets representing microarray time series measurements during the Yeast *Saccharomyces cerevisiae* cell cycle. The datasets, as described in Appendix B.2, are: Spellman, PramilaS, PramilaL and Hasse. Each of these analyse two cell cycles, at different time intervals. Combining these gave six time series measurements of the cell cycle, with a total of 111 time-points. For the first analysis below, the normalised data reported by the authors was used. For the second study, which

[2]This work has been published in PLoS ONE [Sîrbu et al., 2010b].

assesses the effect of different normalisation techniques on time-course data integration, we have drawn on the raw data from the same sources.

4.2 Model inference from cross-platform microarray datasets

The analysis was performed on a subset of 9 genes known to be involved in the cell cycle, retrieved from KEGG database [Aoki-Kinoshita and Kanehisa, 2007]. These were selected to form a sub-network that is poorly connected to the rest of the GRN, to facilitate separate analysis.

Despite initial normalisation by the authors of the four datasets used, data had different amplitudes. Consequently, a further scale normalisation was performed as follows. Firstly (i), each dataset was standardised (Equation 4.1). Secondly (ii), the values in all datasets were scaled to the interval $[0, 1]$ (by subtracting the minimum overall value and then dividing by the maximum value), since the S-System model requires positive values for gene expression levels. Additionally (iii), the time spans were modified to bring the cell cycle length to the same level, i.e. 120 minutes. This heavy pre-processing, involving so many stages, is mandatory for the integration to be possible. However, the risk of removing important features, introducing false correlations and influencing the resulting GRN model [Lim et al., 2007] is considerable. Nevertheless, this approach found good compatibility between datasets in our evaluation, and was necessary in order to perform an initial assessment of integration potential.

$$x' = \frac{x - \overline{x}}{s} \tag{4.1}$$

In order to analyse performance when moving from one to more datasets, we have split the four time series into two subsets: *inference (training)* and *test* datasets. The *inference* subset has been used during model inference, then models have been applied to simulate the *test* series. This bootstrapping approach has been used several times, resulting in thirteen experiments, each using a different combination of datasets for inference. Twenty runs have been performed for each experiment. All results presented in this section are based on

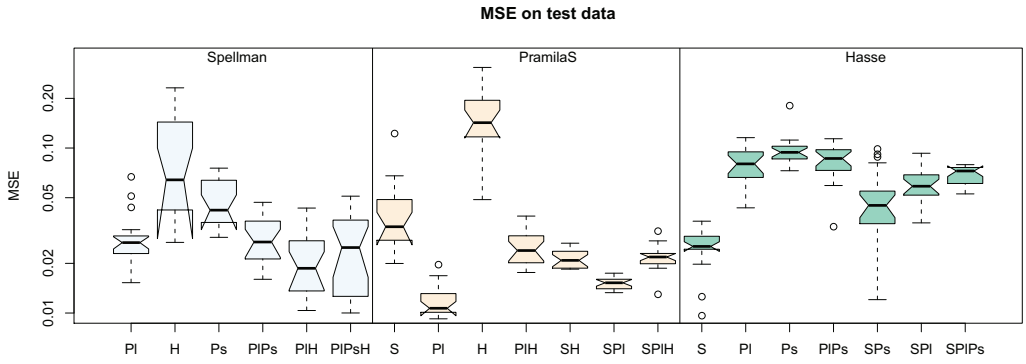


Figure 4.1: Performance of models on test datasets. Graphs are displayed as notched box-plots, (showing medians and quartile ranges of MSE values for 20 models for each experiment). On the x axis different experiments are represented, using one to three inference datasets as follows: S - Spellman, Pl - PramilaL, Ps - PramilaS, H - Hasse. The y axis is log-scale. The three boxes correspond to the datasets used for test, as labelled.

the models obtained for gene *CLN2*. This gene was chosen, being differentially expressed during different stages of the cell cycle. A repeat analysis, for gene *CLN1*, produced similar results (not shown).

4.2.1 Performance on Test Datasets

Figure 4.1 displays MSE (Mean Squared error, Appendix C) between simulated and real test data, for models obtained from different combinations of training datasets. Results for 20 optimisation runs are displayed as notched box-plots (Appendix C). These provide a representation of the distribution of MSE values in multiple runs, by showing medians and quartile ranges. Additionally, notches around medians define intervals that should not overlap in case of significant difference between medians (at the 5% level).

Figure 4.1 enables assessment of dataset compatibility and effect of integration on model performance on test data. This is expected to improve when integrating different datasets from a single platform, which is not, however, straightforward for cross-platform data. MSE values are generally low, indicating that datasets are compatible.

In using the Spellman dataset to test models inferred from PramilaL and Hasse together, Figure 4.1 shows MSE values to be, on average, lower than when using each dataset sepa-

rately. This indicates that even though the datasets are from a different source and platform, integration does enable capturing of more essential features by the resulting models. A similar behaviour is seen for models inferred from the Spellman and Hasse dataset, when validated with PramilaS data.

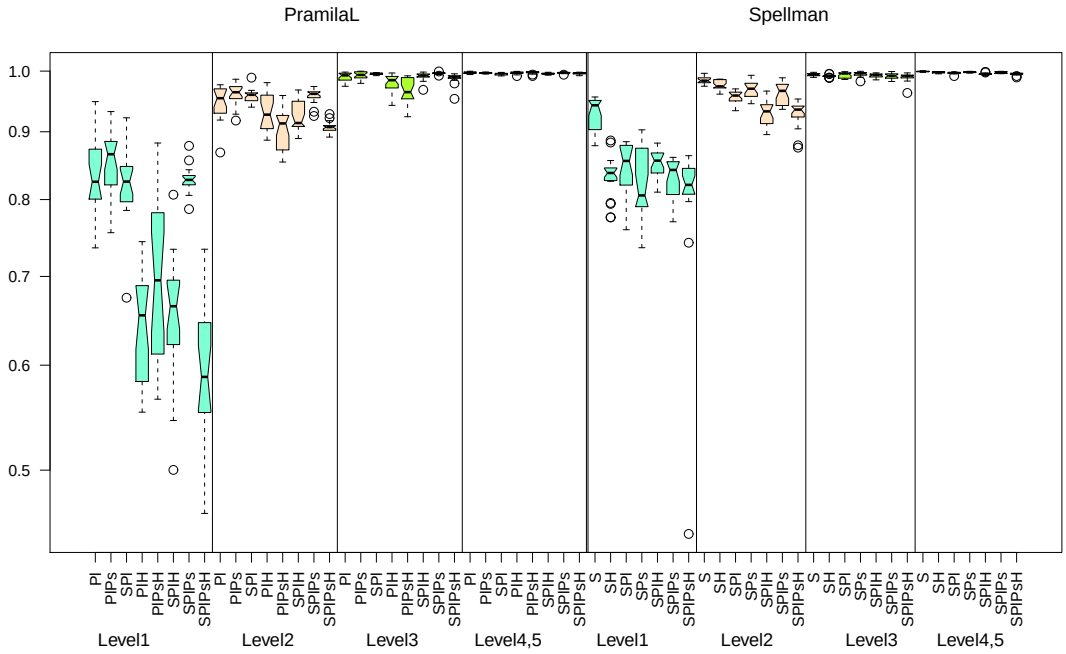
The lowest MSE on the PramilaS test dataset is obtained when the PramilaL dataset only is used for training. This is, probably, due to the fact that measurements for the two datasets are performed on the *same platform*, in the *same laboratory* and the time points overlap. This highlights the reproducibility of data from the same platform. When the two datasets are combined for inference (PlPs on horizontal axis), models obtained do not show decrease in MSE compared to each dataset individually, when tested on Spellman and Hasse. This shows that, under the MSE criterion, using such similar datasets for training does not lead to improved fit, as the data lie in the same space.

On the Hasse test dataset, (single-channel), models trained with Spellman data (dual-channel) achieve best MSE. When further adding dual-channel datasets (PramilaL and PramilaS) to inference, models do not become better in simulating the Hasse dataset. This indicates that by integration, more features characteristic of dual-channel data are modelled. This, together with the improved MSE when integrating the Hasse with the PramilaL and Spellman datasets above, supports the necessity for cross-platform data integration in order to be able to better extrapolate to test datasets, without over-fitting platform-related features.

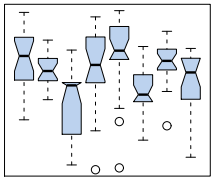
4.2.2 Wavelet Analysis of multiple time series

Naturally, integrating heterogeneous datasets decreases overall variability of models, as only the main features common to all datasets are incorporated. For this, variability between datasets with respect to main features has to be small enough to facilitate integration, while noise has to be heterogeneous. In this section, we use wavelets to show that the main features of the time series datasets are of comparable variability, and that models obtained from more datasets exhibit lower noise over-fitting.

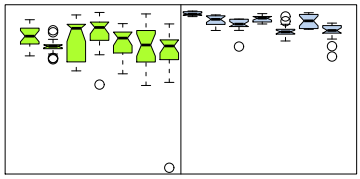
Wavelets [Kaiser, 1994], Appendix C, are a mathematical tool used for signal processing, which permit a simultaneous time and scale analysis of the signal. At *large scale*, i.e.



(a) PramilaL and Spellman all levels.



(b) PramilaL level 4,5 enlarged.



(c) Spellman levels 3 and 4,5 enlarged.

Figure 4.2: Correlation of wavelet coefficients for datasets PramilaL and Spellman at different levels. Image 4.2(a) shows all levels for both datasets, while 4.2(b) and 4.2(c) show enlarged images of three levels for which details are poorly visible in the main image, due to scale differences. The y axis is log-scale. When moving from one to more inference datasets, correlations at levels 1 and 2 (high frequencies) decrease, while those at levels 3 and 4 remain very high, indicating less noise over-fitting, while maintaining important features in the data.

Table 4.1: Correlation of wavelet coefficients for four gene expression time series (Spellman - S, PramilaS - Ps, PramilaL - Pl, Hasse - H). Level 1 corresponds to the smallest scale, i.e. highest frequencies, while level 4,5 to lowest frequencies.

	Level 1			Level 2			Level 3			Level 4,5		
	Ps	Pl	H	Ps	Pl	H	Ps	Pl	H	Ps	Pl	H
S	.582	.742	.087	.950	.890	.143	.907	.996	.844	.994	.998	.966
Ps		.283	.078		.902	.167		.939	.707		.991	.938
Pl			.156			.316			.828			.975

low frequencies, general features of the data can be analysed, while at *small scale*, i.e. *high frequencies*, more detailed aspects are investigated. In real world applications, *noise effects are high frequency*, thus visible at *small scale* in wavelet analysis, while much of the *signal is reflected in low frequencies*, i.e. *large scale*.

Gene expression time series measurements can be considered as signals; we thus performed a wavelet decomposition of the real signal (seen in the four datasets) corresponding to gene *CLN2* in the cell cycle GRN. For this, the signal was resampled using Spline Interpolation to generate 32 data points. This was necessary because the number of points required for the wavelet analysis needs to be a power of 2, with 2^5 just larger that the number of samples in each time course dataset. In this way we avoided under-sampling for all datasets and loss of information, as well as over-sampling, to reduce interpolation errors. The Haar [Kaiser, 1994] wavelet was used for decomposition and resulted in 32 coefficients at 5 scales (levels). The last two scales, 4 and 5, containing 4 coefficients, were combined and labelled as level 4,5 coefficients in this chapter.

Firstly, the Pearson Product-Moment correlation coefficient (r) [Boslaugh and Watters, 2008] was computed between wavelet coefficients from each dataset at each level, Table 4.1. Correlation values between coefficients, corresponding to pairs of signals, show that all four datasets are very similar at levels 3 and 4,5 (corresponding to low frequencies, i.e. real features), while at levels 1 and 2, (i.e. noise effects), correlations are low, indicating noise heterogeneity. The high correlation in essential features indicates that the four datasets are compatible for integration. For the Hasse dataset, low correlation, at levels 1 and 2, indicate differences with other datasets, which may be because of the different microarray

platform (one-channel vs. two-channel). At the same time, for the three dual-channel datasets, correlations are larger at levels 1 and 2 (high frequencies), indicating less noise heterogeneity.

To analyse noise over-fitting, wavelet coefficients for different model simulations were also computed. Models were trained starting with one dataset and then iterated by adding more datasets to the inferential process. For each model obtained, the coefficients at each level were compared to those in the initial training dataset, by computing r. The values obtained are displayed in Figure 4.2 for the Spellman and PramilaL datasets.

Results show that correlations at high frequencies, (i.e. noise effects), decrease when adding more time series to the inference process, while those corresponding to real features in the data (levels 3 and 4,5), are stable. This suggests that using heterogeneous time series reduces noise effects, while main features in the data are maintained. For instance, for the Spellman dataset alone, correlations are high at all frequency levels, indicating over-fitting of noise. However, adding other datasets to the inference shows decrease in high frequency level correlation to initial Spellman data, while low frequency effects are relatively unaffected.

4.2.3 Robustness to noise and parameter perturbation

Given the stochastic nature of the transcription process [Schlitt and Brazma, 2007], real GRNs are robust to relatively small changes in expression values, so quantitative models should be robust also to small data and parameter perturbations [Fomekong-Nanfack et al., 2009]. By integrating heterogeneous data, with different types of noise, more robust models should be obtained. This has been tested in this chapter by performing a sensitivity analysis for noise and parameter perturbation.

Firstly, models obtained in different runs were analysed on data containing Gaussian noise, which was added to the initial datasets, similar to [Noman and Iba, 2007]. Although the noise distribution is artificial, this analysis gives good indication of the model robustness. Noisy data were simulated by models, and MSE values obtained were compared to MSE in simulations of initial data, without added noise, (by computing ratios). Figure 4.3

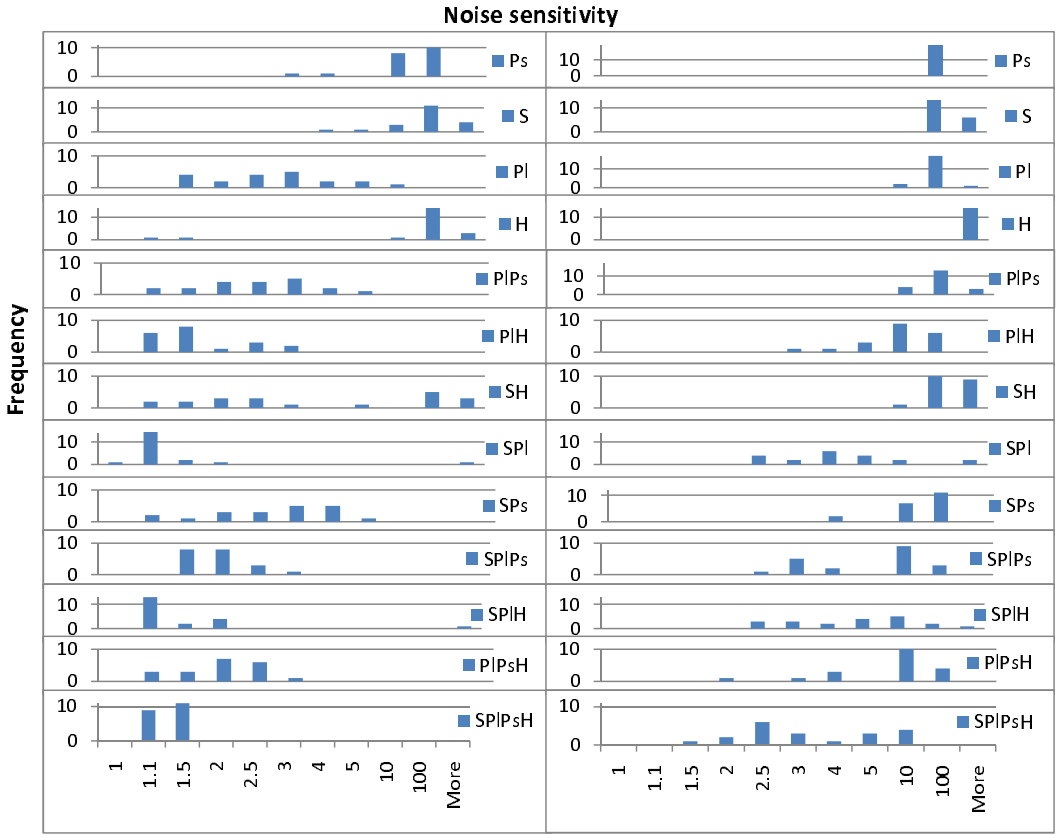


Figure 4.3: Sensitivity to noise. Histograms show MSE ratios over 20 models for different experiments, ranging from one (top) to four (bottom) inferential datasets, for two noise levels (standard deviation of added Gaussian noise 0.01 - left - and 0.05 - right). Ratios cluster to the left as moving from top to bottom for both noise levels.

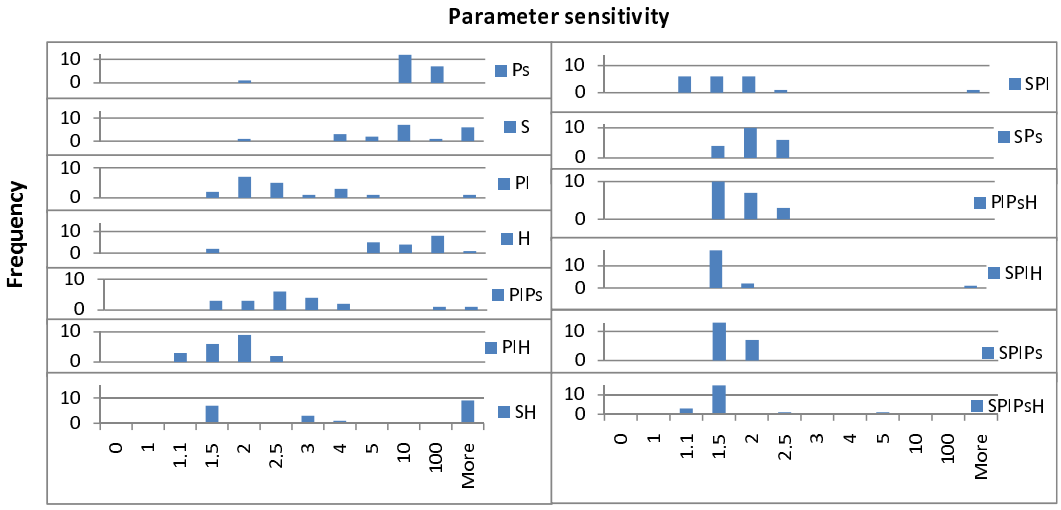


Figure 4.4: Sensitivity to parameter perturbations. Histograms of MSE ratios are plotted for 20 models obtained in each experiment. Ratios are smaller for models obtained from multiple datasets (bottom right), compared to those from one dataset only (top left).

plots histograms of MSE ratios for the 20 models in each experimental run, for two noise levels (standard deviations of 0.01 and 0.05). When using more inference datasets, ratios are closer to unity, indicating that models are more robust to data perturbations.

Secondly, a parameter sensitivity analysis was performed. For each model, individual parameters were slightly modified and the time series simulated and compared. The ratio of MSE (modified) to initial MSE was computed, analogous to the noise sensitivity analysis. Figure 4.4 shows histograms of ratios obtained for the 20 models in each experiment, with perturbations of $\pm 1\%$ of initial search interval, (0.2 in our case). Again, a clustering of the MSE ratios to the left, (lower values), can be seen when moving from experiments using one dataset to those using three or four datasets, indicating better resilience to parameter perturbations for the latter.

Robustness to perturbation is improved for any dataset combination, whether the constituent series are similar or not. However, this is markedly the case when combining dissimilar datasets. For example, the PramilaL and Hasse datasets are less similar than the PramilaL and PramilaS (as shown by wavelet analysis), resulting in models more robust to both noise and parameter perturbations for the former pair (Figures 4.3 and 4.4). Nevertheless, improvement is obtained for the latter pair too, compared to the individual datasets,

indicating that even when time series are very similar, integration does improve robustness to perturbations.

4.3 Cross-platform microarray data normalisation for GRN inference

This section presents the influence of different normalisation techniques on cross platform data integration for GRN model inference.

4.3.1 Normalisation techniques

Raw datasets from Section 4.2 have been used for cross-platform normalisation, where the common genes were extracted, resulting in 5337 for analysis. The normalisation performed consists of two stages. Initially, noise pre-processing was performed, using three different approaches. On the resulting datasets, three cross-platform normalisation techniques were applied, resulting in a total of nine normalised datasets for comparison. Additionally, the time spans were scaled so that the cell cycle length is the same across datasets.

Stage 1: Noise reduction within each dataset. Several normalisation techniques exist in the literature, especially tailored for single- and dual-channel arrays [Do and Choi, 2006]. However, these methods usually yield data of different type and scale, i.e. log ratios for dual-channel and 'absolute' expression values for single-channel, which are difficult to integrate in a qualitative model. In this context, three approaches (*W(i-iii)*), (one standard and two integrative), were used for within-dataset normalisation and compared for each dataset previously described.

W(i) PMLoess, applies different normalisation techniques depending on platform type: *PMOnly*, (available in the dChip software, [Li and Wong, 2001]) for Affymetrix, and *Loess* normalisation, (available in the Limma Bioconductor package, [Smyth and Speed, 2003]), for dual-channel data. *PMOnly* was chosen as a preferred method in previous studies, [Shakya et al., 2009], while *Loess* normalisation is an established method for pre-processing dual-channel arrays, [Do and Choi, 2006]. The logarithm of expression levels resulting from

dChip was computed for the Affymetrix dataset, to obtain semantics similar to log-ratios obtained after *Loess* normalisation for the dual-channel datasets.

Additional normalisation aims at reconciling use of both log ratios and log values by applying Loess normalisation to Affymetrix data and *PMOnly* normalisation to dual-channel data. These two methods are, henceforth, referred to as *W(ii) LoessOnly* and *W(iii) PMOnly*. *LoessOnly* applies Loess normalisation [Smyth and Speed, 2003] to both dual- and single-channel arrays, by considering the average of the perfect-match probes to be the red channel, and the mismatch probes to be the green channel, (where red and green correspond to the two samples compared in dual channel arrays). In dual-channel datasets (PramilaL, PramilaS and Spellman), the red channel corresponds to samples taken at the different time points during the cell cycle, and the green channel to a control sample, which is the same for all time points. In single-channel data, both perfect-match and mismatch probes correspond to the same sample, for which values are different at each time point. However, given that mismatch probes measure unspecific hybridisation[3], and that the amount of sample solution used in each experiment is the same, the mismatch signals should be close to one another at different time points. Thus, correspondence applies between the green channel in dual-channel time series and the mismatch probes in single-channel series. *PMOnly*, on the other hand, applies dChip to both types of data, taking the background-normalised red channel to be a perfect match probe.

Stage 2: For each dataset resulting from the first pre-processing stage, we applied cross-platform normalisation techniques, as follows (*X(i-iii)*). *X(i)* A simple standardisation on each dataset,

$$x' = \frac{x - \overline{x}}{s} \tag{4.2}$$

for data values x with sample mean $\overline{x}$ and sample standard deviation s was performed [Shabalin et al., 2008]. This was followed by a scaling of values to lie on the interval (0,1), which restricts the data to the same range. The scaling was performed by subtracting, from all values, the minimum expression level over all four datasets (plus a predefined δ),

[3] genes can hybridise even if their sequence is not the correct complement of the probe

followed by dividing all values by the maximum (plus δ):

$$x' = \frac{x - x_{min} + \delta}{x_{max} - x_{min} + \delta} \tag{4.3}$$

where δ here is used to ensure that the limits of the interval (0,1) are not reached. This scaling was necessary as the S-System model used here requires positive expression values for all genes. Further, *X(ii) ComBat* [Johnson et al., 2007], a Bayesian technique aimed at removing batch effects, and *X(iii) XPN* [Shabalin et al., 2008], a technique based on iterative K-Means Clustering, were also applied for cross-platform normalisation. Additionally, scaling onto the interval (0,1), was performed, as noted. All these techniques aim at standardising data *across platforms*, after a preliminary normalisation within each dataset. The implementations, made available by the authors, were used for the latter two methods. The final datasets are identified in this chapter by the name of the normalisation techniques used for each stage: *PMLoess* methods (*PMLoess_St, PMLoess_ComBat, PMLoess_XPN*), *PMOnly* methods (*PM_St, PM_ComBat, PM_XPN*) and *LoessOnly* methods (*Loess_St, Loess_ComBat, Loess_XPN*). The rest of this section briefly describes the cross-platform normalisation procedures *ComBat* and *XPN*.

ComBat [Johnson et al., 2007] is a normalisation method for eliminating batch effects, which models the gene expression level for gene g in experiment i and platform j as:

$$x_{gij} = \alpha_g + X\beta_g + \gamma_{ig} + \delta_{ig}\varepsilon_{ijg} \tag{4.4}$$

with α_g the overall expression level, X a design matrix for experiment conditions, β_g the vector of regression coefficients for X, γ_{ig} and δ_{ig} the batch effects, and ε_{ijg} the noise term (Normally distributed with zero mean and σ_g variance).

The method consists of three steps. (a) The data are standardised to obtain similar overall mean and variance for genes. This involves fitting of parameters α_g, β_g and γ_{ig} by using a least-squares approach, estimation of σ_g, and computation of a standardised data

point as:

$$z_{gij} = \frac{x_{gij} - \hat{\alpha}_g - X\hat{\beta}_g}{\hat{\sigma}_g} \tag{4.5}$$

where $\hat{\alpha}_g$, $\hat{\beta}_g$ and $\hat{\sigma}_g$ are the estimated α_g, β_g and σ_g. Further (b), the batch effect parameters are estimated, using the assumptions that γ_{ig} are Normally distributed ($N(x_i, \tau_i^2)$) while δ_{ig}^2 follow the $Inverse\ Gamma(\lambda_i, \theta_i)$[4] distribution. The parameters for these distributions are estimated using the method of moments. Finally (c), the data are adjusted for batch effects:

$$x^*_{gij} = \frac{\hat{\sigma}_g}{\hat{\delta}_{ig}}(z_{gij} - \hat{\gamma}_{ig}) + \hat{\alpha}_g + X\hat{\beta}_g \tag{4.6}$$

XPN [Shabalin et al., 2008] is a cross-platform normalisation procedure based on the assumption that subsets of genes have the same pattern in subsets of experiments. The expression level for a gene g in sample s and platform p is considered to be a *block mean*, A_{GSp}, which is the same for a subset of samples (S) and genes (G), and common across platforms (p), transformed by a scaling and a shifting factor, i.e. b_{gp} and c_{gp}, specific to each gene (g) and platform (p), and a noise term ε_{gsp}, (specific to each gene, sample and platform):

$$x_{gsp} = A_{GSp}b_{gp} + c_{gp} + \sigma_{gp}\varepsilon_{gsp} \tag{4.7}$$

In order to find G and S, i.e. the groups of genes and samples where the block mean values apply, K-means clustering is applied separately on sample and gene patterns obtained by combining the datasets to be normalised. Based on cluster assignment, the model described in Equation 4.7 is fitted to the data, using a maximum likelihood method. Normalised expression values are computed based on the model obtained:

$$x^*_{gsp} = \hat{A}_{GS}\hat{b}_g + \hat{c}_g + \hat{\sigma}_g\frac{x_{gsp} - \hat{A}_{GSp}\hat{b}_{gp} - \hat{c}_{gp}}{\hat{\sigma}_{gp}} \tag{4.8}$$

where $\hat{A}_{GS}$, $\hat{b}_g$, $\hat{c}_g$ and $\hat{\sigma}_g$ are weighted averages of parameters $\hat{A}_{GSp}$, $\hat{b}_{gp}$, $\hat{c}_{gp}$ and $\hat{\sigma}_{gp}$, obtained for each platform. The procedure is iterated 30 times to obtain 30 normalised values,

[4] $Inverse\ Gamma(x; \lambda, \theta) = \frac{\theta^\lambda}{\Gamma(\lambda)}(x)^{-\lambda-1}\exp(-\frac{\theta}{x})$, where Γ is the *Gamma* function, $\Gamma(x) = (x-1)!$.

corresponding to different cluster assignments and final expression values are computed as the average of the values obtained in each run.

4.3.2 Evaluation criteria for normalisation methods

Evaluation of normalisation methods applied has been carried out using four different criteria (*E(i-iv)*). Firstly, *E(i)*, variability between replicates has been computed, as the average over all genes of the RMSE (Root Mean Squared Error, Appendix C) between replicate expression values, normalised by the average gene expression level for each gene, (Equation 4.9).

$$var = \frac{1}{N}\sum_{i=1}^{N}\frac{\sqrt{\frac{1}{T}\sum_{j=1}^{T}(x_{ij_1}-x_{ij_2})^2}}{\overline{x_i}} \tag{4.9}$$

Here, x_{ij_k} represents the expression level of gene i in experiment j and replicate k, N is the total number of genes and T is the total number of experiments. The datasets contain a dye-swap replicate for one dual-channel dataset (PramilaL) and one technical replicate for the single-channel (Hasse) dataset. This allows for a comparison on both replicate types. Ideally, after normalisation, replicates should be approximately the same, so the distance between them is a criterion widely used for validation of normalisation techniques, [Shakya et al., 2009].

Secondly, *E(ii)*, wavelet analysis was used to compare the normalisation techniques, using the Daubechies discrete wavelet transform [Kaiser, 1994], similarly to the previous study (Section 4.2). The *average absolute value* of the high frequency coefficients, corresponding to 9 genes known to be involved in the cell cycle, (from KEGG database [Aoki-Kinoshita and Kanehisa, 2007]), was computed, as these components are a good indication of the magnitude of noise in the data. Also, wavelet coefficients for gene signals from different datasets were compared at different scales, (by computing *RSS values*, Residual Squared Error, Appendix C), in order to assess which normalisation techniques bring the data closer together.

Thirdly, *E(iii)*, a correlation analysis was performed to test whether pair-wise gene correlations vary between normalisation techniques, as well as to determine whether genes known to interact are correlated after normalisation. Ideally, normalisation should remove spurious or noise effects, while real gene correlations should be preserved, which is very important for GRN model inference [Xulvi-Brunet and Li, 2010]. The Pearson correlation coefficient (r_{ij}) was computed between all gene pairs (i, j) and three aggregation criteria were used for analysis.

Aggregated criterion 1: The number of gene pairs with absolute correlation larger than 0.9 was computed and compared across normalised datasets, to determine whether normalisation techniques affect high correlation values. Aggregated criterion 2: average of absolute correlations for each gene i were calculated as shown in Equation 4.10.

$$avg_i = \frac{1}{n} \sum_{j \neq i} |r_{ij}| \tag{4.10}$$

where n is the number of gene pairs in each dataset. These values give a measure of how the gene relates to the rest of the system, within each dataset. Aggregated criterion 3: the correlation variability between microarray datasets (Spellman, Hasse, PramilaL, PramilaS), for each normalisation technique, was computed for each gene pair, as indicated in Equation 4.11. Ideally, the same pair of genes should have similar correlation across microarray datasets, but, due to platform differences and normalisation, these can vary. Correlations common to the different datasets are most reliable, while others are more likely to be spurious.

$$var_{ij} = \frac{1}{\sqrt{6}} \left[\sum_{a,b} (r_{ij}^a - r_{ij}^b)^2 \right]^{\frac{1}{2}} \tag{4.11}$$

where $a, b \in$ S,Pl,Ps,H and $a \neq b$, r_{ij}^d represents the Pearson coefficient between genes i and j in dataset d, with d having values S (Spellman), Pl (PramilaL), Ps (PramilaS) and H (Hasse). This results in a matrix, for each normalisation technique, referred to as *correlation variability matrix* in the rest of this chapter, which shows how correlations between pairs of genes differ from one microarray dataset to another. This can be viewed as an indicator of

the amount of spurious correlation or of increased variability in each normalised dataset. We use the *average* of the values in the correlation variability matrix to quantify the correlation similarity (between datasets) for each normalisation technique.

A different method of identifying spurious correlations would be to analyse partial correlation coefficients in the data. These higher-order correlations, (as opposed to zero-order coefficients such as Pearson), account for other genes (variables) in the data, rather than considering each pair in isolation. However, this is non-trivial, as the pattern of covariance is very complex, with many gene pairs having high zero-order correlation and circuits known to exist in the networks. Hence our use of the correlation variability matrix, described above, as a (weaker) criterion.

The fourth evaluation criterion used, *E(iv)*, was the capability of single gene models to translate between datasets. For this, models were built from each dataset individually, and then applied to simulate the same genes in the other three datasets. S-System models of regulation for two genes (*CLN1*, *CLN2*), in a 9-gene network, were developed, similar to the previous Section (4.2). Twenty runs were performed for each inference task, and RMSE values, normalised by the mean expression values (RMSE/Mean)[5], were averaged across these. Additionally, models have been inferred from combining two datasets and testing on a third, to analyse how the data fit changes compared to using each training dataset individually.

4.3.3 Results

Table 4.2 summarises overall values obtained for different cross- and within-platform normalisation, for the criteria above, to support the discussion of results.

	Cross-platform						Within-platform					
	Standardisation		ComBat		XPN		PMOnly		Loess Only		PMLoess	
	SC	DC	SC	DC	SC	DC	SC	DC	SC	DC	SC	DC
Variability between replicates	↑ with PML; ↓ with PM & L	↑ with PM & PML; ↓ with L	↑ with PM & L; ↓ with PML	↑ with L; ↓ with PM & PML	↑ with PM & L; ↓ with PML	↓	↑	↑	↓	↓	↓	↓
Amplitude of noise frequencies	↑ with PM & PML; ↓ with L		↑ with L; ↓ with PM & PML		↑ with L; ↓ with PM & PML		↑		↓		↓	
Number of highly correlated genes	↑	↑	↑	↑	↓	↓	↑	↑ in Pl & Ps; ↓ in S	↓	↓	↑	↑ in S; ↓ in Ps & Pl
Average absolute correlation	↑	↑	↑	↑	↓	↓	↑	↓	↓	↑	↑	↑

Table 4.2: Summary of variability and aggregated correlation values for different within- and cross-platform normalisation. SC and DC identify results for single- and dual-channel datasets; Pl, Ps and S represent the three dual-channel datasets (PramilaL, PramilaS and Spellman), while PM, PML, L stand for *PMOnly*, *PMLoess* and *LoessOnly*, respectively. Arrows indicate whether variability and correlations are increased (↑) or decreased (↓) relative to the other normalisation procedures in the same category (cross- or within-platform).

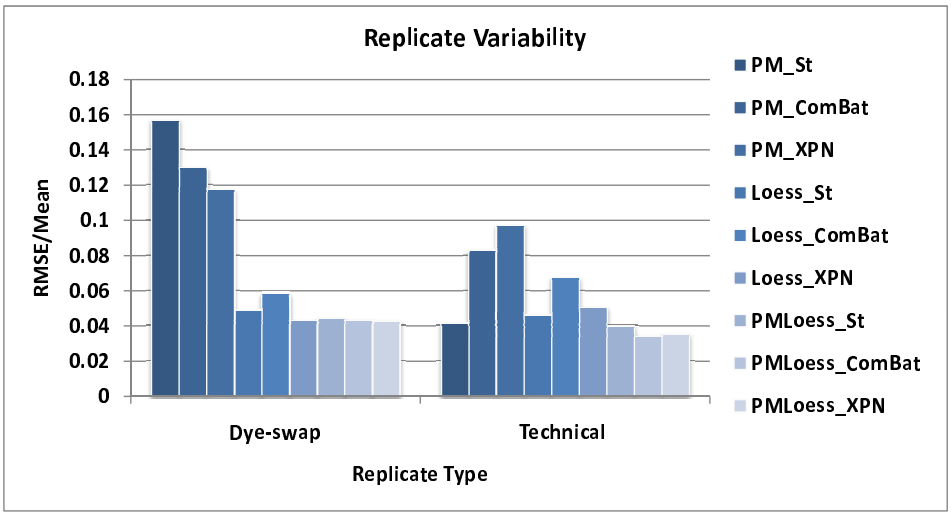


Figure 4.5: Variability between replicates in 9 datasets obtained by different normalisation techniques. The graphs show average RMSE/Mean (Equation 4.9) values for dye-swap (dual-channel arrays, PramilaL dataset) and technical replicates (single-channel arrays, Hasse dataset).

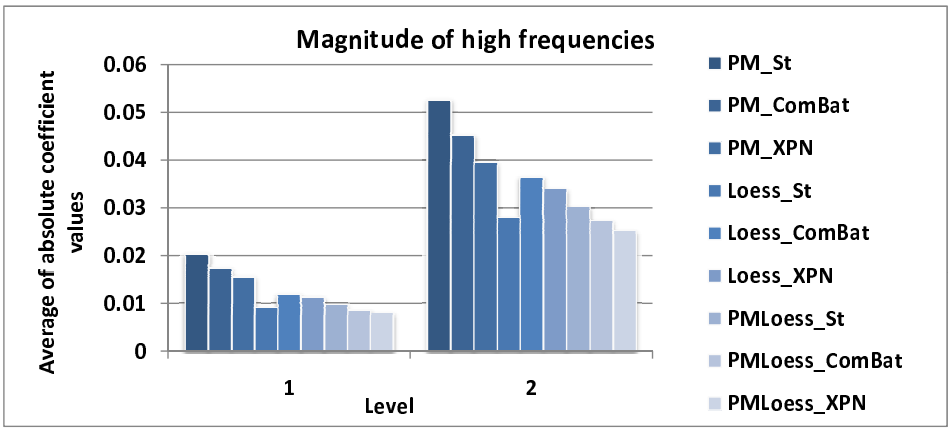


Figure 4.6: Magnitude of high frequencies. Graph shows average absolute value of wavelet coefficients for levels 1 and 2, corresponding to highest frequencies in the data, i.e. noise. Averages are computed over all four datasets.

4.3.3.1 Replicate variability analysis

Figure 4.5 indicates that *PMOnly* methods display increased variability in both dye-swap (dual-channel) and technical replicates (single-channel). *LoessOnly* methods exhibit low fluctuation even in single-channel technical replicates, indicating that, although not developed for this type of data originally, they perform well with respect to the variability criterion. Also, *ComBat* and XPN give increased variability between replicates compared to standardisation in some cases, showing that cross-platform normalisation comes with a cost.

4.3.3.2 Wavelet analysis of normalised datasets

Based on wavelet decomposition, the amplitude of high frequencies in the different normalised datasets was measured. High amplitudes indicate stronger noise compared to low amplitudes. Figure 4.6 shows average absolute values for wavelet coefficients for the highest frequencies in the data, over all four datasets. Results show that *PMOnly* methods display the largest fluctuations, while *PMLoess* methods give the smallest. This was expected to some extent, as the latter methods apply normalisation techniques especially tailored for

[5]Normalisation of RMSE values was necessary to enable comparison of model performance between the different datasets (Appendix C).

each type of data. Again, *LoessOnly* methods display good behaviour, very close to *PM-Loess*. However, in *LoessOnly*, *ComBat* and *XPN* seem to increase variability, in contrast to *PMOnly* and *PMLoess*, where variability decreases. This is in agreement with the replicate variability analysis (Section 4.3.3.1), and shows, again, that cross platform normalisation has a variability cost.

Secondly, wavelet coefficients corresponding to different scales are compared, for signals describing expression levels for the same gene occurring in different datasets. Nine genes known to be involved in the cell cycle, (analysed as a GRN also in Section 4.3.3.4), are compared across the four datasets and results are summarised in Figure 4.7. This shows that for levels 1, 2 and 3, (corresponding to higher frequencies), *PMOnly* methods generally show the largest differences between gene signals, while *LoessOnly* and *PMLoess* are comparable. This is probably due to high variability in high frequency *PMOnly* data, noted earlier. However, the behaviour seen for levels 4 and 5, indicates how different the core gene expression levels are. As Figure 4.7 shows, cross-platform normalisation methods bring the data significantly closer together, compared to simple standardisation.

4.3.3.3 Correlation analysis of normalised datasets

Firstly, the number of highly correlated gene pairs has been studied in each dataset. The correlation threshold used was 0.9 and Figure 4.8 shows the number of gene pairs with absolute correlation larger than this, for each normalised dataset. Results show a very large difference on the log scale between normalisation techniques used. *PMOnly* methods display a large number of highly correlated gene pairs in the Hasse dataset and in two of the three dual-channel datasets (PramilaS and PramilaL), while *LoessOnly* methods eliminate a large part of these correlations, especially in the Hasse dataset. The question here is whether this high number of correlations is an artefact of the *PMOnly* normalisation method, or whether *Loess* methods do, in fact, substantially decrease correlations. A second important observation is that *ComBat* and *Standardisation* display the same correlation values, while, in comparison, *XPN* causes significant decrease in the number of high correlations for all datasets.

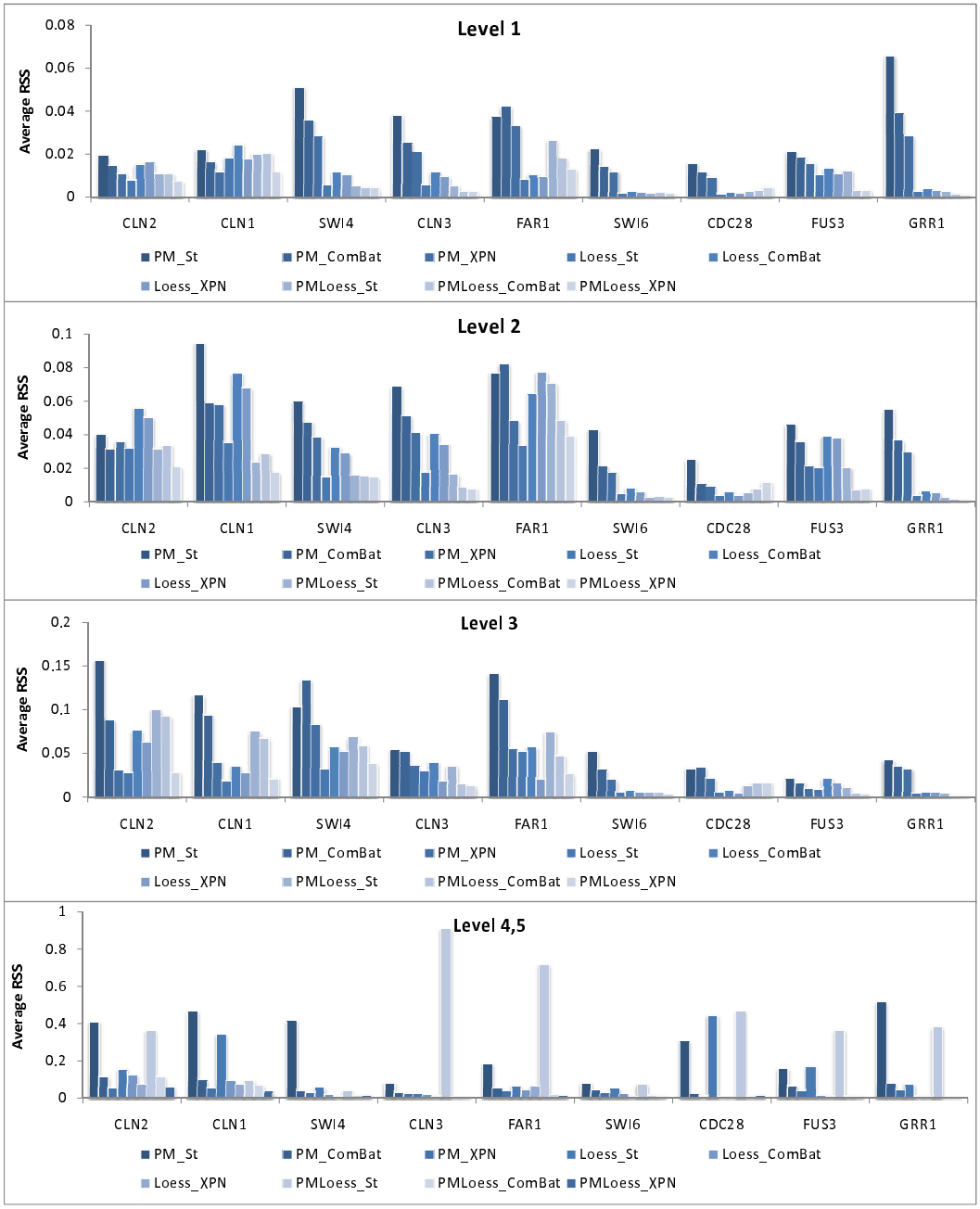


Figure 4.7: Dissimilarity between gene signals in different datasets. Graphs show average RSS between wavelet coefficients corresponding to nine genes in the four datasets, at different scales(levels). Level 1 corresponds to highest frequencies, i.e. noise, while level 4 and 5 to lowest frequencies, i.e. the real signal.

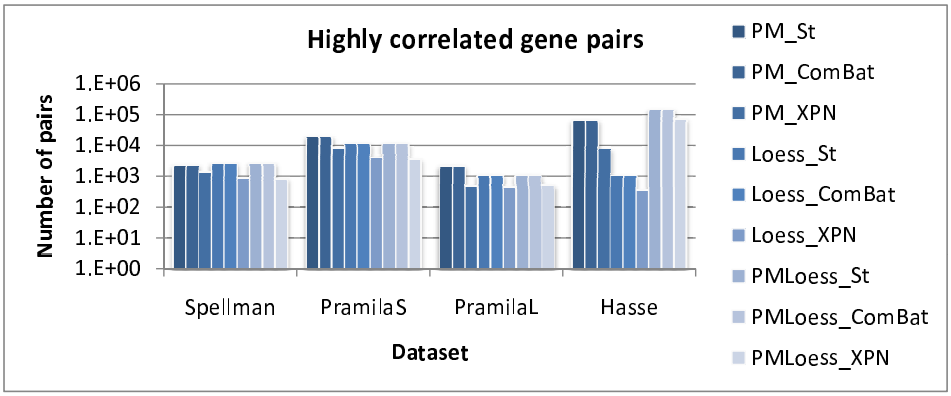


Figure 4.8: Number of highly correlated gene pairs in each dataset, for each normalisation technique (in logarithmic scale). The correlation threshold used was 0.9.

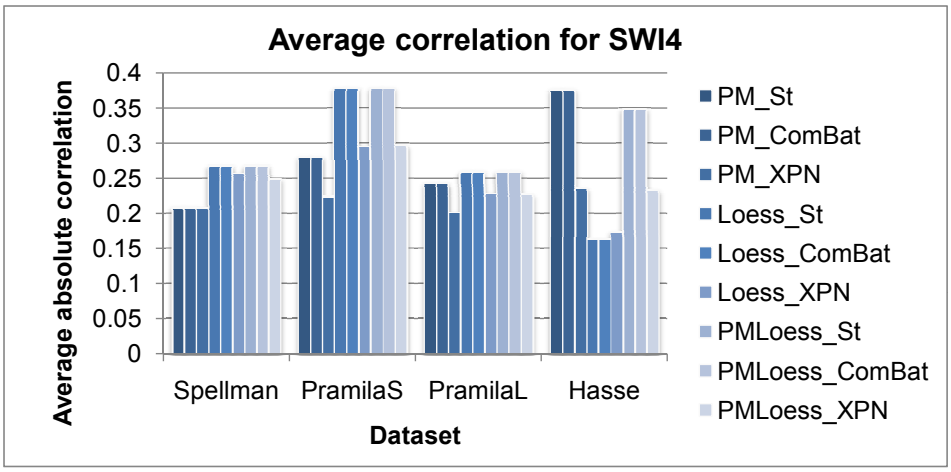


Figure 4.9: Average correlation for gene *SWI4*. This shows an aggregated measure of correlation of this gene with all other genes in the network, for each normalisation technique. Note that *ComBat* cross-platform normalisation does not affect correlations, while *XPN* decreases the average values.

Secondly, the average of absolute correlations for a subset of nine genes was computed, with results for gene *SWI4*, (which is a known transcription factor involved in cell cycle regulation), shown in Figure 4.9. For the dual-channel datasets, (Spellman, PramilaL, PramilaS), LoessOnly methods show large average correlation, in contrast to the low number of highly correlated pairs, noted for the same methods (previous paragraph). This suggests that, for dual-channel data, large correlations are only slightly decreased by *LoessOnly* methods, whereas, (given the large variability for *PMOnly*), the larger number of highly correlated genes may be an artefact of the *PMOnly* normalisation technique. For the single-channel dataset (Hasse), however, the average correlation is decreased by *Loess* normalisation. Considering the significant drop in highly correlated gene pairs (*Loess*), it can be concluded that, although *PMOnly* normalisation may lead to spurious correlations in the Hasse dataset, *Loess* normalisation might also decrease real correlations for these data, by over-smoothing, which is not uncommon of heavy processing. To test this, a further analysis for quantifying spurious correlation has been performed.

Correlation Variability Matrix In order to assess the amount of spurious correlation for each normalisation technique, the *Correlation Variability Matrix* (Equation 4.11) was computed for each normalisation procedure, and averages over all gene pairs (i.e. all elements in these matrices), are shown in Figure 4.10. *ComBat* does not affect correlations, compared to standardisation for cross-platform normalisation, so the corresponding datasets, (i.e. *PM_ComBat*, *Loess_ComBat* and *PMLoess_ComBat*), are not included in the analysis. Results show that *Loess* methods display smaller averages compared to *PM*, while *XPN* is lower still, indicating less spurious correlation. In conclusion, *Loess_XPN* exhibits the best behaviour, as coefficients are in good agreement across microarray datasets. This performance is closely followed by that of *PM_XPN*. This indicates that cross-platform normalisation has a larger effect than within-dataset normalisation on the correlation differences, which is to be expected. Given the use of the correlation variability matrix as a criterion for studying spurious correlation, it can be argued that agreement between datasets may just be due to systematic bias in the normalisation procedure. Although this can not be ruled out,

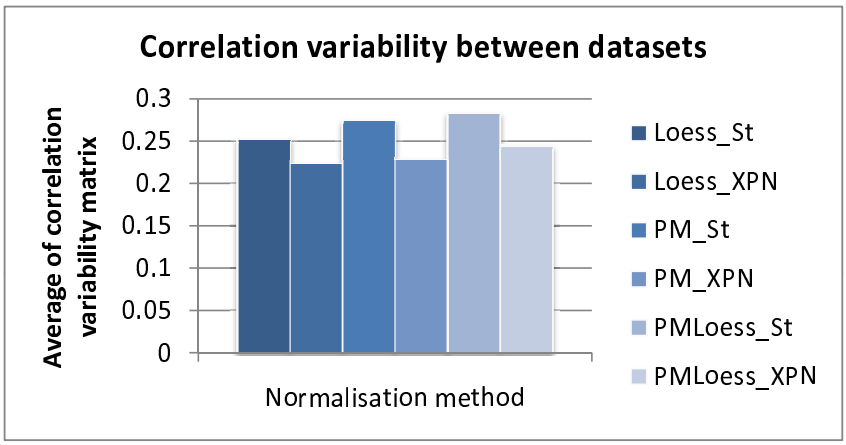


Figure 4.10: Average of correlation variability matrix. (Equation 4.11). Plotted here is the average of values in matrices for different normalisation procedures. Note that *LoessOnly* methods display lowest correlation variability, indicating less presence of spurious correlation, and better agreement between datasets. *XPN* normalisation also decreases differences, compared to standardisation. Thus *Loess_XPN* exhibits fewest differences, closely followed by *PM_XPN*.

compatibility between datasets is still required for data integration. Hence it may be concluded that methods that display large correlation variability perform less well. To further study the value of correlation, a small number of genes known to interact are analysed in depth in the rest of this section.

Analysis of genes known to interact In the context of GRN modelling, it is very important that known interactions between genes are preserved within correlation patterns. To examine this, we have chosen a set of 5 gene pairs, which are known to interact in reality ([Aoki-Kinoshita and Kanehisa, 2007]). These include pairs (a) *CLN1/2* of genes working together as a complex, (i.e. co-regulated), (b) *SWI4/CLN1* and (c) *SWI4/CLN2*, where *SWI4*, in a protein complex, is known to activate genes *CLN1/2*, and the pairs (d) *FAR1/CLN1* and (e) *FAR1/CLN2*, where *FAR1* represses the formation of *CLN1/2*. Ideally, for (a), (b) and (c), a high positive correlation should be seen in the data, while for (d) and (e), a high negative correlation should be present. Figure 4.11 shows correlations for each dataset, and each normalisation technique.

For the first three datasets, (Spellman, PramilaL, Pramilas - dual-channel), *Loess* nor-

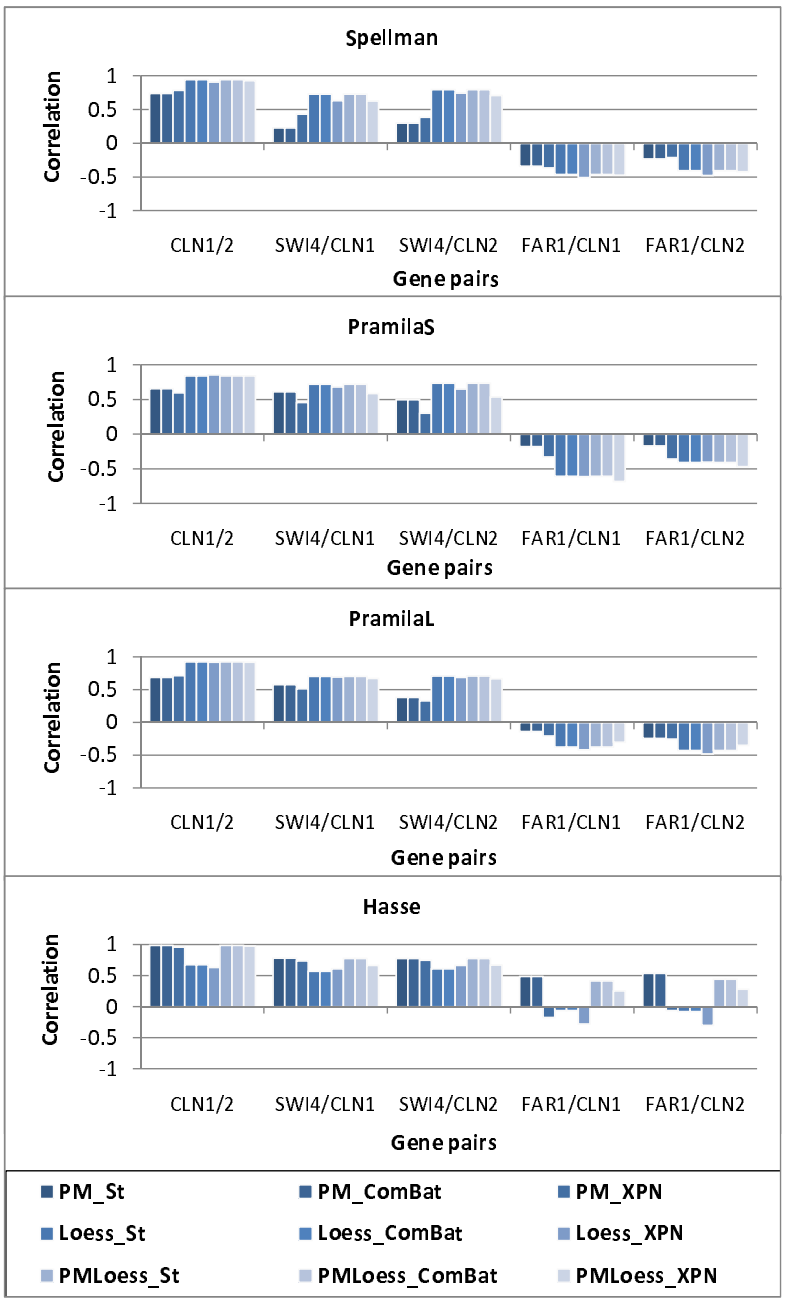


Figure 4.11: Correlation between genes known to interact. The first three pairs of gene are positively interacting, and the positive correlation values correctly indicate the interaction type, in all datasets. The fourth and fifth gene pairs, on the other hand, should display negative correlations, as they are repressor/target pairs. However, while for the dual-channel datasets this relationship is confirmed by negative correlations, in the Hasse dataset it is only visible with *PM_XPN* and *LoessOnly* methods, with *Loess_XPN* displaying largest absolute value. This indicates that *Loess_XPN* enhances correlations in this case.

malisation displays better behaviour, with *PMOnly* methods giving significant decrease in correlations between genes known to interact. It is important to note that the correlation values do correctly indicate the nature of these interactions, with positive values for (a), (b), (c), and negative for (d) and (e). However, correlations between *CLN1/2* are higher than those corresponding to activation/repression pairs, which can be explained by the regulatory time delay[6], which causes a shift in the expression signal of the target, compared to the regulator. For the Hasse dataset, on the other hand, the negative correlation between *FAR1/CLN1/2* is not present, except after *Loess* normalisation, and, even then, absolute values are very small. This supports the hypothesis that *PMOnly* methods introduce spurious correlations into the data, probably due to the higher noise present (discussed in Sections 4.3.3.1 and 4.3.3.2). For the other gene pairs, positive correlations are decreased using *Loess*, (Hasse dataset), but agreement with values obtained for dual-channel datasets (i.e. PramilaL, PramilaS and Spellman) remains good.

It is very important to note, when analysing gene pairs known to interact, that, although average correlations over all gene pairs are smaller, as noted earlier, *XPN* does not decrease correlations in all cases; some increases are observed, compared to other methods. This, combined with the low correlation variability between the different datasets, indicates that *XPN* may act as a better filter for spurious high correlations. At the same time it conserves or even amplifies 'useful' correlations, even where other techniques fail to do so (e.g. Hasse dataset, Figure 4.11).

4.3.3.4 Model translation between datasets

Applying models to test data can indicate whether pre-processing improves agreement between datasets. To assess this, we have computed the average RMSE/Mean between simulations of 20 S-System models for each dataset and the real expression values. Results are displayed in Figure 4.12, for gene *CLN2* models inferred from Spellman, PramilaL and Hasse datasets. These show that, in general, cross-platform normalisation, (as opposed to simple standardisation), significantly decreases error on all test datasets, making it a

[6]The time elapsed between the expression of the regulator and that of the regulated gene

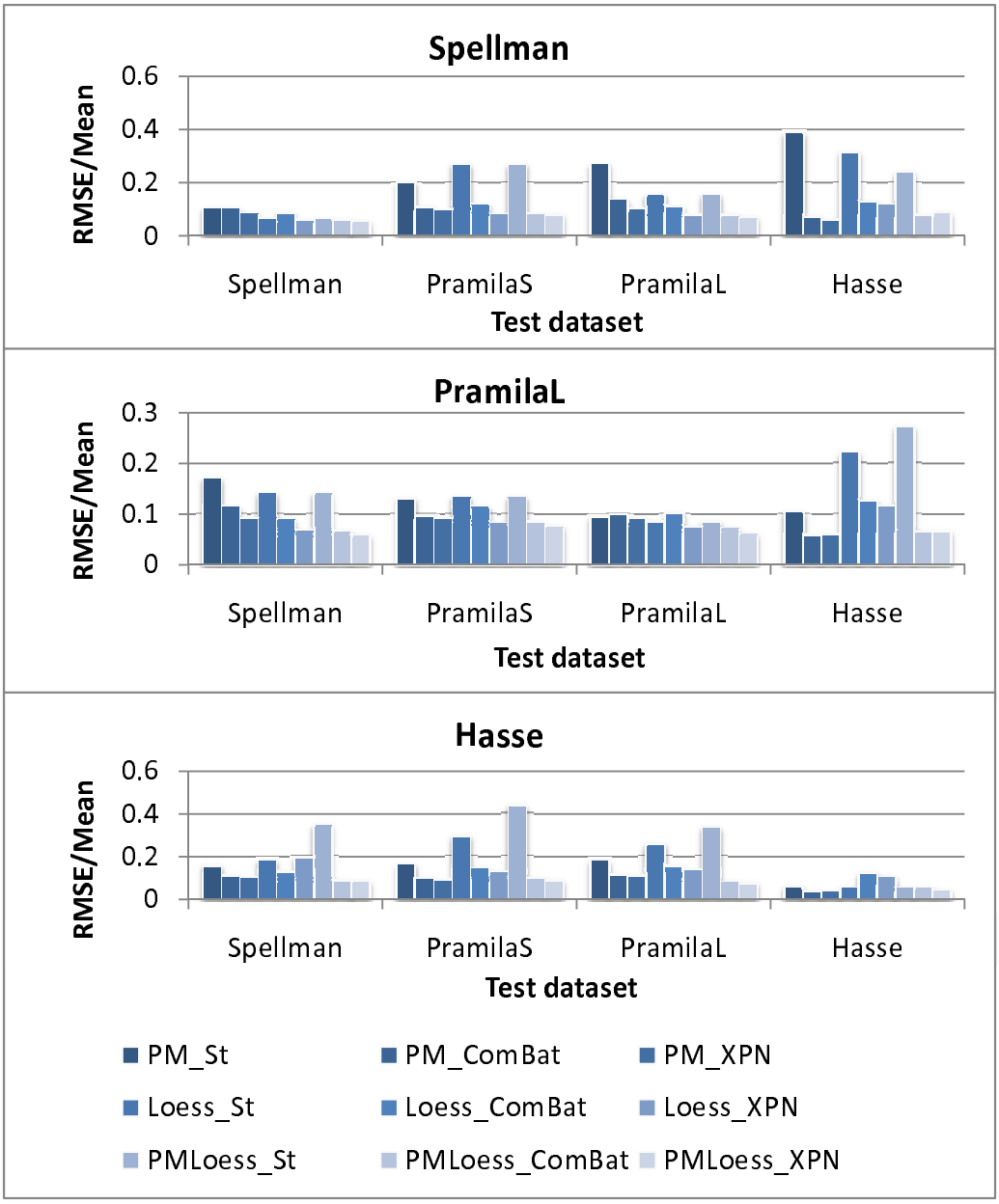


Figure 4.12: Average RMSE/Mean on all datasets for 20 S-System models for gene *CLN2*. Models were inferred from datasets Spellman, PramilaL and Hasse, separately, (identified by graph titles), and then tested on the rest of the datasets (horizontal axis). Graphs show that cross-platform normalisation, other than standardisation, decreases fitting errors for the test datasets.

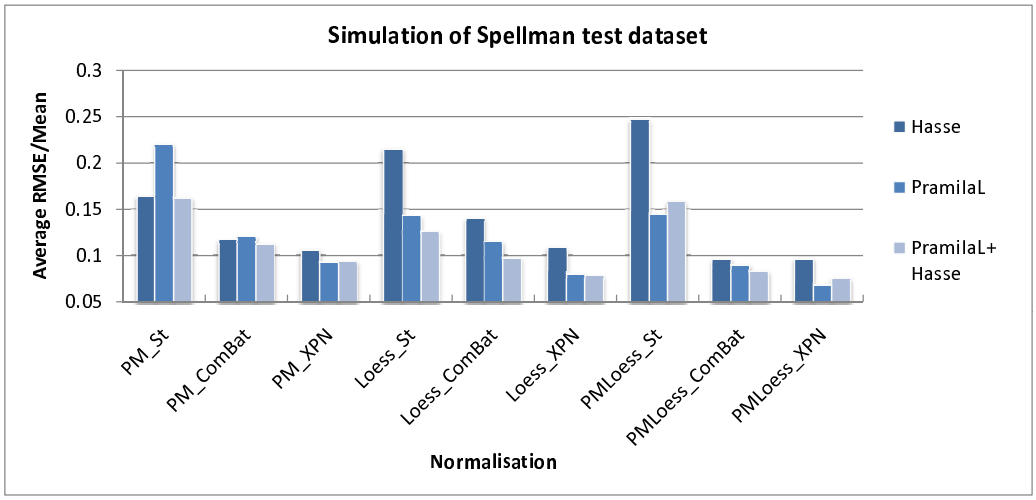


Figure 4.13: Average RMSE/Mean on the Spellman dataset for gene *CLN2*. Twenty inference runs have been performed with datasets Hasse, PramilaL and Hasse+PramilaL combined, and average errors, tested on the Spellman dataset, displayed for each normalisation technique. These show that, for *PMOnly* and *LoessOnly* methods, behaviour on the test dataset improves when using combined data, regardless of the cross-platform normalisation technique used, while for *PMLoess* methods this happens only for *ComBat* cross-platform normalisation. This is a good indication that these within dataset normalisation methods improve integrated data inference. *Loess_XPN* displays lowest RMSE values, suggesting that this is a suitable normalisation method for cross-platform data integration.

very important step in data integration for GRN modelling. Also, it is important to note that *PMOnly* and *LoessOnly* methods display behaviour comparable to *combined PMLoess* methods, indicating that these normalisation approaches are also suitable for time series model inference. Similar results were obtained for gene *CLN1*, but are not shown here.

Combining datasets In order to test how data integration improves model inference with different normalisation techniques, a second analysis was performed. This involved inferring models for the same gene (*CLN2*) from datasets PramilaL and Hasse together and testing these on the Spellman dataset. The resulting error, (averaged over 20 runs), has been compared to that obtained by models inferred from PramilaL and Hasse individually, with results displayed in Figure 4.13, as RMSE/Mean values. This shows that, for most normalisation techniques, increasing the number of datasets used to build models in the first place, helps to reduce error for subsequent application to a new test dataset. Exceptions are

PMLoess_St and *PMLoess_XPN*, where the error for the models inferred from combined data is larger than from the PramilaL dataset alone. This is not too surprising since, in these cases, within dataset normalisation is different for dual-channel (PramilaL and Spellman) and single-channel (Hasse) data, i.e. log-ratios are derived for the former and log-values for the latter. Consequently, model performance, tested on the Spellman dataset, is decreased by including the Hasse dataset in the training set. In *PMLoess_ComBat*, no increase in error when using two datasets exists, even though this method also uses different within dataset normalisation for single- and dual-channel data. This may indicate that the cross-platform normalisation employed (i.e. *ComBat*) is better able to reconcile the different measures. The decrease in error on the test dataset for *PMOnly* and *LoessOnly* methods, when using two training datasets as opposed to one only, indicates that the *integrative* within dataset normalisation procedures introduced here, (*PMOnly* and *LoessOnly*, which yield the same measures of expression levels for both single- and dual-channel data), do aid combined data inference.

Based on lowest error obtained for model inference, (using two datasets as opposed to one), *Loess_XPN* performs best, providing strong indication of its suitability as a normalisation method for data integration in GRN modelling.

4.4 Conclusions

We showed that integration of multiple time series for GRN quantitative model inference is possible and can result in improved models. We inferred GRN S-System models from four gene expression datasets measured on different platforms and analysed these and their simulated data. A robustness analysis showed that models obtained from multiple datasets were more resilient to both noise in the data and parameter perturbations. Additionally, a wavelet decomposition of signals corresponding to gene *CLN2* was performed. Results demonstrated that integrating heterogeneous time series minimised noise effects on models.

A further analysis of the influence of normalisation techniques on integrated GRN inference was presented. Three pre-processing approaches (*LoessOnly*, *PMOnly* and *LoessPM*)

have been applied to integrated raw microarray data from three different platforms. This has included application of techniques developed for dual-channel, (*Loess* [Smyth and Speed, 2003]), on single-channel data and vice-versa, (*PMOnly* [Li and Wong, 2001]). Following initial within-sample pre-processing, three cross-platform normalisation techniques, (*Standardisation*, *ComBat* [Johnson et al., 2007] and *XPN* [Shabalin et al., 2008]), were applied, resulting in nine normalised datasets. These have been compared for four criteria, relevant for data integration in the context of GRN quantitative modelling: variability between replicates, wavelet coefficient analysis, simple gene-gene correlations and GRN differential equation model translation between datasets.

In terms of the variability criteria, *LoessOnly* methods performed better than *PMOnly*, although combined *PMLoess* methods exhibited best performance overall. Wavelet analysis and model translation indicated that a second normalisation stage, (cross-platform), as opposed to simple standardisation, is required in order to align the datasets for the same inferential process. However, variance is increased for experimental replicates by cross-platform processing. Additionally, combining datasets was shown to improve model performance on a test dataset, especially when using integrated-within dataset normalisation, with best data fit obtained by *Loess_XPN*. Analysis of correlation between genes showed that *Loess* methods may over-smooth high correlation values, although patterns between genes that are known to interact are preserved. *XPN* also reduces some highly correlated gene values, but, in many cases, correlations between genes known to interact are amplified, even for those gene pairs for which other methods failed to obtain the correct correlation sign. This suggests that it is a fairly sensitive probe for determining true interaction patterns in the data.

In conclusion, results indicate that *Loess_XPN* was found to be the best method for normalisation of time-series data for quantitative model inference, as variability is acceptably low, datasets are well aligned and correlations between interacting genes are enhanced. Further, combined datasets produce models which perform better on test data than those inferred from one dataset only. The method permits integrated pre-processing across platforms, facilitating model inference from heterogeneous datasets.

Chapter 5

EGIA - a novel framework for GRN inference

In the previous chapter, a first step for data integration, consisting of combining microarray time-series, was discussed. Other types of data also exist in the literature, however, and are complementary to time series. In consequence, integration may further enhance model reverse engineering. The integration process, however, is not straightforward, as data types are not homogeneous, and computational complexity is increased, so careful analysis is necessary to (i) assess the viability of combination and (ii) identify suitable ways to achieve this. Here, a novel integrative framework based on evolutionary computation is presented (EGIA - Evolutionary optimisation of GRNs - an Integrative Approach), which seeks to exploit several related types of data. These include knockout experiments, Gene Ontology annotation of known transcription factors, binding site affinities and promoter sequence information (including known cis-regulatory modules), which are introduced in Chapter 2. Such data types contribute at different stages of the evolutionary algorithm. The framework is based on a previously introduced inferential algorithm, [Keedwell and Narayanan, 2005].

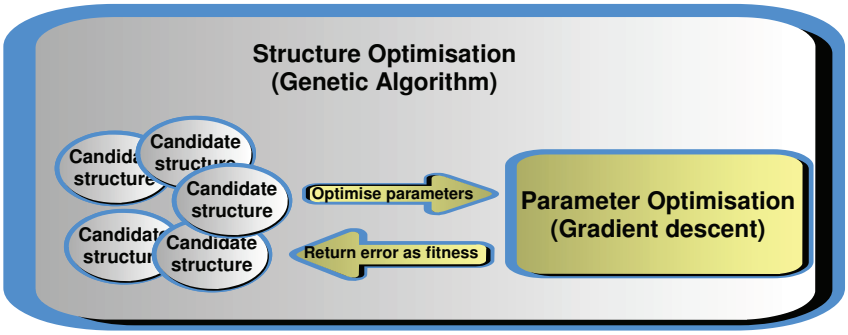


Figure 5.1: Structure and parameter search in EGIA.

5.1 The basic algorithm

In [Keedwell and Narayanan, 2005], a neural-genetic hybrid approach to GRN inference was introduced. This models the GRN as a single-layered ANN (Chapter 3, Section 3.2), consisting of one neural unit per gene. Each unit i takes as input the expression values of the regulators of gene g_i (i.e. g_j) at time point t and computes the expression level for gene g_i at time $t+1$, using the input weights w_{ij} and the logistic function for activation:

$$g_i(t+1) = S\left(\sum_j w_{ij} g_j(t) + b_i - d_i g_i(t)\right) \tag{5.1}$$

where b_i accounts for external input, while d_i represents the degradation rate.

$$S(x) = \frac{1}{1+e^{-x}} \tag{5.2}$$

One common issue with quantitative models is that most are black-box approaches, i.e. information on suggested interactions is difficult to obtain. However, biological interpretability is an important feature of GRN models, as it allows for both validation of results and extraction of meaningful information on possible interactions. Here, the single-layered approach facilitates interpretation of results, as input weights indicate the type of interaction between genes, while the capability to simulate non-linear behaviour is retained.

The basic algorithm divides optimisation into two phases: structure and parameter search (Figure 5.1). The first phase involves optimising network topology, i.e. the set of reg-

ulators for each gene. This is implemented as a Genetic Algorithm, where each individual encodes a candidate structure, as a subset of the possible regulators for the current gene. We have used an array of natural numbers identifying the set of transcriptional regulators, with size limited to a maximum connectivity level - maximum allowed number of regulators. Each candidate structure is assigned a fitness value during the parameter search phase. Parameter search employs Gradient Descent to optimise the input weights for the neural unit for the current gene, by minimising the squared error between data and simulation. The final error obtained is considered the fitness of the candidate structure under evaluation. The population of the genetic algorithm, (structure search), is initialised randomly and *Random One-Point Mutation* and *One-Point Crossover* are used to move through the solution space (see Chapter 3, Section 3.3 for details on these operators). *Tournament Selection* (Chapter 3, Section 3.3) is employed before crossover and mutation to generate offspring from fit parents only. A divide-and-conquer approach is used to optimise parameters for each gene at a time.

We have implemented this approach and compared it to different other methods in the literature (Appendix A) and results have shown that it is among the most scalable and least sensitive to (synthetic added) noise of the techniques implemented. Based on this, we have chosen to extend it further for data integration.

5.2 Algorithm enhancements and data integration

5.2.1 Algorithmic schema extension

The basic algorithm [Keedwell and Narayanan, 2005] optimises the parameters for each neural unit corresponding to each gene separately. While this approach is very useful, as it reduces the dimensionality of the system for each optimisation run, the model obtained by directly combining sub-models may not be able to correctly simulate the whole system, as separate optimisation disregards the feed-back from the genes being modelled. In consequence, we have added a second optimisation stage, which combines single-gene models and performs a fine-tuning of complete-model parameters (similar to [Kikuchi et al., 2003]).

Multiple single-gene runs have been performed, and the connectivity level (i.e. the maximum number of transcription factors for each gene) has been varied. This gave multiple possible models for each gene, which were combined in the complete-model optimisation phase, during the initialisation of possible solutions. The approach thus allows for selection of the best connectivity for each gene, facilitating simulation of the entire GRN.

In Chapter 4, we have seen that integrating cross-platform data reduces noise overfitting. This attempts to overcome experimental bias in the data. However, as discussed (Chapter 2), the gene expression process is itself stochastic, so models have to be robust to natural variability. One way of obtaining models robust to noise involves creating noisy replicates from the available data [Wessels et al., 2001]. This simulates technical replicates, and results in multiple time series to be used during inference. Here, a larger set of time-series has been derived from available data through addition of random Gaussian noise. This has been performed during the parameter optimisation phase, for ANN training.

5.2.2 Initialisation and mutation

The basic algorithm, achieves an initial population of candidate structures by randomly selecting possible transcription factors for a specific gene, from the set of genes in the network. Similarly, mutation is performed by replacing one of the regulators in the candidate structure with a randomly chosen gene. Starting with the initial population, mutation and crossover is used to explore the search space and move towards models with lower simulation error.

The additional data types that can be integrated in the inferential algorithm provide indications on which interactions between genes are most likely possible. For example, in knockout experiments, large log-ratios between wild-type and knockout expression levels may indicate an interaction between the knocked-out gene and the others. Similarly, binding site affinities can indicate what transcription factors can bind to a specific gene promoter. This information is very valuable, and can be used to explore the search space in a more knowledgeable manner. For this, we have developed a customised initialisation and mutation procedure, which uses likelihood assignment for gene regulation, based on additional

data. This results, for each gene g, in a non-uniform probability mass function, which describes which of the genes in the network are more likely to be regulators of gene g. When performing mutation or initialisation, this function is used to select a candidate regulator for gene g. This is similar to *Wheel of Fortune (WOF)* selection (Chapter 3.3), so will be addressed henceforth as WOF mutation and initialisation. The effect of this customisation of the algorithm components is assessed in Chapter 6.

In order to build the probability mass function for each gene g, the strategy is to assign segments on the WOF to each gene in the network, if there is any indication of a possible effect of that gene on the current gene g. All genes start with no slice allocated, and are allocated a number of segments when needed. This number of segments has to be set by the user, and in the following we are providing the values used in our experiments (empirically determined), but of course, these values can be changed to produce a higher or lower effect on the resulting WOF. Several different types of data can be used for this, as follows.

Correlation patterns Pair-wise correlation of gene expression levels has been widely used to determine putative interactions between genes and a good correspondence between correlation-based networks and GRNs has been identified [Xulvi-Brunet and Li, 2010]. In consequence, we have introduced a mechanism to use correlation values between gene patterns to enhance solution space exploration. For this, the Pearson correlation coefficient has been computed between the time-series data of all genes in the network, and fed into the Evolutionary Algorithm. Based on absolute values of the correlation to gene g, each gene i is assigned segments on the WOF:

$$CORR_{gi} = \begin{cases} 0 & \text{if } |r_{gi}| < \text{1st decile} \\ 1 & \text{if 1st decile} < |r_{gi}| < \text{3rd decile} \\ 4 & \text{if 3rd decile} < |r_{gi}| < \text{7rd decile} \\ 6 & \text{otherwise} \end{cases} \tag{5.3}$$

where r_{gi} is the Pearson correlation coefficient of time series data for genes i and g, while $CORR_{gi}$ is the amount of segments allocated to gene i on the WOF of gene g, based on

correlation data. The deciles are based on all correlation values obtained. In this way, genes that show high correlation with the current gene will be more likely to be selected as possible regulators.

Knockout experiments Gene expression data from knockout experiments can also be used to enhance the search for network models. Absolute values of log-ratios between wild-type and knockout samples can be fed into the EGIA framework, and these will be used to allocate segments on the WOF to those genes that display a large effect on other genes. The number of segments (KO_{gi}) allocated for each gene i on the WOF of gene g depends on the magnitude of the log-ratio:

$$KO_{gi} = \begin{cases} 0 & \text{if } |\text{log-ratio}_{gi}| < 0.2 \\ 1 & \text{if } 0.2 < |\text{log-ratio}_{gi}| < 0.5 \\ 4 & \text{if } 0.5 < |\text{log-ratio}_{gi}| < 0.8 \\ 6 & \text{if } 0.8 < |\text{log-ratio}_{gi}| < 1.1 \\ 8 & \text{otherwise} \end{cases} \tag{5.4}$$

Gene Ontology (GO) annotations The GO database contains annotations of which gene products have been observed to have a specific function, and annotations of transcriptional regulator activity can be included in the EGIA framework. For this, a list of the subset of genes in the network that have this annotation needs to be provided as input. These genes will be allocated additional segments (4 in our experiments) on all the wheels of fortune of the genes in the network:

$$ANNOT_{gi} = \begin{cases} 0 & \text{if gene } i \text{ is not annotated as TF} \\ 4 & \text{otherwise} \end{cases} \tag{5.5}$$

where $ANNOT_{gi}$ represents the number of slices allocated to gene i on the WOF corresponding to gene g based on annotation data. In this way, known transcription factors become more likely to be selected as regulators.

Binding site affinities Binding site (BS) affinities can be integrated in a similar manner to the other types of data. In order to compute the affinity between a regulator and a gene, the position weight matrix, (Chapter 2, Section 2.2.2), associated with the regulator is required, as well as promoter sequences for the gene. The promoter sequences can represent known cis-regulatory modules, or, if not available, simply the upstream DNA sequence. Using these two pieces of information, BS affinity values are computed as the maximum score obtained from the position weight matrix on the given sequences (Chapter 2, Equation 2.1). Based on affinity values, for each regulator i, the average ($\overline{A}$) and maximum affinity (A_{max}), over all target genes g, is computed, and segments on the WOF are allocated as follows:

$$BS_{gi} = \begin{cases} 0 & \text{if } A_{gi} < \overline{A} \\ 6 & \text{if } \overline{A} < A_{gi} < \overline{A} + \frac{\overline{A} - A_{max}}{2} \\ 8 & \text{otherwise} \end{cases} \tag{5.6}$$

where A_{gi} represents the affinity of gene i for binding to a promoter of gene g, and BS_{gi} represents the segments allocated to gene i on the WOF corresponding to gene i due to binding site affinity data.

Once all the segments, corresponding to the different type of data, are allocated for all possible regulators, these are summed (Equation 5.7) and the segments distribution is normalised to represent a probability mass function (Equation 5.8), by dividing by the total number of segments on the WOF.

$$WOF_{gi} = CORR_{gi} + KO_{gi} + BS_{gi} + ANNOT_{gi} \tag{5.7}$$

$$f_g(i) = \frac{WOF_{gi}}{\sum_i WOF_{gi}} \tag{5.8}$$

This probability mass function defines the probability that a gene i will be selected as regulator for gene g during mutation and initialisation. Each target gene g is associated with such a probability mass function, which will be used during initialisation and mutation to select new putative regulators and create new candidate network topologies. All data types

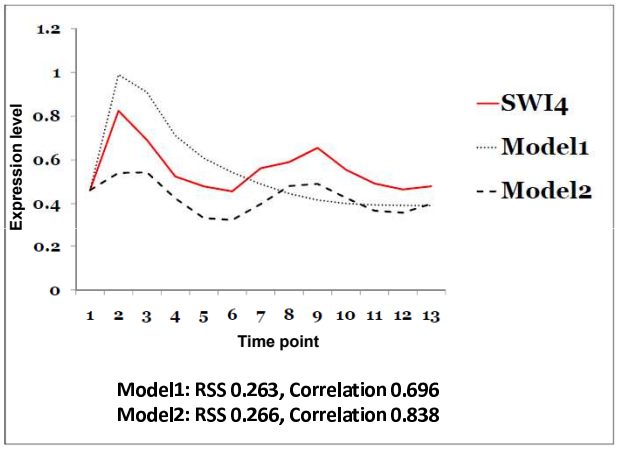


Figure 5.2: Simulations for gene SWI4 (Yeast cell cycle) from two models inferred using RSS for fitness evaluation. Model 1 displays lower RSS; however, it can not simulate the oscillation seen in the data. On the other hand, model 2 can simulate the behaviour, but RSS is larger. The correlation coefficient, however, indicates model 2 as better for simulation.

mentioned can be integrated or left out, depending on availability. When no additional data are available, the WOF mutation and initialisation are equivalent to the random assignment from the basic algorithm.

5.2.3 Evaluation

5.2.3.1 Introducing correlation for evaluation

In the basic algorithm (Section 5.1), the fitness of candidate structures is given by the training error of the corresponding ANN model, obtained after applying the Back-Propagation (BP) algorithm. The typical objective function for ANN BP is the RSS:

$$E = \frac{1}{2}\sum_{i}(o_i - t_i)^2 \tag{5.9}$$

where o_i is the output of the network, t_i is the true value in the training data, while the term $\frac{1}{2}$ is used to simplify computation. This is minimised through Gradient Descent to obtain values for weights, and has been previously used for GRN modelling [Vohradsky, 2001; Tian and Burrage, 2003; Keedwell and Narayanan, 2005]. However, it has some disadvantages in the context of evolutionary optimisation from time series, as it yields low

fitness values for those structures that can simulate general behaviour (i.e. the shape of the time series), but which have shifts in expression values. Hence, these structures can be discarded from optimisation, although, given the known high dependency between network topology and oscillatory behaviour in gene expression [Alvarez-Buylla et al., 2007], they may contain useful information. For instance, Figure 5.2 displays two different model simulations for a gene, where one has higher RSS, but can simulate the oscillation seen in the data (shown by the (Pearson) correlation value between simulated and real time series). During optimisation, this model would be discarded in favour of the one with lower RSS. To avoid this, a correlation term can be included in fitness evaluation. This has been performed in two ways in this work: (i) after BP, by adding the negated Pearson coefficient to the training error to obtain the fitness value for the individual and (ii) during BP, by introducing the correlation term in the training objective function:

$$OBJ = E - cr_{ANN,D} \tag{5.10}$$

where E is the RSS term from Equation 5.9, $r_{ANN,D}$ is the Pearson correlation coefficient, computed between the output of the ANN and the real data, and c is a constant weight for the correlation term, which is an algorithm parameter set by the user. The BP algorithm, employed here, minimises the objective function, i.e. minimises the RSS and maximises the correlation, by computing the gradient of each weight as described in Equation 5.11.

$$\Delta w_i = -\eta \left(\frac{\partial E}{\partial w_i} - c \frac{\partial P}{\partial w_i} \right) \tag{5.11}$$

where η is the BP learning rate. By adding the correlation term to the fitness function, those structures that are able to display the same oscillatory behaviour as the data are also assigned a good fitness, facilitating their selection for crossover and mutation. Furthermore, by including it in the BP calculation, parameter values are optimised to increase correlation between data and simulation. Models thus obtained display both qualitative and quantitative improvements, discussed in detail in [Sîrbu et al., 2010c]. This is one criterion for

enhancing fitness evaluation, with a second one introduced in the next section.

5.2.3.2 Extending evaluation to other types of data

The fitness function described above only considers time-series data for evaluation. Using additional data that might provide information on possible structures, and including it in model evaluation, is one possibility of addressing the noise and under-determination problem, inherent in time-series data. This further changes the fitness landscape so that models that have a plausible topology as well as ability to simulate the time-series data correspond to better fitness.

Section 5.2.2 has presented a mechanism of including different data types that contain indications on possible direct interactions into the mutation and initialisation operators of the EA. For each gene, this constructs a probability mass function that describes the likelihood of direct regulation from other genes in the system (WOF). Based on this, interactions present in a model can also be evaluated, by computing an average of all probabilities assigned to them by the WOF. This, used in combination with the previous fitness function discussed (Equation 5.11), enables construction of a fitness landscape that helps the optimisation algorithm find more plausible structures, as well as models that can simulate continuous behaviour. The final fitness function to be minimised is:

$$F = \frac{1}{2}\sum_{i}(o_i - t_i)^2 - cP - w\frac{1}{n}\sum_{(i,j)\in INT} f_j(i) \tag{5.12}$$

where the first term on the right hand side represents the squared error from Equation 5.9, the second the correlation term from Equation 5.11 while the last term is an average, over all pair-wise interactions present in the model, of the probabilities obtained by the WOF mechanism. INT represents the set of interactions predicted by the model ((i, j) represents an inferred regulatory effect of i on j), while $f_j(i)$ represents the fraction of the WOF allocated to that interaction (Equation 5.8). This term is weighted by w, a parameter which needs to be provided by the user. This evaluation criterion is used at both stages of the optimisation process, i.e. single-gene and complete-model optimisation. Its performance

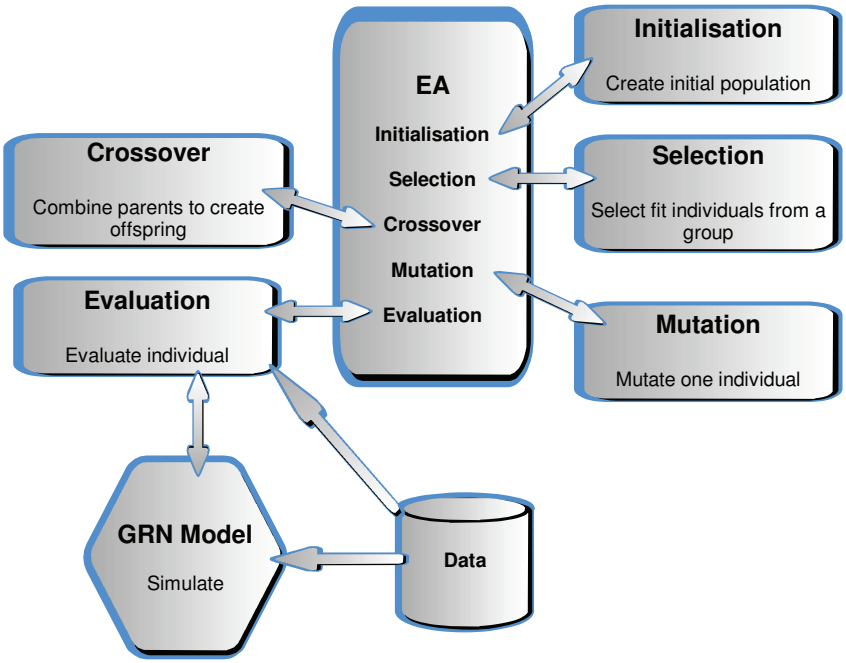


Figure 5.3: Framework implementation: main components.

on different datasets will be discussed in Chapter 6.

5.3 Implementation

The framework is implemented in C++, with an Object-Oriented modular design to facilitate replacement of different components for the analyses presented in the following chapters. For this, we have used virtualisation to build interfaces for each component of the EA. Additionally, the *Template Method*[1] design pattern has been employed at several stages. Figure 5.3 outlines the structure of the framework. Although a general criticism of using virtualisation in C++ is speed, the flexibility of this design type was necessary in order to facilitate analysis of different algorithm versions and models. The EGIA framework discussed above uses a subset of all classes implemented, derived through careful analysis of the different components involved.

The different modules involved correspond to general components of EAs (Chapter 3,

[1]Template Method [Gamma et al., 1995] is a design pattern that involves implementation of the main schema of an algorithm in the base class, and delegation of the implementation of specific steps to subclasses.

Section 3.3). Thus, the modules Initialisation, Mutation, Crossover, Selection and Evaluation are used by the main component, EA, during optimisation (Figure 5.3). These are defined as interfaces, and several classes for different versions of these operators are implemented (e.g. N-Point Crossover, Gaussian Mutation, Tournament Selection). Customised versions have also been developed, to account for additional biological knowledge, such as *WOF Mutation* and *WOF Initialisation*.

The Evaluation module makes the connection between each EA individual and the corresponding GRN model, which is implemented as an additional module. This facilitates replacement of model type, without large effects on the other components. The GRN model component is able to perform simulations of the gene expression process, given the necessary data and a set of parameters. Based on these, the Evaluation module computes the fitness function and returns it to the main component, EA. Two models are currently implemented in the framework, the S-System and the ANN, but the general framework for any differential equation systems exists. Two EA schemas are also implemented: differential evolution and genetic algorithm (Chapter 3). All components exist both for single-gene and complete system optimisation, and the EGIA algorithm uses both, in an iterative manner, as discussed (Section 5.2.1).

The complete implementation of the algorithm is attached to the thesis on CD support.

5.4 Parallelisation

The algorithm implements both coarse and fine-grained parallelisation, outlined in Figure 5.4, using MPI (Message Passing Interface).

An initial level of parallelisation, i.e. coarse-grained, is employed in the first stage, when single gene runs are divided between multiple processor subgroups. Hence, each of these subgroups is responsible for optimising single-gene model parameters for a subset of genes.

Within each subgroup of processors, a second level of parallelisation (fine-grained) is applied, used at the evaluation stage of the GA. Given the complex evaluation process,

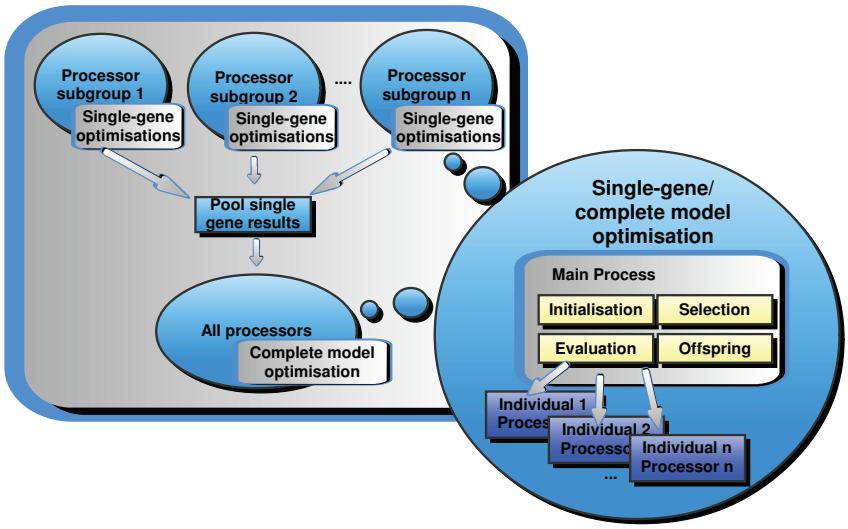


Figure 5.4: Parallelisation of EGIA.

which requires ANN training for each individual, the population is divided between the processors in each subgroup, to speed up computation. Specifically, the main processor performs initialisation, mutation and crossover, but when evaluation is required, it invokes the other processors, that evaluate individuals in parallel and send back the fitness values to the main processors. This approach is very useful as evaluation is the most time-consuming part of the algorithm.

For the second stage of the algorithm, i.e. complete-model optimisation, the single-gene results from the subgroups of processors are combined, and optimisation is performed using all processors (i.e. from all subgroups), by employing the same fine-grained parallelisation as for the previous stage. This involves evaluating each individual by a different processor, and returning the fitness values to the main node.

5.5 Discussion

This chapter presented the structure of the evolutionary integrative framework we have developed, with detailed explanation of each step and algorithm enhancement. The performance on both synthetic and real data will be discussed in subsequent chapters. However, several advantages of this integration approach can be outlined here.

Firstly, a major advantage is the flexibility in the amount of data required for inference. The framework is built so that as much value as possible is extracted from the data available. The only mandatory data type is time series gene expression data, which is required for ANN training. No restriction is imposed on the other types of data. For instance, if only one knockout experiment is available, this can be integrated, even if other genes lack similar data. Similarly, any number of binding site PSWMs or cis-regulatory modules can be used. This strategy enables a better description of interactions supported by previous knowledge, but it is not without risk, as some interactions might be over-represented, if extensive data focusing on them exists. This can be avoided by selecting the most representative subset of these[2].

Second, the parallel implementation of the algorithm allows for analysis of larger networks. In our experiments, we have studied networks ranging from 10 to 100 genes, with good results, and running times of under 10 hours, on cluster computers. Two facilities have been used, one internal, with Dual Quad-Core processors (2.66GHz) and one external, with Dual Hex-Core processors (2.67GHz). Upscaling for larger networks requires a larger number of nodes in order to obtain the model in such a short time. For instance, while for a 10-gene network 16 processing cores have been used, 100-gene network optimisation was performed on 180 cores.

Finally, a very important advantage of the framework is the data integration achieved. The extended number of types of data allows for a more informed exploration of the solution space, and reduces biases coming from one data type alone. Additionally, this approach of integrating additional data, beside time-series, is platform-independent, i.e. data from any experimental setting can be included. This is due to the usage of PSWMs (for binding affinity) and log-ratios (for knockout data), which can be extracted from every independent experiment (i.e. within-platform analysis) and then can be integrated in the EGIA framework directly, without any other pre-processing required. This is very important in the context of existing data, which often come from different laboratories and platforms. How-

[2]Selection can be based on reliability of data source of other pre-inference data analyses (e.g. selecting PSWMs that correctly identify some previously known interactions.

ever, large-scale integration introduces the risk of inferring distorted models if the different data sets are not of good quality. Consequently, a careful analysis prior to integration is required, to identify the data texture and the benefits introduced by each data type. A case study in this regard will be presented in Chapter 6.

Chapter 6

Step 2: Integrating other types of data

In this chapter, we consider a second step for the integration algorithm, i.e. inclusion of other data types. The methodology has been presented in Chapter 5; here we discuss the effects of integration on the algorithm performance. This includes an analysis of the different integration stages, i.e. customised mutation (*WOF mutation*) and evaluation. Further, the effects of including available data types at the different integration stages are discussed.

Algorithm performance is evaluated both quantitatively and qualitatively. Qualitatively, the AUROC (Area Under the ROC Curve) and AUPR (Area Under the Precision-Recall Curve) are computed, using a set of known interactions as gold-standard (see Appendix C for details on these measures). Given that the algorithm is stochastic in nature and the model quantitative, predictions of interactions have been performed by using multiple models obtained in different runs, and employing a voting procedure for possible interactions. In this way, an interaction that appears in more models is considered to be more plausible (this method of voting has been previously used to extract qualitative information in similar settings [Deng et al., 2005; Daisuke and Horton, 2006]). The set of possible interactions is ranked from highest to lowest number of votes, and used for AUROC/AUPR computation. Quantitative evaluation of the inferred models is performed by simulating a set of *test* data,

not used for inference.

Both synthetic (DREAM4 data, described in Appendix B.1) and real (*Drosophila melanogaster*, Appendix B.3) datasets are used to assess algorithm performance. The synthetic dataset includes two networks, of 10 and 100 genes respectively. For these, the gold-standard interactions for the DREAM4 dataset are used for qualitative evaluation, and MSE for dual-knockout experiments for quantitative. For real data, a sub-network of 27 genes involved in embryo development is analysed, (Appendix B, Table B.1). The single-channel (SC) microarray dataset, (as described in Appendix B.3), is used for training, while the dual-channel (DC) dataset is used for quantitative evaluation, (RMSE between real and simulated data). Cross-platform normalisation (namely XPN, Section 4.3.1) has been performed prior to model inference. For qualitative evaluation, interactions from the Drosophila Interactions Database (DROID, Appendix B) are considered gold-standard for the real dataset. The set of known interactions is not complete but is based on experimental evidence and gives a good indication on the efficiency of the algorithm in obtaining known direct interactions.

The hypothesis tested is that integration of large scale biological data improves both qualitative and quantitative performance of models inferred. However, using meta-information carries risk, as discussed (Chapter 5) and results are not always an improvement. Caution is required, as quantitative improvement should not negatively affect qualitative analysis and vice versa.

6.1 Customised mutation (WOF)

A first analysis of data integration studies the effects of using WOF mutation during optimisation, in comparison to random mutation. This mutation employs meta-information, available from different data types, to lead the algorithm towards models with more direct interactions that can simulate the time series. This attempts to reduce the under-determination problem for large GRNs, and, in consequence, improve the inferential performance of the automatic reverse engineering method. In order to identify which type of data is more useful, different variants of WOF mutation have been employed, to assess each type separately,

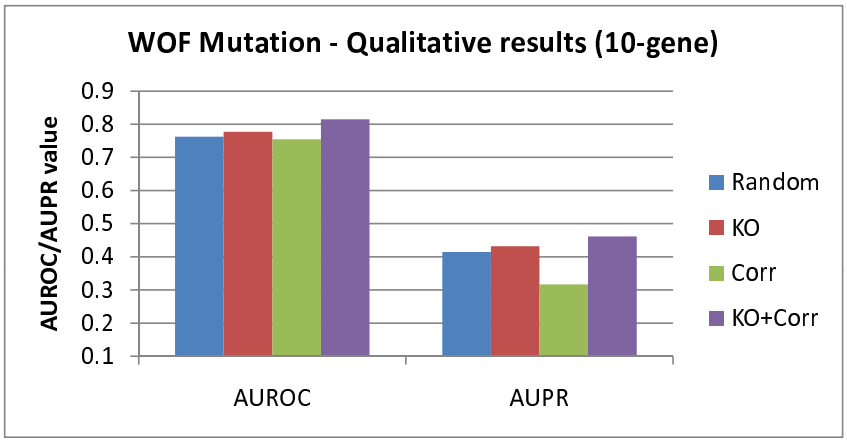


Figure 6.1: WOF mutation for the 10-gene DREAM4 network: qualitative results. The graph shows AUROC and AUPR values obtained after 10 runs with each WOF mutation variant, compared to random mutation (*Random*). The variants are: *KO* (knockout experiments), *Corr* (correlation patterns), *KO+Corr* (both). Indications are that usage of knockout data results in an improved interaction set predicted, as opposed to correlation patterns. However, usage of both provides the best predicted connections.

followed by the integration of all types.

6.1.1 WOF mutation for synthetic datasets

For synthetic data, two types of additional information have been used for WOF mutation, namely pair-wise absolute correlation between time-series for genes and log-ratios from knockout experiments. Three variants of WOF mutation thus apply: *KO* (using only knockout experiments), *Corr* (using only correlation patterns) and *KO+Corr* (using both). These have been compared for the two networks (sizes 10 and 100) with the random mutation.

For the 10-gene network, Figure 6.1 displays qualitative results, i.e. AUROC and AUPR values obtained after 10 different runs with each mutation operator. This indicates that a larger number of correct interactions (i.e. that exist in the gold-standard network) are included in models obtained by *KO* and *KO+Corr*, while correlation only does not display an improvement compared to *Random* mutation. At the same time, Figure 6.2 displays quantitative results, comparing average MSE values for dual-knockout experiments over 10 runs for each algorithm variant. T-tests were performed, to evaluate the significance of the MSE differences observed between the basic algorithm (*Random*) and all other variants, and

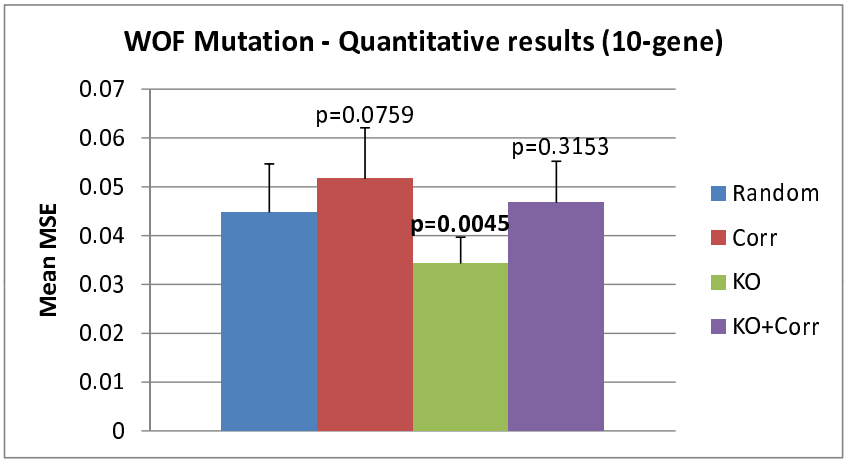


Figure 6.2: WOF mutation for the 10-gene DREAM4 network: quantitative results. The graph shows average MSE on dual knockouts for models obtained with different mutation variants (10 models for each bar) and error bars representing the standard deviation. Additionally, p-values of observed differences between each algorithm variant and the basic one are given. The same mutation variants as Figure 6.1 are present. Models inferred with KO mutation display best simulation ability, significantly different than Random mutation, at the 1% level.

p-values are displayed on the bar-plot. These show improvement in MSE at 1% significance level only for *KO* mutation, while the other variants yield MSE values similar or larger than random mutation. Together with AUROC and AUPR values, this indicates that, for the 10-gene network, correlation patterns are not particularly useful to extract models with increased performance, while *KO* mutation positively affects models, both qualitatively and quantitatively. Using *KO+Corr* yields the best set of predicted connections, but quantitative behaviour is not better than that of random mutation.

Similarly, Figures 6.3 and 6.4 display results for the larger network of 100 genes, for 9 inferential runs. Again, results show positive effect of knockout data on the predicted interactions, while using correlation only disimproves results compared to random mutation. On the quantitative level, *KO* mutation achieves best MSE improvement compared to the basic algorithm, significant at the 1% level. Similar to the 10-gene network, the best set of predicted interactions is obtained by *KO+Corr*, but with a slight increase in average MSE, so we can conclude that *KO* mutation displays best results.

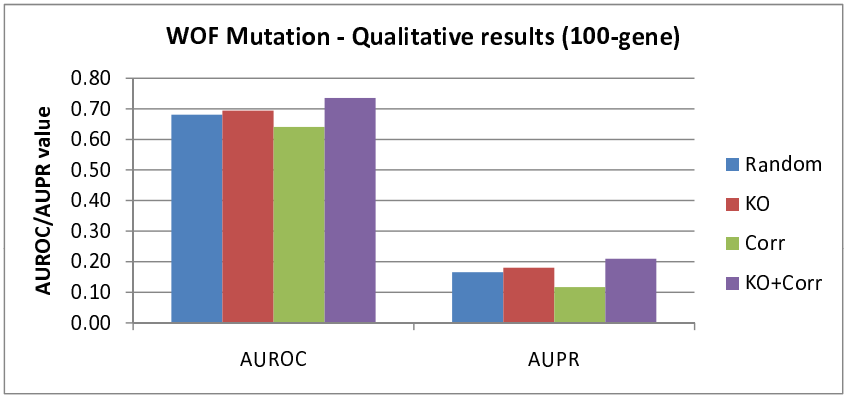


Figure 6.3: WOF mutation for the 100-gene DREAM4 network: qualitative results. The graph shows AUROC and AUPR values obtained after 5 runs with each WOF mutation variant, compared to random mutation. The variants are the same as Figure 6.1. Similarly to the 10-gene dataset, *KO* mutation has a positive effect, while *Corr* a negative one, and the best interactions are found using both types of data.

6.1.2 WOF mutation for the real dataset

For the inference of the *Drosophila melanogaster* 27-gene network, several types of publicly available data have been retrieved and used for building different variants of WOF mutation operators. These include (i) knockout experiments for 8 genes, which were used to compute log-ratios against wild-type experiments, (ii) pair-wise correlation between gene expression patterns, (iii) Gene Ontology (GO) [Ashburner et al., 2000] annotations, which assign the function of *transcriptional regulation* to 17 of these genes and (iv) binding site affinities for 11 transcription factors (computed using known cis-regulatory modules and position weight matrices). Details on these data are included in Appendix B.3, while the methodology to integrate them in the WOF mutation operator has been described in Chapter 5. With these data, five different variants of WOF mutation have been derived and compared to random mutation: (i) *KO*, (ii) *Corr*, (iii) *Annot*, (iv) *BS*, (v) *All* (employing all data available).

Figure 6.5 displays AUROC and AUPR values for the 5 variants, while Figure 6.6 displays RMSE (Root Mean Squared Error) values in simulation of dual-channel (DC) data used for testing only. The set of predicted connections improves when using some additional data types for WOF mutation. However, compared to basic mutation, changes in

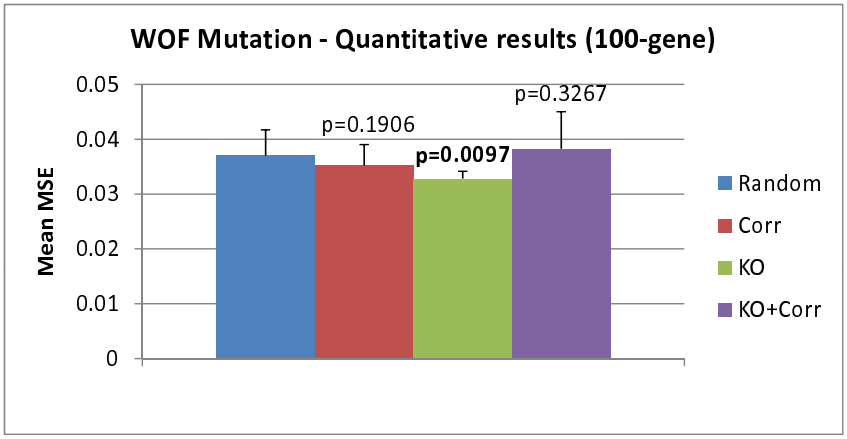


Figure 6.4: WOF mutation for the 100-gene DREAM4 network: quantitative results. The graph shows average MSE on dual knockouts and error bars for models obtained with different mutation variants (9 models for each), and corresponding p-values for comparison with the basic algorithm. The same mutation variants as Figure 6.1 are present. *KO+Corr* seems to decrease simulating abilities, with a larger mean for MSE values, while *KO* performs best, significantly better than the basic algorithm ($p = 0.0097$).

the simulation power seem random. For those data types showing better qualitative results, binding site affinities seem to be most important, followed by knockout experiments, while correlation patterns seem to decrease performance for these data as well. Similarly to synthetic data, the best interaction set is found by integrating all data types, indicating that the collective value of these data increases by integration, compared to the individual input of each data type.

However, the lack of improvement in simulation performance is a concern. It is expected that, if interactions are better described, the ability to simulate patterns in the data should increase. However, when applied to gene expression data, our customised mutation operator still cannot describe better quantitative models, even when additional correct interactions are found. This is particularly true for the real network. This may be because the fitness landscape is still distorted by noise or may be inherently linked to under-determination. The evaluation criterion (MSE and correlation to training data, Chapter 5) is crude so that reverse engineering becomes increasingly fuzzy. To address this, we have derived a novel fitness evaluation criterion that not only accounts for the similarity of simulation to training data, but also includes meta-information on possible interactions. This meta-information

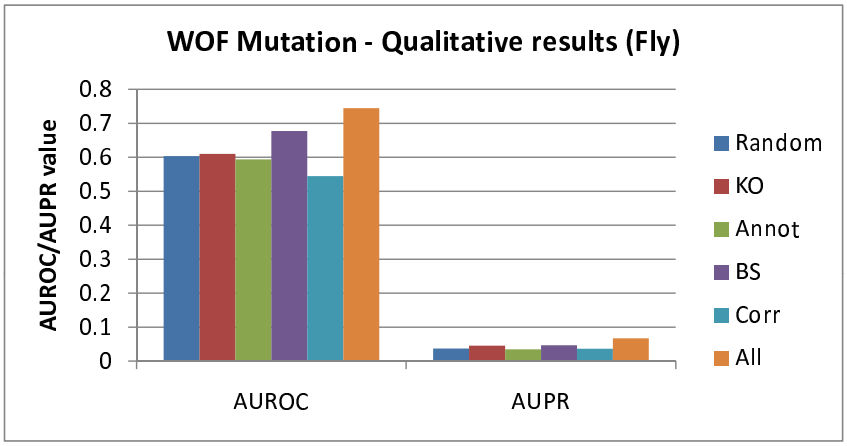


Figure 6.5: WOF mutation for the 27-gene *Drosophila melanogaster* network: qualitative results. The graph shows AUROC and AUPR values obtained after 10 runs with each WOF mutation variant, compared to random mutation. The variants are: *KO* (knockout experiments), *Annot* (GO annotations), *BS* (binding site affinities), *Corr* (correlation patterns), *All* (all data). *BS* displays the largest positive effect on the set of interactions retrieved, followed by *KO*, while *Corr* displays the largest negative effect, similarly to results on synthetic data. However, as for DREAM4 data, the concerted effect of all integrated data types provides the best inferred interaction set.

is based on the same types of data used for WOF mutation (details on derivation of the criterion have been provided in Chapter 5). This new evaluation criterion creates a different fitness landscape, where models that can simulate the training data, but also contain plausible interactions given this meta-information, are assigned a better fitness value. The next section discusses the effect on the resulting models of using this evaluation criterion combined with WOF mutation.

6.2 Extending evaluation

As with the previous section, different variants of the algorithm using different types of data are analysed, for each dataset. To analyse the error structure of each data type, algorithms variants are built by systematic exclusion of each data type from the pool of available data, i.e. step-down rather than step-up, as in the previous section.

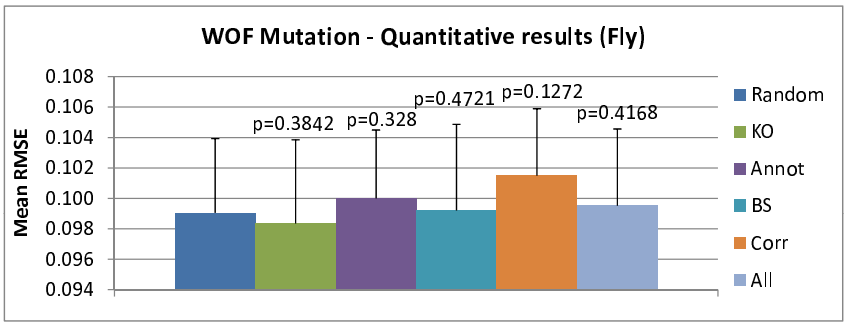


Figure 6.6: WOF mutation for the 27-gene *Drosophila melanogaster* network: quantitative results on test data. The graph shows average over 10 runs of RMSE on test data (DC dataset) for models obtained with different mutation variants. As previously, error bars and p-values of observed differences from the basic algorithm are displayed. The same mutation variants as Figure 6.5 are present. No significant change can be seen in the RMSE values.

6.2.1 Extended evaluation for synthetic datasets

For the synthetic datasets, only correlation patterns and log-ratios for knockout experiments are available, so, again, three versions of the algorithm were compared to the basic one [Sîrbu et al., 2010c] using random mutation and evaluation based on MSE and time series correlation to initial data. These three variants are addressed as *All-eval* (including all data available in WOF mutation and evaluation), -*KO* (all data excluding knockout experiments) and -*Corr* (all data excluding correlation patterns).

Figure 6.7 displays AUROC and AUPR values obtained after 10 runs of each algorithm on the 10-gene synthetic network, while Figure 6.8 displays average over 10 runs of MSE values for dual knockout simulations, and corresponding p-values of differences observed (compared to the basic algorithm- *Random*). As the figures show, extending the evaluation criterion appears to produce both qualitative and quantitative improvement when compared to the basic algorithm. Results are also better than when applying WOF mutation only (Figures 6.1 and 6.2), with larger AUROC/AUPR and lower MSE and p-values. The two types of data used, i.e. correlation patterns and knockout experiments, have different effects on the models. The set of predicted interactions is slightly improved when knockout experiments only are used (-*Corr*), but quantitative behaviour is best (lowest MSE values) when both data types are integrated. When knockout experiments are excluded, AUROC/AUPR

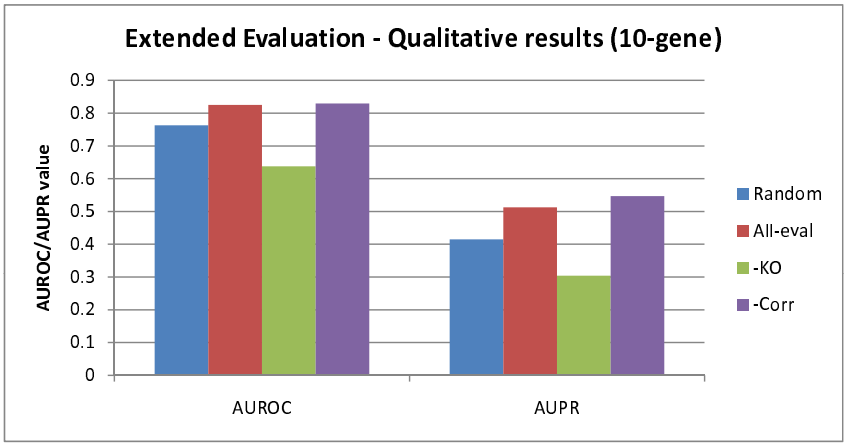


Figure 6.7: Mutation and extended evaluation for the 10-gene synthetic dataset - qualitative results. Three algorithm variants are compared to the basic algorithm without additional data, under two criteria: AUROC and AUPR. These variants are: *All-eval* (including all data available in WOF mutation and evaluation), *-KO* (all data excluding knockout experiments) and *-Corr* (all data excluding correlation patterns). AUROC and AUPR values are increased by using extended evaluation and WOF mutation. The greatest positive effect is from knockout data, while correlation patterns do not appear to be beneficial.

values decrease significantly. This shows that knockout data are very important for extracting direct interactions.

Similarly, for the 100-gene network, qualitative results are displayed in Figure 6.9, while Figure 6.10 evaluates quantitative behaviour on dual knockout data (after 9 runs for each algorithm variant). Introducing the enhanced evaluation criterion largely increases the number of correct interactions discovered, as shown by the AUROC and AUPR values. While WOF mutation only achieved an AUROC value of 0.73 and AUPR of 0.20 (Section 6.1), here we can observe values of 0.83 and 0.46 for these measures. The best results are obtained after excluding correlation patterns from the data types used, indicating again that these are not particularly useful in this context, (as found also for the 10-gene network). On the other hand, when excluding knockout experiments, AUROC/AUPR values decrease significantly, showing that these data are very important in predicting a good set of interactions.

From the quantitative point of view, the novel evaluation criterion yields further models with low MSE in dual knockout simulations (the minimum values obtained achieve values

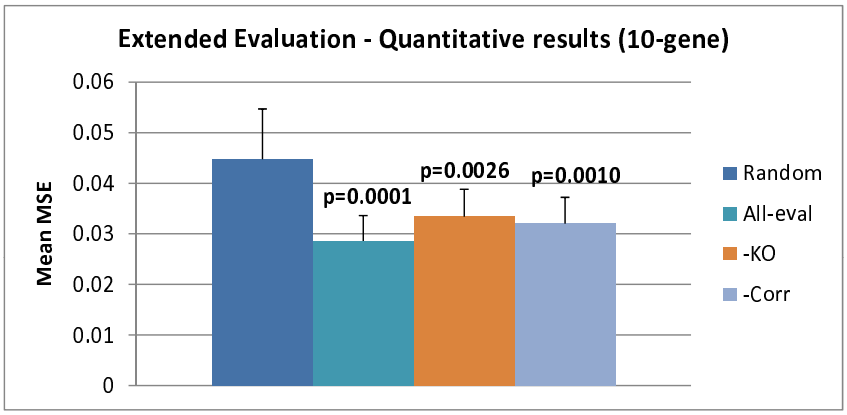


Figure 6.8: Mutation and extended evaluation for the 10-gene synthetic dataset - quantitative results. The same algorithm variants as Figure 6.7 are compared under the criterion MSE on dual-knockouts. Extended evaluation improves MSE on dual knockouts significantly for all algorithm variants, with the largest difference when using both knockout data and correlation patterns.

Table 6.1: Comparison of EGIA with DREAM4 results. For the dual knockout MSE values of EGIA, both the minimum and the average values obtained in repeated runs are provided.

	10-gene $\sqrt{AUROC * AUPR}$	10-gene dual-ko MSE	100-gene $\sqrt{AUROC * AUPR}$	100-gene dual-ko MSE
EGIA	0.6735	**0.019**/0.028	**0.624**	**0.0229**/0.0324
Team 548	0.654	0.038	0.544	0.0349
Team 532	**0.733**	0.020	0.505	0.0303
Team 498	0.702	0.029	0.28	0.0327

under 0.025, which has not been obtained by WOF mutation only), with best results when excluding correlation patterns (with average MSE lower than WOF). However, although minimum and average MSE are lower compared to the basic algorithm, the overall quantitative results from multiple experiments are only statistically significant at the 10% level (*-Corr*).

We have compared these results to those obtained by the participants in the DREAM4 competition. The top three teams that have submitted *quantitative* and *qualitative* results for *both* network sizes have been selected for comparison. For these, AUROC/AUPR and MSE values are given in Table 6.1, with best performances outlined in bold font. EGIA has obtained the *best predicted interactions for the large scale network*, while for the small

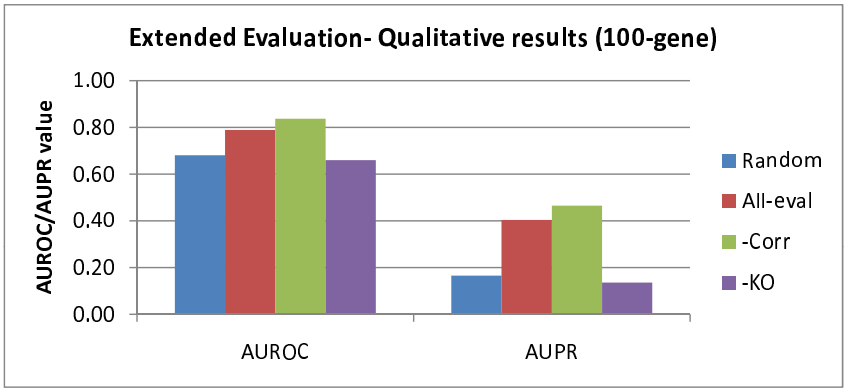


Figure 6.9: Mutation and extended evaluation for the 100-gene synthetic dataset - qualitative results. Three algorithm variants are compared to the basic algorithm without additional data, under two criteria: AUROC and AUPR. These variants are: *All-eval* (including all data available in WOF mutation and evaluation), *-KO* (all data excluding knockout experiments) and *-Corr* (all data excluding correlation patterns). Results show a large increase in AUROC and AUPR values after extending evaluation, with best influence from the knockout data.

scale one it scored 3rd. This shows that our method is more scalable compared to the others. From the quantitative simulation point of view, EGIA has obtained models with lower MSE than the other methods on dual knockouts for both network sizes; however, on average, behaviour is comparable to other methods. Nevertheless, given the good qualitative results, we conclude that this framework has something to contribute for extracting models with correct interactions, while it can also simulate unseen behaviour.

6.2.2 Extended evaluation for the real dataset

For the real dataset, five variants of the algorithm have been analysed: *All-eval* (evaluation and WOF mutation using all data available), *-Corr* (all data excluding correlation patterns), *-KO* (excluding knockout experiments), *-BS* (excluding binding site affinities), *-ANNOT* (excluding GO annotations), enabling assessment of the error structure in these data and how this influences the models obtained. Figure 6.11 displays AUROC and AUPR values for the five algorithm variants, compared to the basic algorithm. These indicate that integrating all types of data yields the best prediction for interactions. The largest effect is from the binding site affinity data, as with Section 6.1. However, all data types seem to

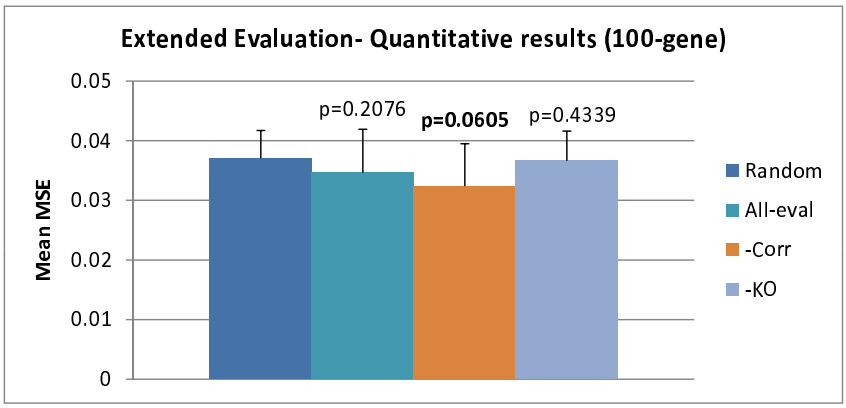


Figure 6.10: Mutation and extended evaluation for the 100-gene synthetic dataset - quantitative results. The same algorithm variants as Figure 6.9 are analysed. As previously, standard deviation and average of MSE values for dual knockout experiments and p-values are displayed. *-Corr* achieves best error.

contribute, unlike the synthetic data where correlation patterns used both for mutation and evaluation disimproved performance compared to the basic algorithm. The AUROC and AUPR values obtained here are no better than those resulting from WOF mutation only, indicating that some of the data types included in the evaluation criterion might favour indirect interactions also. This leads to a decreased qualitative performance compared to WOF mutation only, (but still superior to the basic algorithm).

Quantitative evaluation was performed again by computing the RMSE with the test dataset (DC), and Figure 6.12 shows average results obtained by each of the algorithm variants in 10 runs, with p-values of observed differences from the basic algorithm. Unlike WOF mutation, extending evaluation *does* improve quantitative behaviour, with RMSE values significantly lower than the basic algorithm (at the 1% level for *All-eval* and *-Corr*, and the 5% level for *-KO* and *-Annot*). This improvement is important, as it means that models not only contain more valid interactions, but also simulate test data better, i.e. improvement in both qualitative and quantitative performance. The error structure analysis also indicates that correlation patterns are once again not particularly useful for improving quantitative performance, while binding site affinities seem to be crucial.

For extended evaluation, qualitative results are not enhanced as much as by using WOF mutation only, while quantitative improvements are significant. Given this, we hypothesise

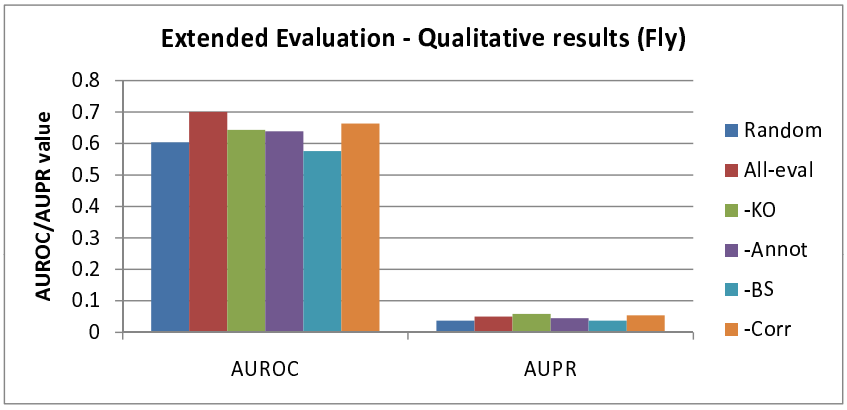


Figure 6.11: Mutation and extended evaluation for the 27-gene real dataset - qualitative results. Five algorithm variants are compared, under the AUROC and AUPR criteria, to the basic algorithm: *All-eval* (evaluation and WOF mutation using all data available), *-Corr* (all data excluding correlation patterns), *-KO* (excluding knockout experiments), *-BS* (excluding binding site affinities), *-ANNOT* (excluding GO annotations). All-eval achieves best AUROC/AUPR. The most important type of data appears to be the binding site affinity set, while the least affecting are the correlation patterns.

that it is possible to match the quality of connections from WOF, while maintaining the simulation abilities (low RMSE values on test data), by using only some additional data types for evaluation. While WOF mutation is a *weak* integration method, as it drives the algorithm only towards promising areas of the search space, without forcing it to choose one model or another, extended evaluation is a *strong* integration criterion, having the final say in which model is better or not. This means that, while the mutation operator can be resilient to some level of noise in the data, the evaluation criterion must include more specific data types. Given the results from the error structure analysis of the available data for the real dataset, correlation patterns, knockout experiments and GO annotation are more suitable for mutation only, as they provide *guidance* information only on potential interactions. Binding site affinities are, however, suitable for formal model evaluation, as they have proved to be crucial for obtaining good quantitative performance (Figure 6.12). For the rest of this section, therefore, we present similar analysis for different algorithm variants employing only binding site affinities in evaluation, but use various forms of WOF mutation: *BS-eval* (using all data types for mutation), *-Corr* (excluding correlation patterns from mutation), *-KO* (excluding knockout experiments), *-Annot* (excluding GO annotations).

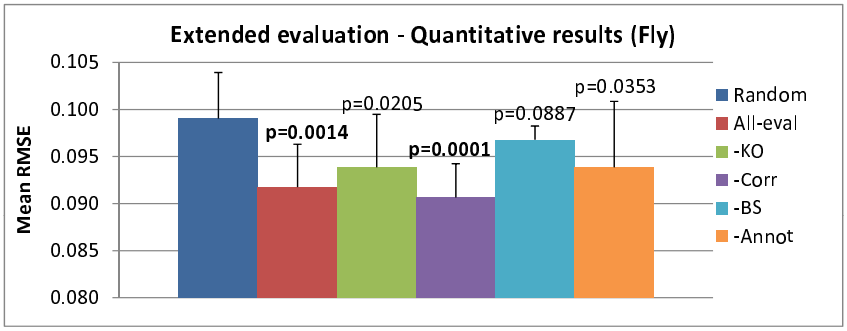


Figure 6.12: Mutation and extended evaluation for the 27-gene real dataset - quantitative results on test data. The same algorithm variants as Figure 6.11 are displayed. Average RMSE values on the test data, for 10 runs of each algorithm, with corresponding error bars and p-values are displayed. These show, finally for these data, a significant decrease in error. The graph also suggests the fact that binding site affinity data is crucial to obtaining better simulation abilities (a larger error is obtained by eliminating these data from the algorithm).

Figure 6.13 displays AUROC and AUPR values for all four algorithm variants above, compared to *All-eval* (evaluation and mutation using all data types) and *Random*, the basic algorithm (no meta-data used), while Figure 6.14 shows average RMSE values for test data. *BS-eval* produces models with better connections compared to *All-eval*, while RMSE on test data is maintained low (*BS-eval* and *-KO* significantly different from *Random* at the 1% level). Connections obtained with *BS-eval* are better than any of the WOF mutation variants (Section 6.1), while quantitative behaviour is similar to that for *All-eval*, suggesting that this is the best approach for integrating these available data for this dataset, providing both qualitative and quantitative improvements compared to the basic algorithm.

By extending evaluation, RMSE values for training data display a slight increase, both for synthetic and real data. One explanation for this is that, during training, due to the use of the advanced evaluation criterion, the *generalisation* ability of models is increased, as discussed (RMSE on test data decreases), and the *over-fitting* of training data is decreased. Generally, machine learning techniques need to obtain a balance between generalisation and over-fitting, which was made possible here by the inclusion of additional data types for training.

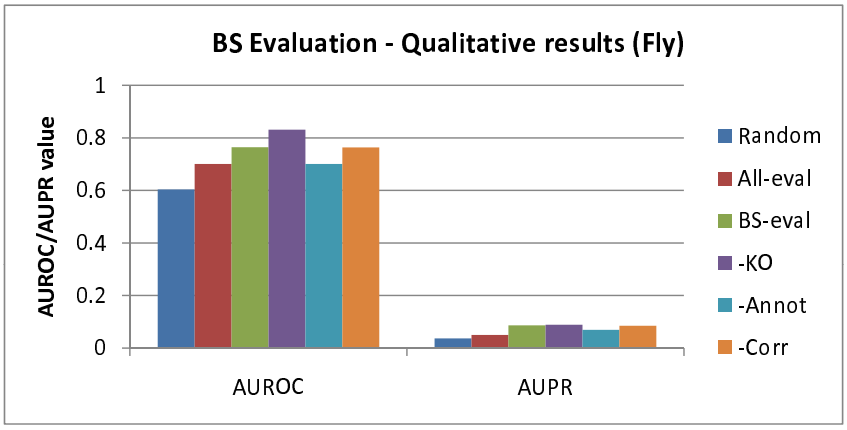


Figure 6.13: Mutation and binding site extended evaluation for the 27-gene real dataset - qualitative results. Four algorithm variants are compared, under the AUROC and AUPR criteria, to the basic algorithm and to *All-eval* (from Figure 6.11): *BS-eval* (using binding site affinities for evaluation and all data types for mutation), *-Corr* (excluding correlation patterns from mutation), *-KO* (excluding knockout experiments), *-Annot* (excluding GO annotations). All four variants are better than *All-eval*, while *-KO* achieves best overall AUROC/AUPR.

6.2.3 Integrated time series

In Sections 6.1 and 6.2, an analysis of integrating different types of data at different stages of the evolutionary algorithm (i.e. mutation and evaluation) was presented. Indications were that, for the real dataset used, using WOF mutation with all data types and evaluation extended for binding site affinities (*BS-eval*), gave best qualitative and quantitative predictions. A single channel time-course dataset (SC) was used for training, and a dual-channel (DC) for testing. This was necessary in order to enable validation of the different integration approaches, and to select an algorithm variant with optimum performance. However, in Chapter 4 it was shown that integrating different available datasets can lead to less noise over-fitting and better prediction of interactions. In this section, we develop this finding and use the best-performing algorithm variant (*BS-eval*) to integrate the two microarray datasets available for the *Drosophila melanogaster* embryo development. The aim is to obtain better prediction of interactions between genes.

The RMSE (root mean squared error) values for the DC and SC dataset are provided in Figure 6.15, both when the SC dataset only is used for training, and when the two time-

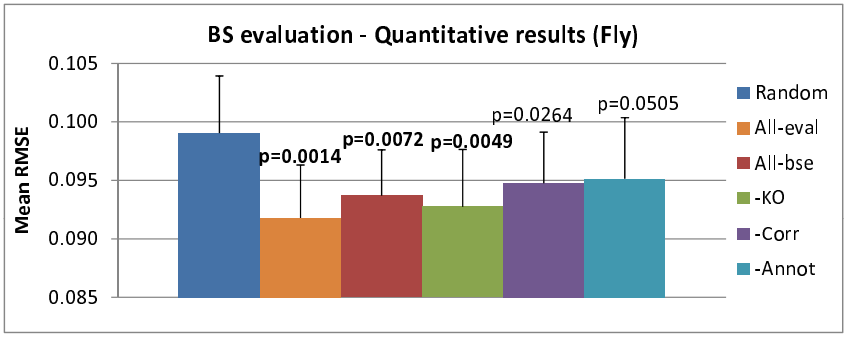


Figure 6.14: Mutation and binding site extended evaluation for the 27-gene real dataset - quantitative results on test data. The same algorithm variants as Figure 6.13 are displayed. Average RMSE values on the test data, for 10 runs of each algorithm, error bars and p-values are displayed. These suggest that the simulation abilities for test data remain significantly different from Random, when *BS-eval* is used.

course datasets are integrated. This shows that, after integration, the RMSE on the SC dataset increases slightly, indicating less noise over-fitting, while that for the DC dataset decreases, as expected when data is transferred from the test to the training set.

Additionally, Figure 6.16 displays AUROC and AUPR values obtained by time-series integration, compared to using only the SC dataset for training. This shows increase in AUROC and AUPR values, suggesting that *Drosophila melanogaster* gene-interactions predicted from the two datasets are better than from one only. This is in agreement with broad conclusions of Chapter 4.

6.3 Conclusions

This chapter has presented an analysis of data integration for gene regulatory network modelling. Two integration mechanisms have been analysed, namely *customised mutation* (WOF) and *extended evaluation*. Additionally, the error structure of available data has been studied in order to identify which data type has a larger effect on the networks analysed here.

WOF mutation yielded better prediction of pair-wise interaction between genes, compared to random mutation, for all networks analysed, with knockout experiments and binding site affinities proving to be most important. However, for the real network, no quantita-

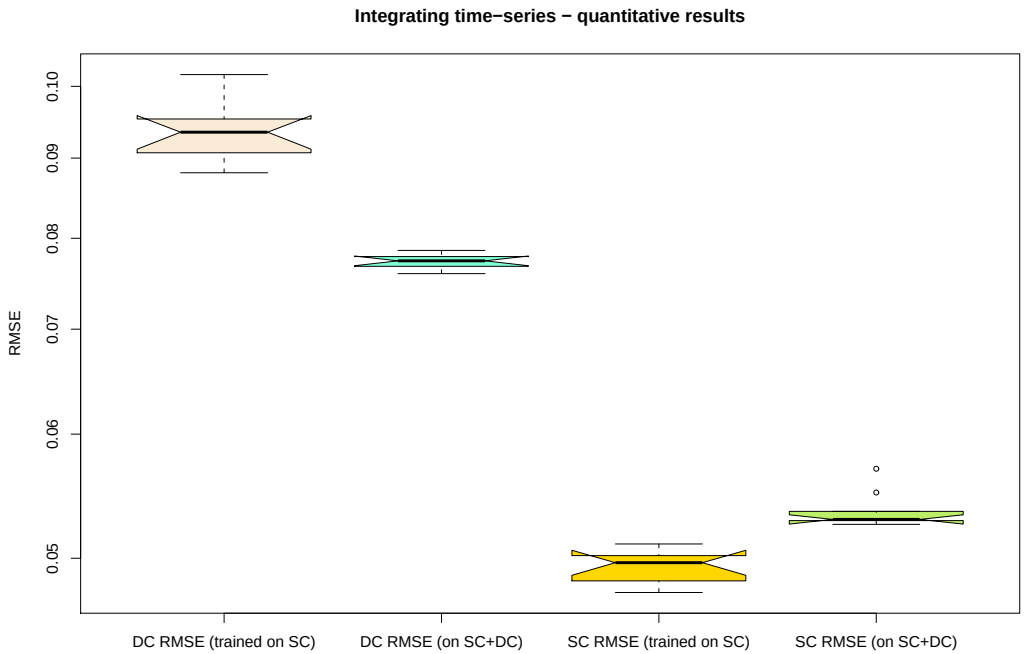


Figure 6.15: Integrating time-course data for mutation and binding site extended evaluation on the 27-gene real dataset - quantitative results. Boxplots of RMSE values on the SC and DC datasets after inference from either SC only or SC+DC (10 runs for each box plot). The image shows that the RMSE on the DC dataset decreases after integration (naturally, as it is transformed from a test dataset into a training dataset), while RMSE on SC data increases, suggesting less over-fitting by integration.

tive improvement could be achieved by considering other factors in mutation alone, which is disappointing as better knowledge of interactions involved should intuitively lead to improved model performance. An extended evaluation criterion, however, led to both quantitative and qualitative improvement, with smaller errors obtained on the test datasets. This supports the hypothesis that, given the limited nature of data, evaluation with time-series alone is not powerful enough for GRN models and that additional information from other data types is needed to aid further selection of models.

The error structure analysis suggested that not all data types are useful for inference, however, and that great caution needs to be taken when integrating these. For synthetic data, knockout experiments proved to be highly important to obtain better predictions of regulatory interactions, while for real data, binding site affinities seemed to have the largest

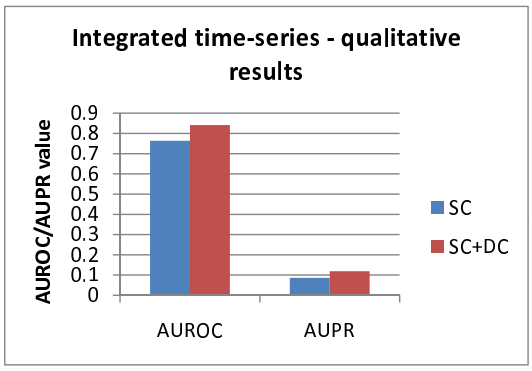


Figure 6.16: Integrating time-course data for mutation and binding site extended evaluation on the 27-gene real dataset - qualitative results. AUROC and AUPR values are displayed for connections obtained with the *BS-eval* version of the algorithm, from the SC dataset and from both SC and DC datasets. Results show that better connections are obtained integrating both datasets for inference.

impact. Correlation patterns, on the other hand, were of some help when integrated in WOF mutation with the other data types, while individually were of less importance. This might be due to the fact that correlation patterns do not indicate only direct interactions, but also indirect effects, which can be captured by the models. Additionally, it is difficult to distinguish between co-regulated genes and those that influence eachother, only from these patterns. This is why integration with the other data types proved to be more valuable.

WOF mutation proved to be a flexible integration tool, while evaluation is a more rigid one. For best quantitative and qualitative results, only very reliable data should be used for the latter, while noisy data can be integrated in the former. In our experiments, best performance on real data was found to be that for the algorithm variant which used only binding affinities for evaluation, and all data types for mutation. This suggests that the other data types can provide only general guidelines for possible structures. For instance, log ratios in knockout experiments, or correlations between gene expression patterns can sometimes be misleading, due to the existence of feedback loops, alternative regulatory paths or indirect interactions in the real network. Also, results are highly dependent on data quality.

Finally, an integration of the two time-series datasets has been performed, and this further improved the ability to predict possible interactions, supporting findings from Chapter

4.

Given the heterogeneity of the different data types and the intrinsic noise, this chapter has also introduced a general methodology for data integration. This consists of dividing the available time-course data into test and training datasets and performing an error structure analysis to identify which data type is important for both qualitative and quantitative performance, and at which stage these data should be integrated (i.e. mutation or evaluation). This exposes the texture of the data. Based on the results obtained, we have concluded that noisy data can be useful if integrated within the mutation operator, to span the solution space. At the same time, data which are more specific with respect to possible connections should be included in evaluation. The results presented here apply for the *Drosophila melanogaster* embryo development and associated datasets available for this system. When changing the system under analysis, e.g. a different process or organism, the data types available and their quality change, so performing an initial error analysis is crucial to determining the best integration strategy.

Chapter 7

Step 3: Extension to NGS data

In this Chapter (Sections 7.1, 7.2 and 7.3), an analysis of RNA-seq expression data in comparison with microarrays is performed. Previous platform comparisons exist in the literature (Chapter 3, Section 3.6), and these concentrate on validating results from RNA-seq on microarrays, and identifying advantages and disadvantages. For this, existing analyses use the same sample for measurements on different platforms. However, in the real setting, large scale data integration means combining heterogeneous datasets measured in different settings and with different samples. In consequence, a discussion of overlapping features in a broader sense, that may allow further integration of these data, is presented here[1]. For this, three gene expression time series datasets measuring embryo development for *Drosophila melanogaster*, (RNA-seq (NGS), single-channel (SC) and dual-channel (DC) microarrays, Appendix B, Section B.3), have been studied for differential expression (DEx) , and results compared to previous analyses focusing on more restrictive samples. Further, a cluster analysis is presented, to identify the structure of the gene space in the different datasets. This has not been previously performed, to our knowledge (sample classification only has been briefly analysed, [Cabanski et al., 2010]).

Additionally, Section 7.4 analyses the performance of the inferential framework presented here (EGIA) on RNA-seq data. Although application of the framework to this differ-

[1]Raw RNA-seq data has been processed by Dr. Gráinne Kerr, German Cancer Research Centre, Heidlberg, to obtain gene counts and expression values.

ent type of data is straightforward, an analysis of results obtained needs to be performed in order to evaluate the extent to which the framework can be extrapolated for these data. For this, the same 27-gene network for *Drosophila melanogaster* embryo development, studied in Chapter 6, has been analysed and results compared to those obtained by using microarray data for inference.

7.1 NGS vs. microarrays: comparison setup

7.1.1 Data pre-processing

7.1.1.1 Sequencing data

The Illumina reads were mapped to the April 2006 assembly of the *Drosophila melanogaster* genome (dm3, BDGP Release 5) using Tophat (v 1.0.14). This tool also makes use of gene annotations to detect reads that map across known and putative splice junctions. HTSeq, a Python package that provides infrastructure to process data from high throughput sequencing experiments, was used to 'count' the number of reads mapping to each gene. Read counts per gene were calculated to be the total number of reads, which mapped uniquely to annotated regions (Release 5.12 annotations). Reads that mapped to more than one location were considered ambiguous and not used. Unique reads, which mapped to a locus with more than one annotated gene, were considered ambiguous and not used (Figure 7.1). Reads per kilo base per million reads mapped (RPKM) values were calculated to estimate and compare mRNA expression levels across samples. Gene length was defined to be the region that encompasses the union of all isoforms of a gene, which do not overlap other genes, (Figure 7.1).

7.1.1.2 Microarray data

For the two microarray datasets, R software, specifically the *limma* package [Smyth and Speed, 2003], was used for normalisation and expression value extraction. Background extraction, within-array and between-array normalisation was performed for the DC dataset

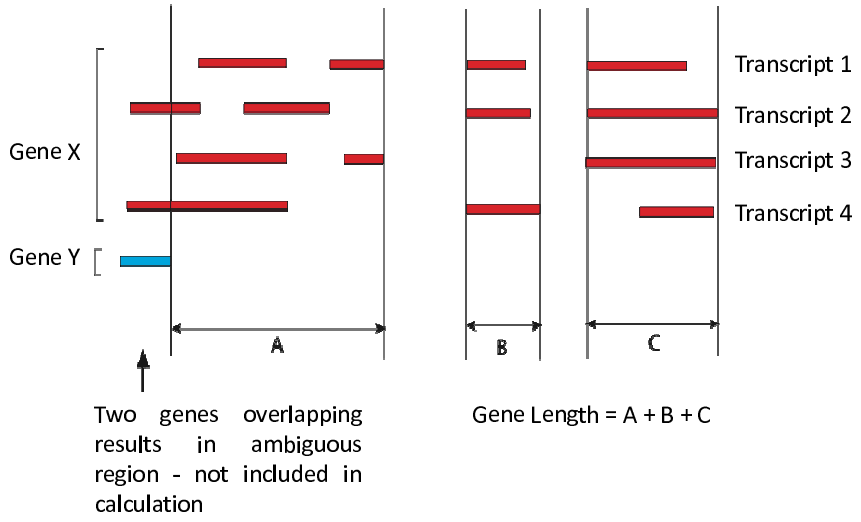


Figure 7.1: Calculation of gene length.

using the *normexp* and *loess* methods in *limma*, while for the SC dataset, the RMA method was employed. The resulting normalised datasets were used for DEx and cluster analysis.

7.1.2 Evaluation of differential expression

DEx analysis was performed, using R software, i.e. the *limma* package (*lmFit* and *eBayes* methods [Smyth, 2004]), for the two microarray datasets (SC and DC), and the *DESeq* package [Anders and Huber, 2010] for the sequencing dataset (NGS). For each dataset, the set of differentially expressed (DEx) genes, for at least one time point, compared to the initial one, was retrieved. Given that the data were sampled at different time points and sampling intervals in the three datasets, only those *common to all datasets* were used, resulting in a total of four experiments for each. This resulted in excluding 42% of the time points from the DC dataset, and 66% from the other two, which was far from ideal. Nevertheless, the purpose of the current exercise was to find a 'kernel' comparison base, for which to examine all three methodologies, and this is the basis for proceeding with the severely truncated datasets. As more data become available, relative performance may be assessed on more extensive and complete datasets [2].

The DEx tests assumed a linear model for the gene expression levels in the two microar-

[2] In the NGS dataset, genes with null counts in all time points, (11% of genes), were removed before DEx analysis.

ray datasets and a negative binomial distribution for counts in the NGS dataset. Based on data replicates, estimates of the expected mean and variance were obtained. The DEx test between two samples is based on the null hypothesis that the expression values of a gene in both come from the same distribution, with p-values obtained for each gene and sample pair ([Anders and Huber, 2010; Smyth, 2004] for more detail).

Sensitivity analyses, based on p-value thresholds for DEx, were performed for $0.000001 < p < 0.01$. These were used to assess how DEx sets of genes were reduced, and whether these were common across different datasets. A similar analysis was performed for log-ratios, (computed between each time point and the first one), as these can provide a more uniform criterion for selection of DEx genes across platforms. Given very similar results for thresholds based on p-values and log-ratios, only the former are discussed here.

Pair-wise comparison between the three datasets was performed, by computing the number of DEx genes in each dataset for each threshold and extracting common DEx genes. Given that some genes were not present in all datasets (as microarray probes differ between platforms, with some having missing values); these were removed from the analysis before each *pair-wise* comparison. Thus, when comparing datasets DC and SC, genes present in both datasets only were considered, whether or not present in dataset NGS. This resulted in eliminating the additional DEx genes from the first dataset that were not sampled in the second, to remove bias due to platform sampling range. On average, 70% of genes were retained between the DC and the other two datasets, while about 80% were retained for the NGS vs. SC analysis. While the full data might reasonably be expected to provide additional insight on the extended gene set by platform, truncation was required for the current study, which aimed to identify strictly overlapping data structures for eventual integration, rather than to provide a ranking of technologies.

As indicated in Appendix B, each of the three datasets contains at least three replicates for each time point. The NGS replicates are technical, while those from microarrays are biological. Given that technical replicates differ only in experimental setting, while biological diversity is not present, the number of differentially expressed genes in the NGS dataset may be inflated, due to variance over-estimation. However, using a pooled approach

[Anders and Huber, 2010] for variance estimation in the NGS dataset resulted in a very low number of differentially expressed genes. On investigating the coefficient of variation for replicates in all datasets, larger values were obtained for the NGS dataset, indicating that these technical replicates are not, in fact, very similar. In consequence, the non-pooled approach was used for these, although this may have resulted in a small bias increasing the number of DEx genes retrieved.

7.1.3 Clustering and evaluation measures

Further analysis involved clustering the DEx sets obtained for different thresholds, to analyse how the quality of the clusters is affected for threshold choice and different datasets. The analysis has been performed on the DEx sets corresponding to p-thresholds of 0.01 and 0.0001, using expression values for all time points available in the datasets, (i.e. values resulting from *limma* normalisation for microarrays, and RPKM [Mortazavi et al., 2008] values for NGS data). Again, p-thresholds were chosen for different level of test significance. Two clustering algorithms (provided by R software) were employed: K-means, with Euclidean distance, and biclustering using the Plaid algorithm; packages *flexclust* [Leisch, 2006] and *biclust* [Kaiser and Leisch, 2008] respectively. K-means was applied with the preset number of clusters ranging from 2 to 400 (with a step of 1 between 2 and 20 and a step of 10 between 20 and 400). This large range was chosen to explore different granularity levels, given that some datasets clustered contained over 8000 genes. The Plaid algorithm was applied 10 times for each dataset. The three datasets were standardised by experiment, (i.e. converted to standard scores) prior to clustering, to remove biases related to scale, that may differ from one time point to another, due to experimental differences. Additional clustering was performed with correlation K-means, (i.e. by using Pearson correlation r to compute a distance measure $1 - r$), but results were very similar and are not, therefore, discussed here.

To evaluate clusters obtained from each dataset and threshold, several criteria were used. For K-means, the Davies-Bouldin index (DBI) [Davies and Bouldin, 1979] was computed for each run, (with a different number of preset clusters), as this indicates whether

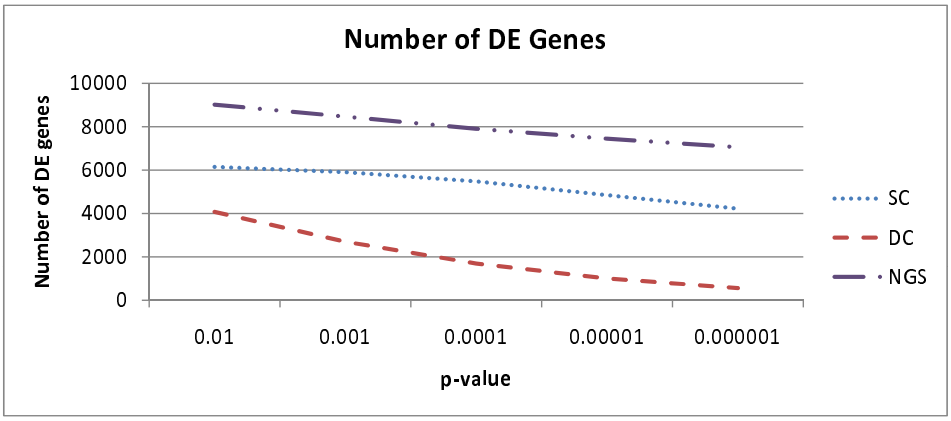


Figure 7.2: Number of differentially expressed genes for each dataset with various p-thresholds. The NGS dataset displays largest sensitivity to the DEx test, followed by SC and DC.

clusters are both well-defined and well-separated (a lower DBI value indicates *compact* and *distinct* clusters). For biclusters, the within-cluster variance was computed, using the *biclust* package. This gives an indication of the bicluster *compactness*, with lower variance corresponding to tighter groupings. Additionally, for both K-means and biclusters, the Biological Homogeneity Index (BHI) [Datta and Datta, 2006], based on Gene Ontology [Ashburner et al., 2000] annotations for molecular function (MF) and biological process (BPr) , was computed for all clusters (using package *clValid* [Brock et al., 2008]). The BHI represents the *percentage of gene pairs* in a cluster with *common annotation*, and allows for evaluation of cluster quality from the biological point of view (complementing the other evaluation criteria based on expression value distance measures alone).

7.2 NGS vs. microarrays: comparison results

7.2.1 Differential expression analysis

In the first analysis performed, DEx sets of genes, obtained from different datasets with different thresholds, were studied. Ideally, the gene sets should show significant overlap, and should be similar in size. Figure 7.2 shows the number of differentially expressed genes in each dataset, while Figure 7.3 shows percentages of common genes identified (for

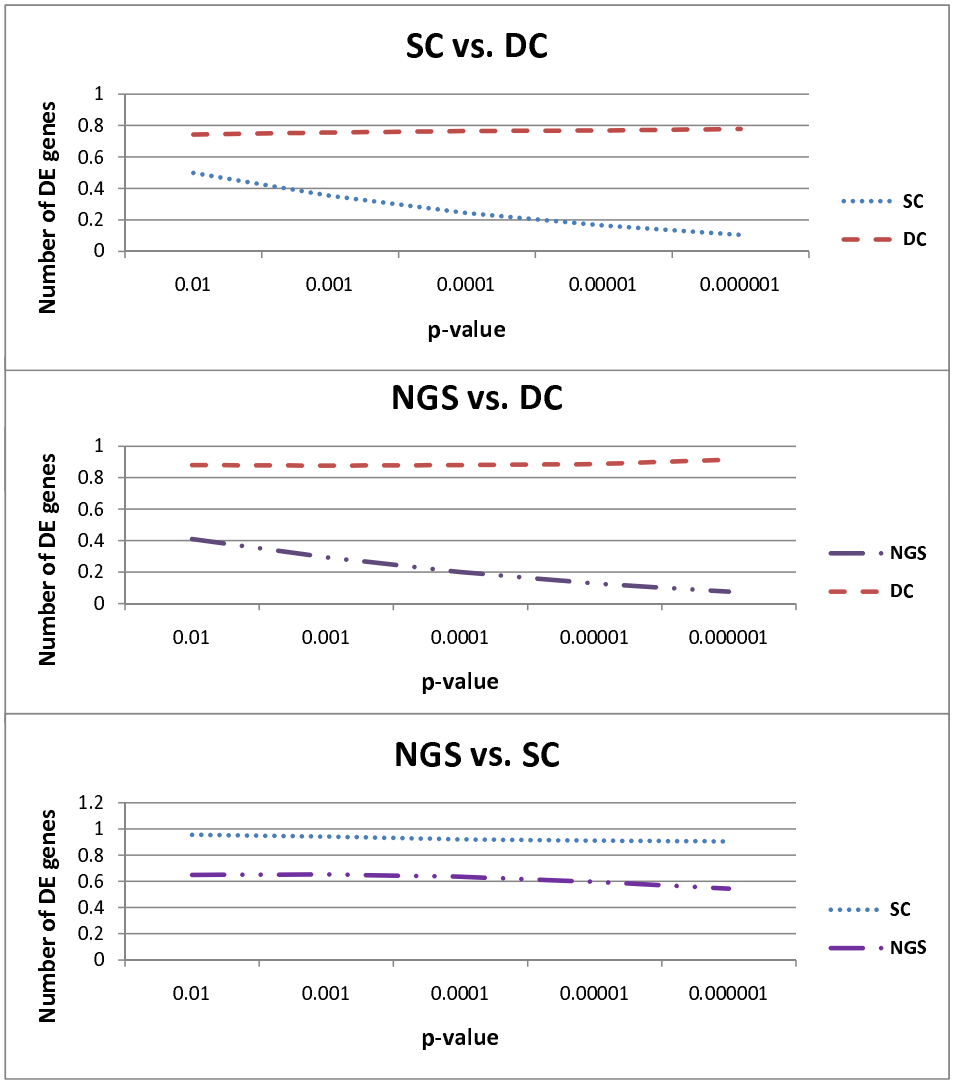


Figure 7.3: Percentage of common differentially expressed genes for dataset pairs with different p-value thresholds. For each pair of datasets, only the genes that exist in both datasets are considered. The percentage of common genes decreases always for the SC and NGS datasets, while for the DC dataset, which has the lowest number of DEx genes, it increases only slightly.

dataset pairs). The NGS dataset identified the largest number of genes, in agreement with previous studies, followed by SC and DC. Datasets SC and NGS show greatest similarity for the DEx sets obtained, with a large number of DEx genes involved, mostly common to both. Compared to this, the DC dataset captures only a restricted DEx gene set, implying that the NGS and SC datasets are more sensitive to the DEx test. Additionally, when more stringent p-thresholds are applied, a large number of DEx genes are retained for both, while a decrease is seen for the DC dataset. One possible explanation for this may be that the secondary channel function in DC data is known to smoothe out differences between time points. The large number of DEx genes in the NGS dataset may also be partly due to use of technical replicates; nevertheless, the SC dataset analysis (with biological replicates) also retrieved many DEx genes. This suggests that the variance estimation assumption for the technical replicates is reasonably robust. The fact that findings for the NGS dataset are in good agreement with those for the SC dataset also indicates good potential for microarray and RNA-seq data integration in future analysis. Similarity between RNA-seq and the Affymetrix platform has been also identified in previous studies [Bottomly et al., 2011].

When a lower p-threshold is applied, the percentage of DEx genes common across datasets is expected to increase, even if the DEx set cardinality decreases for individual datasets, since the more stringent threshold should act as a filter for *non common* genes. Unfortunately, this is not true for datasets SC and NGS, while for the DC dataset the percentage increase is very small, while the number of DEx genes decreases drastically. This suggests that the DEx information on some genes is less precise for *at least one* dataset of the pairing, regardless of thresholds used, probably due to different noise levels and/or other platform differences. This behaviour also occurs when the two microarray platforms are compared, however, so does not necessarily preclude NGS and microarray data integration (not least since dual- and single-channel data have been used in common studies, Chapter 4, [Sîrbu et al., 2010b]). It does indicate, however, that reducing noise remains a persistent issue in gene expression measurements.

The genes recorded as differentially expressed in the NGS dataset, but not in the SC dataset, were also further investigated, and Figure 7.4 displays the average count values,

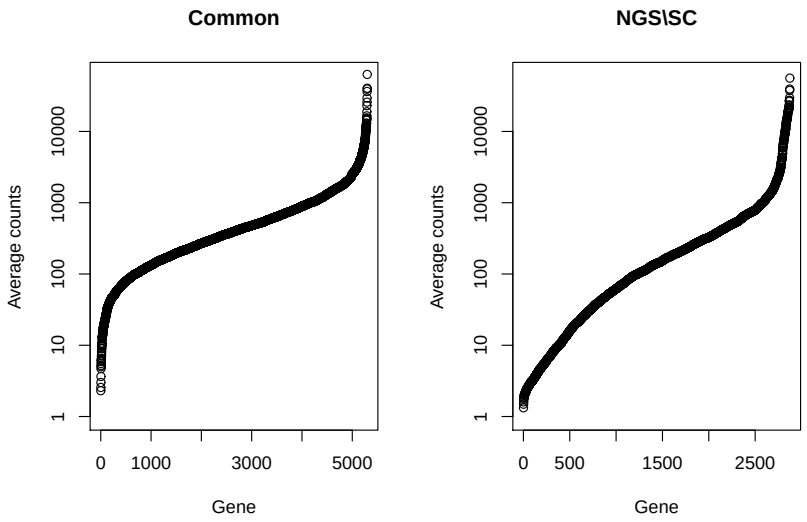


Figure 7.4: Average count values for genes commonly DEx in the NGS and SC datasets, versus those DEx only in the NGS dataset. Note that the vertical axis is in log-scale. Almost half of the genes that are DEx expressed in the NGS and not in the SC dataset display a very low number of counts (i.e. under 100), while more than three quarters have counts under 500. Only genes probed on both platforms were considered for this analysis.

(number of reads mapping to the specific gene), for differentially expressed genes in the common and additional categories. The graph displays results only for genes from the NGS dataset that also have matching probes in the SC dataset. A large number of genes, which are found only in NGS data, have correspondingly low counts, with nearly half occurring less than 100 times, and nearly three quarters less than 500. The fact that these genes were not identified from the SC dataset may be due to background noise interference in microarrays, which hinders correct quantification of rare transcripts, while this problem does not exist in RNA-Seq, giving the latter technology an advantage in handling low expression values. Some highly expressed genes are also missed by microarray analysis, and this might be due to probe saturation, again not present in RNA-seq data. Previous studies have also reported higher DEx sensitivity of RNA-seq for large copy-number transcripts [Bloom et al., 2009]. However, this property has not been previously identified for low count transcripts also [Liu et al., 2011], although supported by known characteristics of the different technologies. This might be due to the sequencing depth used in these previous studies [Bloom

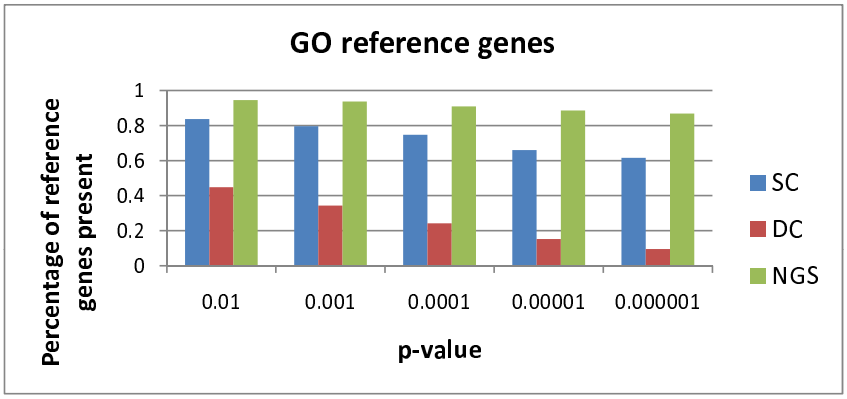


Figure 7.5: Percentage of reference genes represented in the DEx sets obtained from the three datasets with different p-value thresholds. The NGS dataset identifies the largest number of reference genes, and the DC dataset the lowest.

et al., 2009], which is lower than for the NGS dataset. For instance, [Liu et al., 2011] report an average of 11.56 RPKM for their study, while the NGS dataset contains an average of 43.2 RPKM, significantly larger.

A reference set of genes, considered highly likely to be differentially expressed during embryo development, was then selected in order to test whether these were identified from the three datasets. This set of genes consisted of those annotated with the *embryo development* term in the Gene Ontology database (641 genes). Figure 7.5 shows the proportion of these genes identified from each dataset, together with the different threshold values that apply. Genes that were missing from the three datasets were eliminated before computing proportions, (which are thus based, respectively, on 69, 51 and 72% of the 641 genes actually present in the SC, DC and NGS datasets). Even though the total number of DEx genes differs, with the DC dataset identifying very few compared to the other two datasets, this *reduced reference set* of genes is expected to be highly represented in all, even when low p-value thresholds are used. In fact, the DC dataset identifies the lowest percentage of reference genes, decreasing further for low p-values, while the NGS dataset identifies the highest, with over 80% of reference genes present even for the most stringent threshold, again indicating an advantage over the microarray data.

7.2.2 Clustering of differentially expressed genes

Two clustering algorithms were applied to the DEx gene sets, (for different p-value thresholds). When the more stringent threshold is used ($p < 0.0001$), clusters are expected to be well-defined and well-separated (i.e. to have smaller DBI and variance). Similarly, BHI scores should increase, especially when a larger number of clusters is obtained, as those DEx genes included in the same cluster, under these conditions, should share similar processes or function. The rest of this section describes scores obtained for two alternative clustering methods, in order to investigate this hypothesis, and provide insight on the data structure for the three datasets.

7.2.2.1 K-Means

In order to analyse the quality of clusters from each dataset, (in terms of compactness, separation and biological relevance), and the structure of the gene space, four criteria have been used. These provide complementary information on the data space, so are considered together for a complete view. Values obtained for the three DEx datasets (NGS, SC, DC) corresponding to $p < 0.01$ are displayed in Figure 7.6. A first analysis of cluster quality studied numerical separability of groupings obtained. The DBI (Davies-Bouldin index) values for clusters obtained by K-means with the number of clusters (k) ranging from 2 to 400, are displayed in Figure 7.6(a). For a better view over the data space, the size of clusters for selected k was determined for all three datasets (Figure 7.6(b)). The biological relevance of groups obtained was analysed also. Ideally, clusters of 'good quality' (DBI criterion) should contain genes involved in similar processes or with similar functions. Figure 7.6 includes BPr (Biological Process) BHI (Biological Homogeneity Index) values for clusters obtained with different k for: NGS (c), SC (d) and DC (e). The boxplots for each cluster analysis show the distribution of the homogeneity index for clusters obtained. Additionally, the proportion of genes in each dataset, included in clusters with BHI larger than 0.1, was computed for different k values (Figure 7.6(f)).

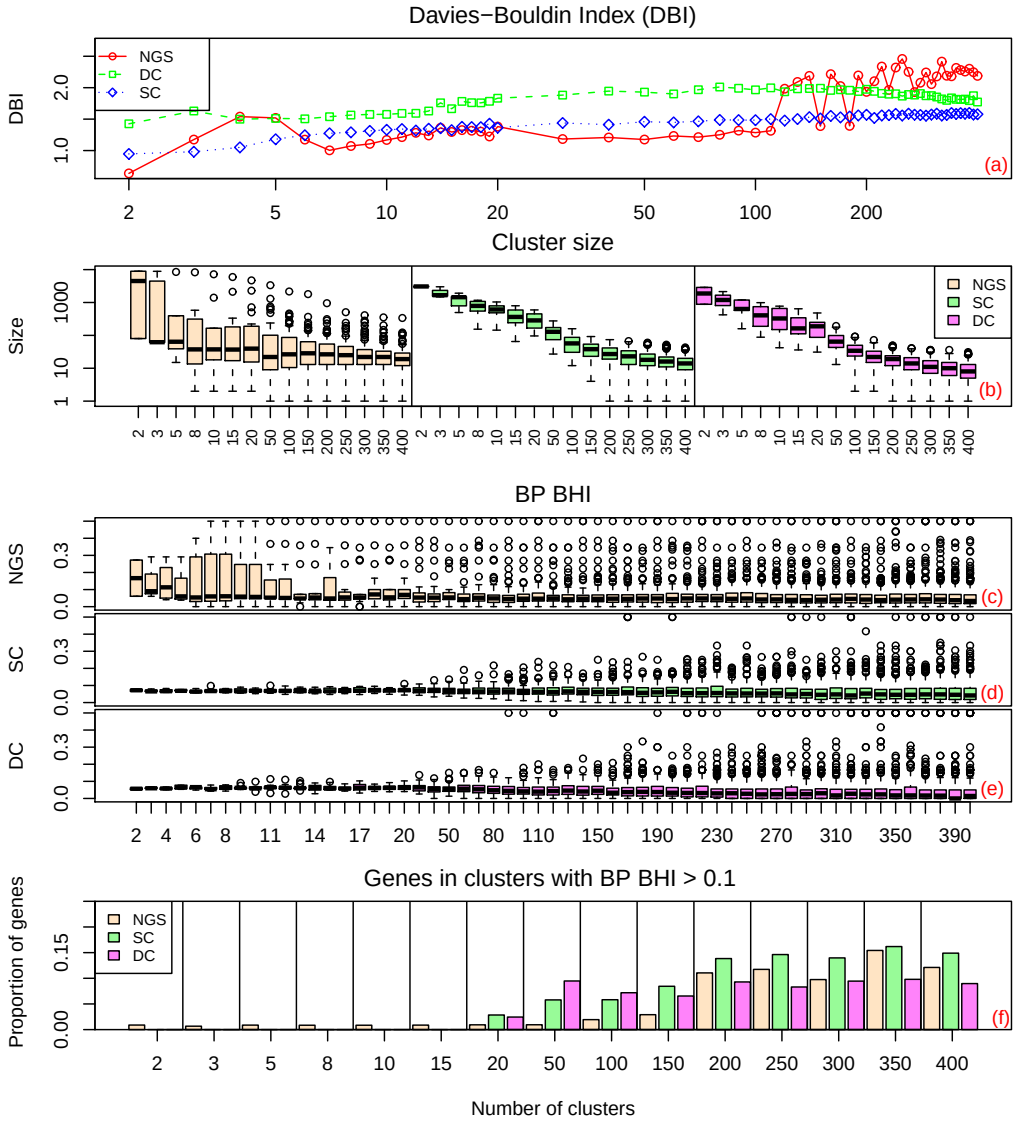


Figure 7.6: K-means clustering evaluation for $p < 0.01$. Six graphs are included with values for four evaluation criteria (vertical axis) with different number of centroids, k (horizontal axis): (a) displays DBI values obtained for the three datasets with different k; (b) shows the range of cluster sizes in each dataset, for selected number of centroids; (c), (d) and (e) display boxplots of the homogeneity index (BHI) for biological process annotation of genes in each cluster, with different k; (f) features the proportion of genes that are included in clusters with BHI larger than 0.1 in each dataset for different k. The graphs show that for small k, the NGS displays a different data space structure, with small gene islands with better homogeneity, that are not visible in the other two datasets. For large k, results are more similar among the three datasets, as more and more clusters with larger BHI are present with the increase of k.

K-choice: For few predefined centroids ($k \leq 15$), the NGS, SC and DC datasets display different high-level features. DBI values (Figure 7.6(a)) indicate similar numerical separability for the first two, which is slightly better than that for the DC dataset. However, cluster size and BHI criteria (Figure 7.6 (b,c,d,e)), indicate a different structure for NGS across the gene space. Cluster sizes are highly *heterogeneous*, compared to those for the two microarray datasets (SC and DC), for which size-range is limited. This suggests that, in the NGS case, the presence of a set of well-separated small gene clusters, which are more biologically homogeneous, is highlighted, in addition to larger, more heterogeneous groupings. This corresponds to an extended distribution of BHI values across clusters for these data, (Figure 7.6(c)), with around 90 genes included in clusters with homogeneity better than 10%. These *small compact clusters* do *not* appear in the two microarray datasets (SC and DC), which exhibit small BHI values for all clusters, (Figure 7.6(d,e)), and have no genes included in groups with BHI greater than 0.1 for low k (Figure 7.6(f)). The high scoring groups in the NGS dataset correspond to genes that exhibit extreme (low or high) expression values at some time points. It appears that these are not as well captured by the microarray technology, possibly due to background noise (for low expression values) and hybridisation saturation (for high expression values).

Clustering for large k, results in cluster size distributions similar to those for small k. The NGS dataset continues to include both large heterogeneous and small homogeneous clusters, while the two microarray datasets produce groups reduced in size, but comparable within each dataset. There is some indication that, at this finer-granularity, more small compact clusters are also detectable, as might be expected. The NGS dataset displays best separability (DBI) up to about 100 centroids, a threshold at which the larger well defined groups start to sub-divide, (so that separability decreases significantly - Figure 7.6(a)). However, homogeneous clusters (i.e. with larger BHI values), increase in number as more small clusters are identified in all three datasets (ref. increased number of outliers in Figure 7.6(c,d,e)). This is also demonstrated, for increased k, by the proportion of genes included in clusters with better than 10% homogeneity. Again, the DC dataset performs least well in this regard, while the SC dataset performs best. These results indicate that for fine-grained

clustering (large k), the data space for the three datasets is closer in structure compared to that for low k, with the difference (again) that the NGS dataset retains a few very large gene clusters, while remaining genes are grouped comparably to SC and DC.

P-threshold: To analyse the effects of using a tighter threshold for p-values, similar criteria were studied for the DEx sets obtained with $p < 0.0001$, (specific values displayed in Figure 7.7). These show that for low k, clusters obtained for all datasets are very similar to those found for the less stringent p-threshold, indicating that the gene space is not much affected at high-level by this filtering by p. For high k, an increase in cluster separability and compactness both from the biological and numerical point of view is expected, compared to the previous p-value, with the proportion of genes in homogeneous clusters increasing or at least stable. The NGS and SC datasets, however, maintain similar DBI and BHI values, which may be due to partial filtering only at the low threshold, as seen in Section 7.2.1. However, for the DC dataset, choice of p-threshold has a large effect (Figure 7.7(a)), with clear DBI pattern changes, as expected, and improved cluster separability as k increases. However, BHI values decrease, indicating that, while more distinct clusters are obtained, groupings are *less biologically plausible*. This is a real concern and may be due to the inclusion of the *reference sample* in these data, which tends to smoothe out differences between time points, resulting in both a low number of differentially expressed genes and a transformed gene space. Additionally, it is important to note that, at this more stringent p-threshold, the proportion of genes included in clusters with BHI over 0.1 *decreases* in general for the DC dataset, while it increases for the NGS and SC datasets. This suggests that in the DC data, the low threshold filters out genes both from the homogeneous *and* less-homogeneous clusters. Homogeneous groupings are less affected for the other two datasets, with genes mostly filtered out from heterogeneous groupings, (as desired). This, together with clustering results observed for large k with the previous p-value threshold (0.01), indicates considerable similarity between the NGS and SC dataset at fine-grained level.

It is important, however, to note that BHI values for all datasets rarely exceed 0.3,

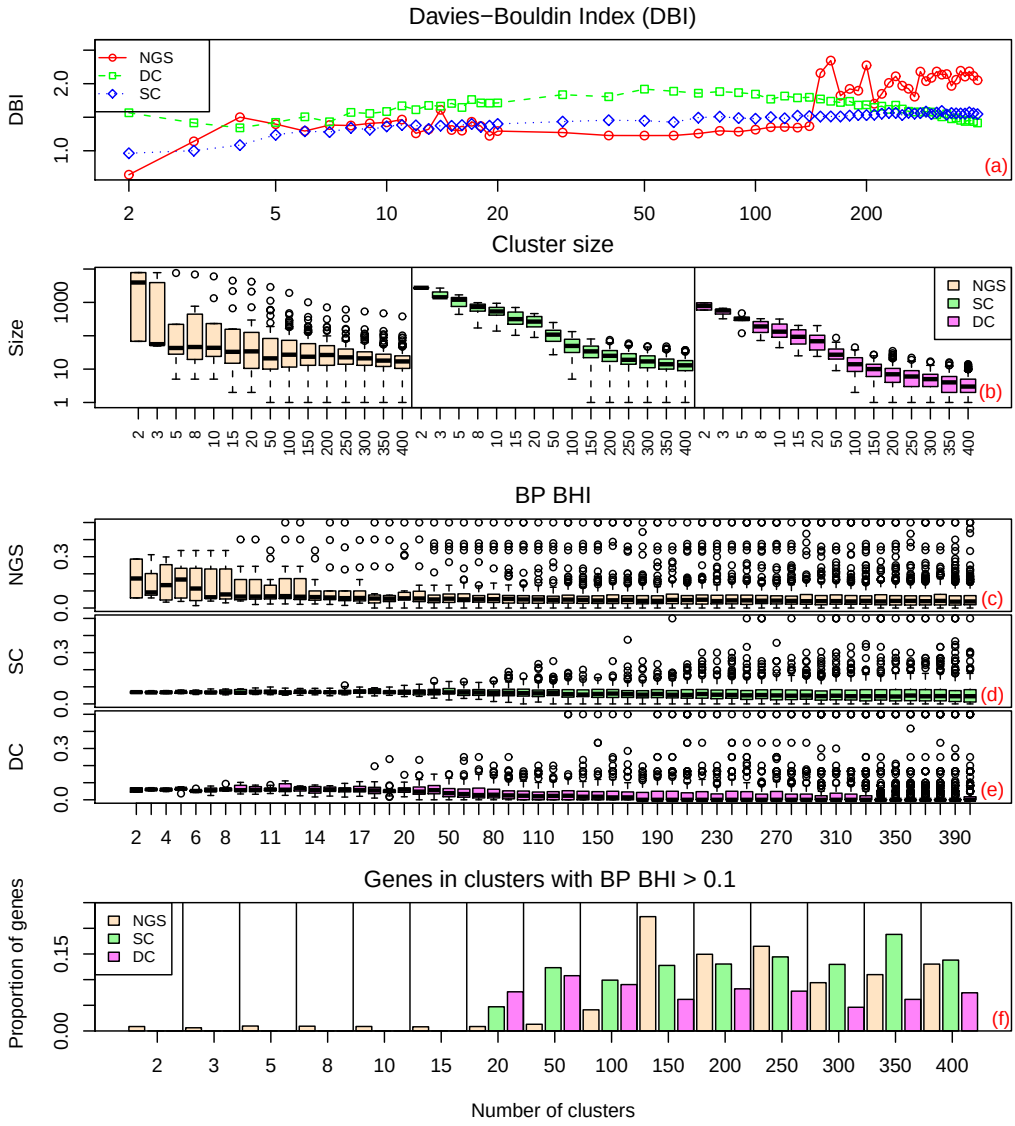


Figure 7.7: K-means clustering evaluation for $p < 0.0001$. The same criteria as Figure 7.6 are displayed, for DEx datasets with a more stringent p-value threshold. The same structure of the data space is present. Additionally, for this p-threshold, the SC and NGS dataset assign a larger proportion of genes to clusters with BHI> 0.1, while the DC dataset the proportion decreases compared to the previous threshold, indicating that the more stringent test does not improve biological homogeneity of clusters obtained.

which indicates only *moderate biological homogeneity* of clusters. However, this is also found for previous K-means clustering analyses for wild-type gene expression data, [Kerr, 2009; Datta and Datta, 2006], and would still support the conclusion that the NGS dataset displays better resolution for coarse-grained clustering (low k), while SC and NGS datasets are similar at finer granularity. Similar results have been obtained using correlation-based K-Means, confirming that the results obtained are due to the structure of the search space, and not the distance measure chosen.

7.2.2.2 Plaid

Plaid biclustering results for repeated runs are displayed in the form of boxplots (Figure 7.8). These show average within-cluster variance, cluster size, number of clusters and MF BHI values for biclusters found over ten runs, with DEx datasets corresponding to $p < 0.01$. For K-Means analysis, BPr (biological process) BHI values were the most significant of the three possible annotations, (cellular component, molecular function and biological process). However, the Plaid algorithm appears to identify better groups of genes with similar molecular function (MF), hence MF BHI values are presented in this section.

As the figure shows, the SC dataset clusters display lowest variability, and the NGS highest. The NGS data include a wide range of cluster sizes, in agreement with K-Means results, and display similar space structure, (namely small, distinct 'islands', together with very large gene clusters). However, the SC dataset also displays similar cluster structure, with larger size-range compared to the DC dataset, again indicating similarities for NGS and SC data. It also appears that the SC dataset contains the best-defined biclusters, both from the numerical (low variance) and biological point of view, with best annotation homogeneity on average. However, the NGS dataset contains more homogeneous clusters (represented by the maximum and outliers in the boxplots), even though most of them have somewhat lower BHI values. The DC data space is more compact, with clusters having similar size-range and lower BHI values compared to SC.

Figure 7.9 shows values for similar criteria, for DEx datasets obtained with $p < 0.0001$. For this threshold, the SC dataset again displays, on average, lowest within-cluster variance

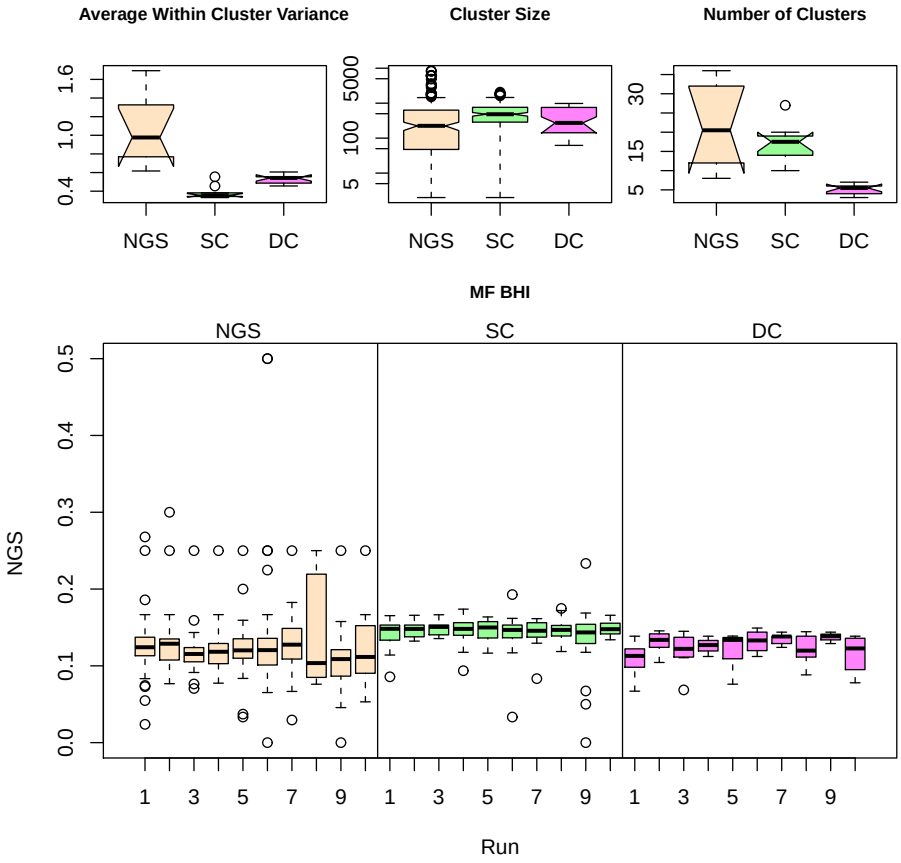


Figure 7.8: Bicluster properties: average additive variance, size and number of clusters, and BHI distribution of values in the ten runs performed, for DEx datasets corresponding to $p < 0.01$. These suggest the same space structure for the NGS dataset, with both large and small islands with correspondingly small and large homogeneity. However, here, the SC dataset shows similar behaviour, as a similar distribution of cluster sizes is present, but clusters are more compact (lower variance). The DC dataset displays a smaller range for cluster sizes, and lower BHI values compared to SC.

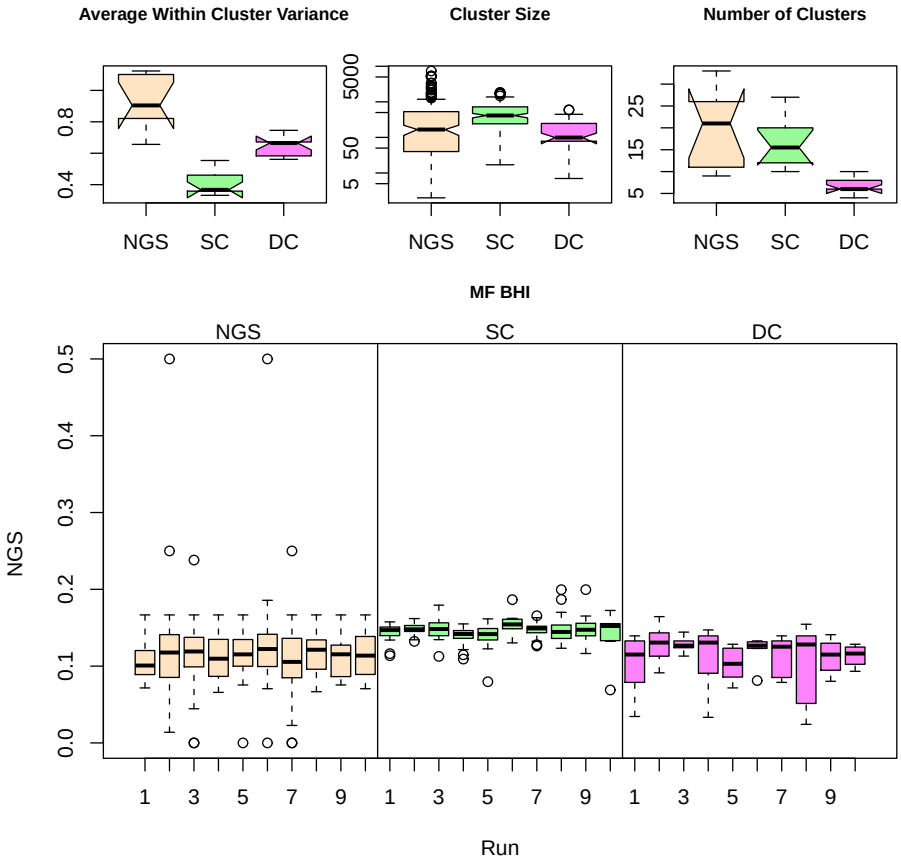


Figure 7.9: Bicluster properties for $p < 0.0001$. The same criteria as Figure 7.8 are displayed. These show again better cluster homogeneity and compactness for the SC dataset, and similar structure of the data space as for the previous p-value, for all three datasets.

and largest homogeneity. For the NGS dataset, a more stringent DEx threshold appears to increase the robustness of the Plaid algorithm, (i.e. results in slightly reduced range for variance and cluster numbers). This indicates improved separability of clusters, (also suggested by the slightly lower variance). BHI values are also more stable between runs, with fewer outliers and with average values close to those obtained with the previous p-threshold. Again, a few clusters with higher BHI compared to those found for the SC dataset are also present. The DC dataset loses biological homogeneity when a more stringent p-threshold is applied, as more clusters with low BHI are identified. Again, this is in agreement with the K-means results, discussed earlier.

7.3 NGS vs. microarrays: Conclusions

An analysis of three types of gene expression data for *Drosophila melanogaster* embryo development time series was presented; these include both dual- and single-channel microarrays, and RNA-seq. The aim was to identify similar and complementary features of these datasets, with a view to investigating the potential for future data integration from the three platforms. A sensitivity analysis was employed to study the sets of differentially expressed (DEx) genes obtained with different p-value thresholds, and, subsequently, to assess cluster quality on applying two clustering algorithms: Euclidean K-means and Plaid biclustering.

Differential analysis indicated that the NGS and SC datasets are more sensitive to the DEx test, with large numbers of DEx genes identified, in contrast to findings for the DC dataset. However, agreement on which genes are DEx is not comprehensive, even for low p-thresholds. The highest commonality is found between the NGS and SC datasets, with lowest between NGS and DC. These findings are in agreement with previous studies of differential expression, [Bloom et al., 2009], although those have been performed in a less broad setting, i.e. by using the same sample for all experiments. This suggests that integration of highly heterogeneous datasets may be feasible. Many of the additional DEx genes in the NGS dataset have relatively low expression values. Additionally, some very abundant

genes have been identified only by RNA-seq data. This suggests that, as postulated, the new technology has an advantage in quantifying extreme transcription levels.

K-Means clustering indicated that specific features of the data space are different for the three datasets. *High level* analysis, (clustering with a small number of predefined centroids, k), highlighted differences between the NGS and the two microarray datasets, in particular. For the former, small groups of genes are clearly identified, distinct from larger groupings, and usually correspond to similar biological process GO annotation, (BPr BHI). For the two microarray datasets, such small gene islands are not identifiable and large heterogeneous gene groups are present, resulting in poorer cluster annotation homogeneity.

For *large* k, the NGS and SC datasets exhibited an increase in the number of clusters with similar annotation, but at the expense of considerable heterogeneity in others. This was true for all p-value thresholds. The DC dataset, in contrast, displayed lower annotation homogeneity for large k and low thresholds, although numerical separability (Davies-Bouldin Index) seemed to improve. One explanation is that the coherence of this dataset may suffer from the presence of the secondary channel, which over-smoothes differences between time points.

Clustering with the Plaid algorithm confirmed the existence of small islands in the NGS data, with correspondingly large homogeneity values. However, the SC dataset displayed similar data structure, indicative again of some feature overlap between the two datasets.

In conclusion, *similarities between the NGS and SC* datasets have been identified, both for *differential expression* and *fine-grained* (large k) clustering results. For coarse-grained clustering (small k), the NGS dataset appears to provide more information, since small gene islands with similar annotation (containing genes with extreme expression values), are readily identifiable, which is not the case for the SC dataset. These similarities suggest that integration of NGS and SC for further analyses is feasible and may have extendable complementarity (e.g. to less directly-overlapping datasets). Moreover, the possibility of cross-platform analysis for all three types of data is not excluded, given that single and dual-channel microarrays have been analysed in common before. However, indications are that DC results are more useful for identifying a restricted gene set, with limited information on

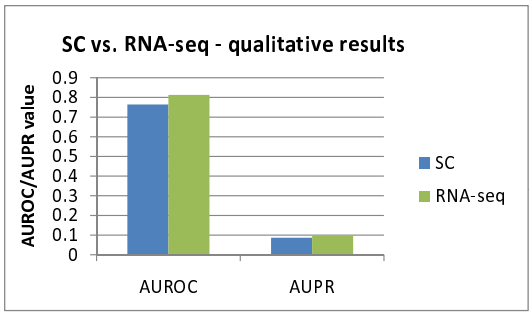


Figure 7.10: Qualitative evaluation of models trained with NGS and SC data. The graph shows AUROC and AUPR values on a set of known interactions from the DROID database, for interaction sets obtained after 10 runs of the algorithm, using NGS and SC data for training. Models trained with NGS data display a larger number of correct interactions.

groupings, while NGS offers a refined probe for identifying smaller gene groupings with extreme expression levels. This study, however, is a preliminary assessment of similarity between such heterogeneous datasets. Further analyses should assess DEx sensitivity to q-value thresholds, and clustering of DEx sets obtained. This would not change the ranking of genes, but would, however, reduce the cardinality of DEx sets.

7.4 NGS data for EGIA

In order to analyse how the EGIA framework extends to RNA-seq data, the inferential algorithm has been applied to the NGS dataset for inference of a 27-gene regulatory network. Following the analysis presented in Chapter 6, we have applied the *BS-eval* version of the algorithm, which used binding site affinity to enhance model evaluation and annotations, correlation patterns and knockout experiments to customise the mutation operator of the evolutionary algorithm. The same evaluation criteria used in the previous chapter, i.e. AUROC/AUPR on 16 known interactions from the DROID database (Section B.3) and RMSE on test data have been employed here as well. The NGS dataset has been used for training, and the two microarray datasets (SC and DC) for testing. Prior to inference, *XPN* cross-platform normalisation ([Shabalin et al., 2008], Section 4.3.1) has been applied to the three datasets.

Figure 7.10 displays AUROC/AUPR values obtained after 10 runs of the algorithm, us-

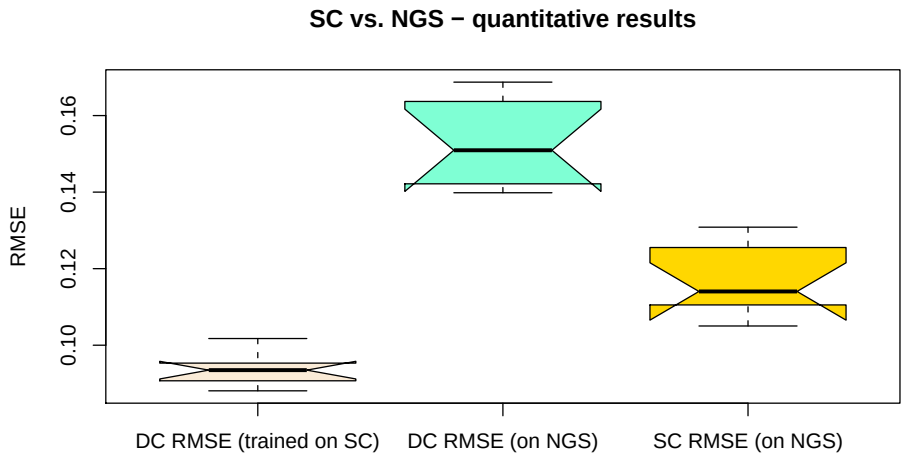


Figure 7.11: Quantitative evaluation of models trained with NGS and SC data. Box-plots represent the distribution of RMSE values on test datasets (SC and DC) for models trained on the SC and NGS datasets. Models obtained from SC data display a lower error when applied to simulate DC data, compared to models obtained from NGS data. Testing the NGS-inferred models on SC, as opposed to DC data, results in lower RMSE values.

ing the NGS and the SC dataset for inference. This shows that the NGS dataset highlights more correct interactions compared to the SC dataset. It is important to underline the fact that the two datasets contain exactly the same number of time points for training, resulting in the same level of determination of the system, with the difference that, for the NGS dataset, the time-span between points is twice as large as the one for the SC dataset. The improved quantitative behaviour for models obtained from NGS data suggests that these data may prove to be less noisy and have an important impact in gene expression quantification for discovery of regulatory interactions. Together with the more mature technologies, this should enable more reliable reverse engineering of such networks. For instance, one straightforward integration methodology, at the qualitative level, would be to look at the union of the interaction sets obtained from each of the available datasets and sum the votes obtained for each interaction. In the case of SC and NGS data, analysed here, this union results in AUROC/AUPR values of 0.873/0.105 respectively, higher than using each dataset individually (indicating that integration is useful from the qualitative point of view at least).

Quantitative results are displayed in Figure 7.11, which show the distribution of RMSE (root mean squared error) on test data simulations (for the DC and SC datasets). Results for models obtained in 10 runs using the NGS dataset for training are displayed. For comparison, the RMSE obtained on the DC dataset by models inferred from SC data is also provided in this figure. The values show that models obtained from SC data are better able to simulate DC data, compared to those obtained from NGS data. This is probably due to the fact that the single- and dual-channel microarray platforms are more alike than the dual-channel and the NGS platform, and suggests work on a new cross-platform normalisation method, specifically tailored for microarray and NGS data integration. Nevertheless, the RMSE values obtained, along with the AUPR/AUROC values, do provide an initial indication that the EGIA framework can be successfully applied to RNA-seq data.

When simulating SC, as opposed to DC time series, the models trained with NGS data generate smaller errors. This also indicates closer similarity between the NGS and SC data compared to NGS and DC, supporting the tentative conclusions of the previous section. Given the qualitative results discussed above, an integration of the different time series may prove beneficial; however, for quantitative integration, strict attention to normalisation criteria is likely to be required.

Chapter 8

Concluding discussion and future work

8.1 Summary and conclusions

Gene regulatory networks (GRNs) are an important mechanism for protein level control, having a major role in most organism processes. Determining GRNs is thus crucial to understanding organism behaviour and finding disease markers and treatments. Mathematical modelling is one tool used to analyse gene expression and regulation. To date, although extensive work has been performed in this field, quantitative modelling is still limited, due to data constraints such as noise and reduced length of time series. In this work, a detailed analysis of existing methods was provided, focusing on evolutionary computation. Given the limited power of current methods and models, new algorithms and criteria are needed to enhance model inference. One possibility is integration of different data types, which are widely available.

Here, we presented an analysis of data integration for quantitative model discovery. A first step was combining cross-platform microarray time series from different sources. Secondly, a novel inferential framework that includes different types of data and knowledge was introduced. Thirdly, new gene expression data from RNA-seq measurements were

analysed in comparison to microarray data to identify integration prospects at this level.

Main findings

1. The first part of the thesis provided an in-depth analysis of the state of the art in GRN modelling. The *role of evolutionary computation* at different modelling stages was outlined, and advantages such as flexibility, stochasticity and implicit parallelism were shown to have a major role in data integration. A further detailed comparison of seven evolutionary algorithms for quantitative model inference was performed. This showed that classical approaches have reduced power, hence can be used for small-scale analysis only, and that *hybrid methods* are required in order to cope with larger-scale and data limitations. However, to date, issues related to under-determination and applicability to real datasets still exist.

2. The *first step* towards data unification, i.e. *cross-platform integration of time series data*, indicated that such datasets are compatible and that models based on combined data exhibit better fit for test datasets and are more robust to noise and parameter perturbations. Furthermore, a wavelet analysis showed that combined datasets lead to less noise over-fitting. An analysis of cross-platform normalisation techniques demonstrated that quantitative integration of such data is enhanced by applying correct pre-processing.

3. A novel *integrative framework* based on evolutionary computation was developed and presented here (EGIA). This was implemented in C++ and takes advantage of parallel computing to increase the inferential power, (networks of different sizes, up to 100 genes have been studied). The novelty of the framework rests on the *additional data* that can be incorporated in the analysis (i.e. time series, knockout experiments, binding site affinities, known cis-regulatory modules and Gene Ontology annotations) and in the *flexibility* offered by allowing integration at different algorithm stages.

4. One enhancement of the EGIA framework is to *customise initialisation and mutation* in order to facilitate more *knowledgeable exploration* of the solution space. This was

shown to result in *better topology* of the resulting GRN models. However, data fit was *not improved*, especially in the case of real networks. This suggested that enhanced exploration only is not enough for obtaining better models, due to the noise in the data and the *strongly data-dependent evaluation criterion*.

5. A *customised evaluation criterion* was derived, aiming to address problems such as noise and under-determination, by addition of two further evaluation terms. *Correlation* inclusion was shown to result in better structures and data fit even when random mutation and initialisation were used. The second evaluation term introduced reflected *additional data types* used to assess the plausibility of the candidate network topologies. This was benchmarked on three datasets and shown to have a significant positive effect on *both* qualitative and quantitative value of resulting models. Improvement was obtained after a *consistent* analysis of the *error structure* of the data.

6. The *error structure* analysis of additional data that can be used for inference identified a set of important aspects (by repeating the reverse engineering process using different subsets of these data). Firstly, it showed that, due to data variability, not all additional data add value. This underlined the need for *careful analysis* of datasets, as opposed to just combining all data available. Secondly, it has demonstrated the value of the EGIA framework, which is *flexible* enough to allow for different configurations of datasets for integration. This is possible due to the existence of the two integrative mechanisms, within *exploration* and *evaluation*, with different resilience to noise in the data. Specifically, the former provides a more *flexible* integration than the latter, so that less reliable data can still be used, to extract some positive effects. Extended evaluation proved to be crucial in order to observe improvements *both* quantitatively and qualitatively, but at the same time less resilient to bias in the data. Thirdly, the analysis has built a general methodology for GRN inference and validation, where data is separated into training and test during the evaluation of the error structure and then integrated using the best procedure identified. As data become more abun-

dant and of better quality, due to technical advances, this framework has potential for building GRN models.

7. The *third step* in data integration involved inclusion of *RNA-seq* measurements for inference purposes. The analysis presented showed that microarray and RNA-seq data do have overlapping features, especially in the case of single-channel microarrays. Additionally, the EGIA framework was successfully applied to NGS data, resulting in *better prediction* of previously known interactions, compared to microarrays. *Qualitative integration* of the predicted interactions (based on model inference from single-channel and RNA-seq data separately) indicated that the unified set of interactions is better than each of the two, suggesting the presence of both *overlapping* and *complementary* features. This is very important in guiding future quantitative integration of these data, which is likely to prove particularly challenging to achieve.

8.2 Future work

Following the findings presented in the previous section, several directions for extension of the framework can be identified.

Quantitative integration of RNA-seq and microarray time series We have shown that RNA-seq and microarray data display overlapping features, especially with respect to single-channel microarrays (Chapter 7, Section 7.2). Furthermore, the union of the predicted sets of interactions from one RNA-seq and one single-channel time series dataset resulted in improved quality of inferred connections. This gives good prospects for quantitative time series data integration. However, as the experiments presented in this thesis showed, quantitative compatibility between datasets is not as high as for microarrays (i.e. models inferred from one data type translate poorly to the other data types), indicating that more advanced cross-platform normalisation techniques are required, which take into account platform specific features of these data.

Multi-objective optimisation The customised fitness function of the evolutionary algorithm (Chapter 5) can be easily transformed into separate objectives, corresponding to each of the three terms. Multi-objective optimisation can be used to minimise these, as this can have an improved performance over single-objective search [Koduru et al., 2004]. The flexibility of this optimisation type would allow for further additional objectives to be added, to account for model robustness to perturbation or known properties of GRNs.

Development of a user interface and visualisation Computational tools are, nowadays, critical for analysis of biological data, and one important requirement for these is usability by non-computational scientists. With this in mind, we plan to develop an intuitive graphical interface to the EGIA algorithm. Additionally, exporting the resulting models and networks in standard formats, such as Cytoscape, is planned, in order to facilitate integration with other available tools, which are currently widely used in biological research.

Framework extension to a multi-scale model The EGIA framework currently makes use of an Artificial Neural Network to model GRNs; however, hybrid models, such as that introduced by [Kirkilionis et al., 2011a] (Chapter 3, Section 3.2) exist. This model represents more accurately the molecular interactions involved in gene regulation, as it models binding sites, cooperative binding and loop formation, which is an advantage over other quantitative models. However, this additional level of detail makes inference more challenging. Based on the framework introduced here, we plan a collaboration to extend the algorithm for this hybrid model. The number of binding sites and associated transcription factors can be optimised in a similar manner as the structure search performed in EGIA, with an additional nested optimisation stage to infer continuous-valued parameters. Additional knowledge can be used to enhance the structure optimisation phase, both as metadata for space exploration and in evaluating candidate network topologies.

Modelling RNA interference Transcriptional regulation is one of the many mechanisms of controlling protein levels in cells, which happens at the beginning of the gene expression process. An additional regulatory effect appears before translation, when short strands of

RNA (small interfering RNAs - siRNA- or micro RNAs - miRNA) hybridise to the mRNA and stop translation. This is known as RNA interference, and models presented here do not take it into account when modelling gene expression. However, this mechanism has been identified as being involved in many processes and diseases. Taking it into account in the modelling process may be expected to lead to models with higher resemblance to real gene expression.

Bibliography

Akutsu, T., Miyano, S., and Kuhara, S. (2000). Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, 16(8):727–734.

Alvarez-Buylla, E. R., Benitez, M., Davila, E. B., Chaos, A., Espinosa-Soto, C., and Padilla-Longoria, P. (2007). Gene regulatory network models for plant development. *Current Opinion in Plant Biology*, 10(1):83 – 91.

Anders, S. and Huber, W. (2010). Differential expression analysis for sequence count data. *Genome Biology*, 11(10):R106.

Ando, S. and Iba, H. (2003). Estimation of gene regulatory network by genetic algorithm and pairwise correlation analysis. *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*, 1:207–214.

Ando, S., Sakamoto, E., and Iba, H. (2002). Evolutionary modeling and inference of gene network. *Information Sciences*, 145(3-4):237–259.

Aoki-Kinoshita, K. F. and Kanehisa, M. (2007). Gene annotation and pathway mapping in KEGG. *Methods in Molecular Biology*, 396:71–91.

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

Baeck, T., Fogel, D. B., and Michalewicz, Z. (2000). *Evolutionary Computation 1: Basic Algorithms and Operators*. Institute of Physics Publishing, Bristol and Philadelphia.

Baldi, P. and Hatfield, W. (2002). *DNA Microarray and Gene Expression. From experiments to data analysis and modeling*. Cambridge University Press, Cambridge, UK.

Bar-Joseph, Z. (2004). Analyzing time series gene expression data. *Bioinformatics*, 20(16):2493–2503.

Barrett, T., Troup, D. B., Wilhite, S. E., Ledoux, P., Evangelista, C., Kim, I. F., Tomashevsky, M., Marshall, K. A., Phillippy, K. H., Sherman, P. M., Muertter, R. N., Holko, M., Ayanbule, O., Yefanov, A., and Soboleva, A. (2011). NCBI GEO: archive for functional genomics data sets - 10 years on. *Nucleic Acids Research*, 39(suppl 1):D1005–D1010.

Bergman, C. M., Carlson, J. W., and Celniker, S. E. (2005). Drosophila DNase I footprint database: a systematic genome annotation of transcription factor binding sites in the fruitfly, *Drosophila melanogaster*. *Bioinformatics*, 21(8):1747–1749.

Bernard, A. and Hartemink, A. J. (2005). Informative structure priors: joint learning of dynamic regulatory networks from multiple types of data. In Altman, R. B., Dunke, A. K., Hunter, L., and Klein, T. E., editors, *Proceedings of Pacific Symposium on Biocomputing*, pages 459–470, Duke University, Dept of Computer Science, Durham, NC 27708, USA. World Scientific.

Bloom, J., Khan, Z., Kruglyak, L., Singh, M., and Caudy, A. (2009). Measuring differential gene expression by short read sequencing: quantitative comparison to 2-channel gene expression microarrays. *BMC genomics*, 10(1):221.

Boot, J. C. (1964). *Quadratic programming. Algorithms - Anomalies - Applications*. Rand McNally & Company, Chicago, Illinois.

Boslaugh, S. and Watters, P. A., editors (2008). *Statistics in a Nutshell*. O'Reilly Media, Inc., Safari Books Online.

Bottomly, D., Walter, N. A. R., Hunter, J. E., Darakjian, P., Kawane, S., Buck, K. J., Searles, R. P., Mooney, M., McWeeney, S. K., and Hitzemann, R. (2011). Evaluating gene expression in c57bl/6j and dba/2j mouse striatum using rna-seq and microarrays. *PLoS ONE*, 6(3):e17820.

Bradford, J., Hey, Y., Yates, T., Li, Y., Pepper, S., and Miller, C. (2010). A comparison of massively parallel nucleotide sequencing with oligonucleotide microarrays for global transcription profiling. *BMC genomics*, 11(1):282.

Brock, G., Pihur, V., Datta, S., and Datta, S. (2008). Clvalid: An r package for cluster validation. *Journal of Statistical Software*, 25(4):1–22.

Brown, T. (2002). *Genomes*. BIOS Scientific Publishers Ltd., Oxford.

Bryan, K. (2005). Biclustering of expression data using simulated annealing. In Tsymbal, A. and Cunningham, P., editors, *CBMS '05: Proceedings of the 18th IEEE Symposium on Computer-Based Medical Systems*, pages 383–388, Washington, DC, USA. IEEE Computer Society.

Buck, M. J. and Lieb, J. D. (2004). Chip-chip: considerations for the design, analysis, and application of genome-wide chromatin immunoprecipitation experiments. *Genomics*, 83(3):349 – 360.

Bullard, J., Purdom, E., Hansen, K., and Dudoit, S. (2010). Evaluation of statistical methods for normalization and differential expression in mrna-seq experiments. *BMC Bioinformatics*, 11(1):94.

Cabanski, C. R., Qi, Y., Yin, X., Bair, E., Hayward, M. C., Fan, C., Li, J., Wilkerson, M. D., Marron, J. S., Perou, C. M., and Hayes, D. N. (2010). Swiss made: Standardized within class sum of squares to evaluate methodologies and dataset elements. *PLoS ONE*, 5(3):e9905.

Chakraborty, A. and Maka, H. (2005). Biclustering of gene expression data using genetic

algorithm. In *Computational Intelligence in Bioinformatics and Computational Biology, 2005. CIBCB '05. Proceedings of the 2005 IEEE Symposium on*, pages 1 – 8.

Chambers, L. D. (2000). *The Practical Handbook of Genetic Algorithms: Applications, Second Edition*. CRC Press, Inc., Boca Raton, FL, USA.

Chen, K. C., Csikasz-Nagy, A., Gyorffy, B., Val, J., Novak, B., and Tyson, J. J. (2000). Kinetic Analysis of a Molecular Model of the Budding Yeast Cell Cycle. *Molecular Biology of the Cell*, 11(1):369–391.

Cheng, C., Shen, K., Song, C., Luo, J., and Tseng, G. C. (2009). Ratio adjustment and calibration scheme for gene-wise normalization to enhance microarray inter-study prediction. *Bioinformatics*, 25(13):1655–1661.

Cheng, Y. and Church, G. M. (2000). Biclustering of expression data. In Altman, R., Bailey, T. L., Bourne, P., Gribskov, M., Lengauer, T., Shindyalov, I. N., Eyckand, L. F. T., and Weissig, H., editors, *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103. AAAI Press.

Cho, S.-B. and Won, H.-H. (2003). Machine learning in dna microarray analysis for cancer classification. In *APBC '03: Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics 2003*, pages 189–198, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Collas, P. (2010). The current state of chromatin immunoprecipitation. *Molecular biotechnology*, 45(1):87–100.

Cybenko, G. (1989). Approximation by superposition of sigmoidal functions. *Mathematics of Control, Signals and Systems*, 2:303–314.

Daisuke, T. and Horton, P. (2006). Inference of scale-free networks from gene expression time series. *Journal of Bioinformatics and Computational Biology*, 4:503–514.

Datta, S. and Datta, S. (2006). Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes. *BMC Bioinformatics*, 7(1):397.

Davies, D. L. and Bouldin, D. W. (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182 –197.

Deng, X., Geng, H., and Ali, H. (2005). Examine: a computational approach to reconstructing gene regulatory networks. *Biosystems*, 81:125–136.

Di Gesu, V., Giancarlo, R., Lo Bosco, G., Raimondi, A., and Scaturro, D. (2005). Genclust: A genetic algorithm for clustering gene expression data. *BMC Bioinformatics*, 6(1):289.

Do, J. H. and Choi, D.-K. (2006). Normalization of Microarray Data: Single-labeled and Dual-labeled Arrays. *Molecules and Cells*, 22(3):254–261.

Donoho, D., Maleki, A., and Shahram, M. (1995). Wavelab and reproducible research. *http://www-stat.stanford.edu/˜wavelab/Wavelab_850/index_wavelab850.html*. Date accessed: Feb. 2010.

Eick, C. F., Zeidat, N., and Zhao, Z. (2004). Supervised clustering algorithms and benefits. In *In proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI04) , Boca*, pages 774–776.

Elgar, S. J., Han, J., and Taylor, M. V. (2008). Mef2 activity levels differentially affect gene expression during drosophila muscle development. *Proceedings of the National Academy of Sciences of the United States of America*, 105(3):918–923.

Eriksson, R. and Olsson, B. (2004). Adapting genetic regulatory models by genetic programming. *Biosystems*, 76(1-3):217 – 227.

Esmaeili, A. and Jacob, C. (2009). A multi-objective differential evolutionary approach toward more stable gene regulatory networks. *Biosystems*, 98(3):127 – 136.

Estrada, B., Choe, S. E., Gisselbrecht, S. S., Michaud, S., Raj, L., Busser, B. W., Halfon, M. S., Church, G. M., and Michelson, A. M. (2006). An integrated strategy for analyzing the unique developmental programs of different myoblast subtypes. *PLoS Genetics*, 2(2):e16.

Ferrazzi, F., Magni, P., Sacchi, L., Nuzzo, A., Petrovic, U., and Bellazzi, R. (2007). Inferring gene regulatory networks by integrating static and dynamic data. *International Journal of Medical Informatics*, 76(Supplement 3):S462 – S475.

Fomekong-Nanfack, Y., Kaandorp, J. A., and Blom, J. (2007). Efficient parameter estimation for spatio-temporal models of pattern formation: case study of *Drosophila melanogaster*. *Bioinformatics*, 23(24):3356–3363.

Fomekong-Nanfack, Y., Postma, M., and Kaandorp, J. (2009). Inferring *Drosophila* gap gene regulatory network: a parameter sensitivity and perturbation analysis. *BMC Systems Biology*, 3(1):94.

Fox, R. M., Hanlon, C. D., and Andrew, D. J. (2010). The CrebA/Creb3-like transcription factors are major and direct regulators of secretory capacity. *The Journal of Cell Biology*, 191(3):479–492.

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601–620.

Friedman, N., Murphy, K., and Russell, S. (1999). Learning the structure of dynamic probabilistic networks. In Laskey, K. B. and Prade, H., editors, *15th Annual Conference on Uncertainty in Artificial Intelligence*, pages 139–147, San Francisco. Morgan Kaufmann.

Fu, X., Fu, N., Guo, S., Yan, Z., Xu, Y., Hu, H., Menzel, C., Chen, W., Li, Y., Zeng, R., et al. (2009). Estimating accuracy of rna-seq and microarrays with proteomics. *BMC genomics*, 10(1):161.

Gallo, S. M., Gerrard, D. T., Miner, D., Simich, M., Des Soye, B., Bergman, C. M., and

Halfon, M. S. (2010). Redfly v3.0: toward a comprehensive database of transcriptional regulatory elements in *Drosophila*. *Nucleic Acids Research*, 39:D118–D123.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Professional.

Hampshire, A. J., Rusling, D. A., Victoria, and Fox, K. R. (2007). Footprinting: A method for determining the sequence selectivity, affinity and kinetics of DNA-binding ligands. *Methods*, 42(2):128–140.

He, F., Balling, R., and Zeng, A.-P. (2009). Reverse engineering and verification of gene networks: Principles, assumptions, and limitations of present methods and future perspectives. *Journal of Biotechnology*, 144(3):190 – 203.

Heath, A. and Kavraki, L. (2009). Computational challenges in systems biology. *Computer Science Review*, 3(1):1–17.

Hecker, M., Lambeck, S., Toepfer, S., van Someren, E., and Guthke, R. (2009). Gene regulatory network inference: Data integration in dynamic models–a review. *Biosystems*, 96(1):86 – 103.

Hurd, P. J. and Nelson, C. J. (2009). Advantages of next-generation sequencing versus the microarray in epigenetic research. *Briefings in Functional Genomics & Proteomics*, 8(3):174–183.

Huttenhower, C., Mutungu, K. T., Indik, N., Yang, W., Schroeder, M., Forman, J. J., Troyanskaya, O. G., and Coller, H. A. (2009). Detailing regulatory networks through large scale data integration. *Bioinformatics*, 25(24):3267–3274.

Iba, H. (2008). Inference of differential equation models by genetic programming. *Information Sciences*, 178(23):4453–4468.

Iba, H. and Mimura, A. (2002). Inference of a gene regulatory network by means of interactive evolutionary computing. *Information Sciences*, 145(3-4):225–236.

Imade, H., Mizuguchi, N., Ono, I., Ono, N., and Okamoto, M. (2004). "Gridifying" an Evolutionary Algorithm for Inference of Genetic Networks Using the Improved GOGA Framework and Its Performance Evaluation on OBI Grid. In *Grid Computing in Life Science - LSGRID*, volume 3370, pages 171–186.

Imade, H., Morishita, R., Ono, I., Ono, N., and Okamoto, M. (2003). A grid-oriented genetic algorithm for estimating genetic networks by S-Systems. *SICE 2003 Annual Conference*, 3:2750–2755.

Jin, L. and Lloyd, R. V. (1997). In situ hybridization: Methods and applications. *Journal of Clinical Laboratory Analysis*, 11(1):2–9.

Johnson, W. E., Li, C., and Rabinovic, A. (2007). Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8(1):118–127.

Jothi, R., Cuddapah, S., Barski, A., Cui, K., and Zhao, K. (2008). Genome-wide identification of in vivo proteindna binding sites from chip-seq data. *Nucleic Acids Research*, 36(16):5221–5231.

Kabir, M., Noman, N., and Iba, H. (2010). Reverse engineering gene regulatory network from microarray data using linear time-variant model. *BMC Bioinformatics*, 11(Suppl 1):S56.

Kaiser, G. (1994). *A Friendly Guide to Wavelets*. Birkhauser, Boston, MA.

Kaiser, S. and Leisch, F. (2008). A Toolbox for Bicluster Analysis in R.

Kapranov, P., Cawley, S. E., Drenkow, J., Bekiranov, S., Strausberg, R. L., Fodor, S. P. A., and Gingeras, T. R. (2002). Large-scale transcriptional activity in chromosomes 21 and 22. *Science*, 296(5569):916–919.

Keedwell, E. and Narayanan, A. (2005). Discovering gene networks with a neural-genetic hybrid. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 2(3):231–242.

Kerr, G. (2009). Computational analysis of gene expression data. PhD Thesis.

Kerr, G., Ruskin, H. J., Crane, M., and Doolan, P. (2008). Techniques for clustering gene expression data. *Computers in Biology and Medicine*, 38(3):283–293.

Kikuchi, S., Tominaga, D., Arita, M., Takahashi, K., and Tomita, M. (2003). Dynamic modeling of genetic networks using genetic algorithm and S-System. *Bioinformatics*, 19(5):643–650.

Kim, S. Y., Imoto, S., and Miyano, S. (2003). Inferring gene networks from time series microarray data using dynamic bayesian networks. *Briefings in Bioinformatics*, 4(3):228–235.

Kimura, S., Hatakeyama, M., and Konagaya, A. (2003). Inference of S-system models of genetic networks using a genetic local search. *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*, 1:631–638 Vol.1.

Kirkilionis, M., Janus, U., and Sbano, L. (2011a). Multi-scale genetic dynamic modelling i : an algorithm to compute generators. *Theory in Biosciences*, pages 1–18.

Kirkilionis, M., Janus, U., and Sbano, L. (2011b). Multi-scale genetic dynamic modelling ii: application to synthetic biology. *Theory in Biosciences*, pages 1–19.

Kita, E., editor (2011). *Evolutionary Algorithms*. InTech, Rijeka, Croatia.

Koduru, P., Das, S., Welch, S., and Roe, J. L. (2004). Fuzzy dominance based multi-objective GA-Simplex hybrid algorithms applied to gene network models. In Deb, K., Poli, R., Banzhaf, W., Beyer, H.-G., Burke, E., Darwen, P., Dasgupta, D., Floreano, D., Foster, J., Harman, M., Holland, O., Lanzi, P., Spector, L., Tettamanzi, A., Thierens, D., and Tyrrell, A., editors, *Genetic and Evolutionary Computation - GECCO 2004*, pages 356–367.

Koduru, P., Das, S., Welch, S., Roe, J. L., and Lopez-Dee, Z. P. (2005). A co-evolutionary hybrid algorithm for multi-objective optimization of gene regulatory network models. In

GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation, pages 393–399, New York, NY, USA. ACM.

Koduru, P., Das, S., and Welch, S. M. (2007). Multi-objective hybrid pso using μ-fuzzy dominance. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 853–860, New York, NY, USA. ACM.

Koduru, P., Dong, Z., Das, S., Welch, S., Roe, J. L., and Charbit, E. (2008). A multiobjective evolutionary-simplex hybrid approach for the optimization of differential equation models of gene networks. *IEEE Transactions on Evolutionary Computation*, 12(5):572–590.

Kundaje, A., Xin, X., Lan, C., Lianoglou, S., Zhou, M., Zhang, L., and Leslie, C. (2008). A predictive model of the oxygen and heme regulatory network in yeast. *PLoS Computational Biology*, 4(11):e1000224.

Lee, W.-P. and Tzou, W.-S. (2009). Computational methods for discovering gene networks from expression data. *Briefings in Bioinformatics*, 10(4):408–423.

Lee, W.-P. and Yang, K.-C. (2008). A clustering-based approach for inferring recurrent neural networks as gene regulatory networks. *Neurocomputation*, 71(4-6):600–610.

Leisch, F. (2006). A toolbox for k-centroids cluster analysis. *Computational Statistics and Data Analysis*, 51(2):526–544.

Li, C. and Wong, W. H. (2001). Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection. *Proceedings of the National Academy of Sciences of the United States of America*, 98(1):31–36.

Li, L. (2001). Gene Assessment and Sample Classification for Gene Expression Data Using a Genetic Algorithm/K-Nearest Neighbor Method. *Combinatorial Chemistry and High Throughput Screening*, 4:727.

Li, L., Umbach, D. M., Terry, P., and Taylor, J. A. (2004). Application of the GA/KNN method to SELDI proteomics data. *Bioinformatics*, 20(10):1638–1640.

Liang, S., Fuhrman, S., and Somogyi, R. (1998). Reveal: a general reverse engineering algorithm for inference of genetic network architectures. In *Proceeding of the Pacific Symposium on Biocomputing*, pages 18–29.

Lim, W. K., Wang, K., Lefebvre, C., and Califano, A. (2007). Comparative analysis of microarray normalization procedures: effects on reverse engineering gene networks. *Bioinformatics*, 23(13):i282–288.

Linden, R. and Bhaya, A. (2007). Evolving fuzzy rules to model gene expression. *Biosystems*, 88(1-2):76 – 91.

Liu, J., Ghanim, M., Xue, L., Brown, C. D., Iossifov, I., Angeletti, C., Hua, S., Negre, N., Ludwig, M., Stricker, T., Al-Ahmadie, H. A., Tretiakova, M., Camp, R. L., Perera-Alberto, M., Rimm, D. L., Xu, T., Rzhetsky, A., and White, K. P. (2009a). Analysis of *Drosophila* Segmentation Network Identifies a JNK Pathway Factor Overexpressed in Kidney Cancer. *Science*, 323(5918):1218–1222.

Liu, K.-H., Zhang, J., Li, B., and Du, J.-X. (2009b). A GA-Based Approach to ICA Feature Selection: An Efficient Method to Classify Microarray Datasets. In Yu, W., He, H., and Zhang, N., editors, *ISNN 2009: Proceedings of the 6th International Symposium on Neural Networks*, pages 432–441, Berlin, Heidelberg. Springer-Verlag.

Liu, S., Lin, L., Jiang, P., Wang, D., and Xing, Y. (2011). A comparison of rna-seq and high-density exon array for detecting differential gene expression between closely related species. *Nucleic Acids Research*, 39(2):578.

Liu, X., Fu, J., Gu, D., Liu, W., Liu, T., Peng, Y., Wang, J., and Wang, G. (2008). Genome-wide analysis of gene expression profiles during the kernel development of maize (*Zea mays L.*). *Genomics*, 91(4):378 – 387.

Lockhart, D. J., Dong, H., Byrne, M. C., Follettie, M. T., Gallo, M. V., Chee, M. S., Mittmann, M., Wang, C., Kobayashi, M., Horton, H., and Brown, E. L. (1997). Expres-

sion monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 15:1359–1367.

Logan, J., Edwards, K., and Saunders, N., editors (2009). *Real-Time PCR: Current Technology and Applications*. Caister Academic Press, Norfolk, UK.

Lu, Y., Lu, S., Fotouhi, F., Deng, Y., and Brown, S. (2004a). FGKA: A Fast Genetic K-means Algorithm. In *Proceedings of the 2004 ACM Symposium on Applied Computing (SAC)*, New York, USA. ACM.

Lu, Y., Lu, S., Fotouhi, F., Deng, Y., and Brown, S. (2004b). Incremental genetic K-Means algorithm and its application in gene expression data analysis. *BMC Bioinformatics*, 5(1):172.

Maki, Y., Tominaga, D., Okamoto, M., Watanabe, S., and Eguchi, Y. (2001). Development of a system for the inference of large scale genetic networks. *Pacific Symposium on Biocomputing*, 1:446–458.

Manning, C. D. and Schtze, H. (1999). *Foundations of Statistical Natural Language Processing*, chapter 14. MIT Press, Cambridge, MA, USA.

Marbach, D., Prill, R. J., Schaffter, T., Mattiussi, C., Floreano, D., and Stolovitzky, G. (2010). Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences of the United States of America*, 107(14):6286–6291.

Marbach, D., Schaffter, T., Mattiussi, C., and Floreano, D. (2009). Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of Computational Biology*, 16(2):229–239.

Marioni, J., Mason, C., Mane, S., Stephens, M., and Gilad, Y. (2008). Rna-seq: an assessment of technical reproducibility and comparison with gene expression arrays. *Genome research*, 18(9):1509.

McGill, R., Tukey, J. W., and Larsen, W. A. (1978). Variations of box plots. *The American Statistician*, 32(1):12–16.

Mitchell, M. (1999). *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA.

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill, Columbus, OH.

Mitra, S. and Banka, H. (2006). Multi-Objective Evolutionary Biclustering of gene expression data. *Pattern Recognition*, 39(12):2464 – 2477.

modENCODE Consortium, T., Roy, S., Ernst, J., Kharchenko, P. V., Kheradpour, P., Negre, N., Eaton, M. L., Landolin, J. M., Bristow, C. A., Ma, L., Lin, M. F., Washietl, S., Arshinoff, B. I., Ay, F., Meyer, P. E., Robine, N., Washington, N. L., Di Stefano, L., Berezikov, E., Brown, C. D., Candeias, R., Carlson, J. W., Carr, A., Jungreis, I., Marbach, D., Sealfon, R., Tolstorukov, M. Y., Will, S., Alekseyenko, A. A., Artieri, C., Booth, B. W., Brooks, A. N., Dai, Q., Davis, C. A., Duff, M. O., Feng, X., Gorchakov, A. A., Gu, T., Henikoff, J. G., Kapranov, P., Li, R., MacAlpine, H. K., Malone, J., Minoda, A., Nordman, J., Okamura, K., Perry, M., Powell, S. K., Riddle, N. C., Sakai, A., Samsonova, A., Sandler, J. E., Schwartz, Y. B., Sher, N., Spokony, R., Sturgill, D., van Baren, M., Wan, K. H., Yang, L., Yu, C., Feingold, E., Good, P., Guyer, M., Lowdon, R., Ahmad, K., Andrews, J., Berger, B., Brenner, S. E., Brent, M. R., Cherbas, L., Elgin, S. C. R., Gingeras, T. R., Grossman, R., Hoskins, R. A., Kaufman, T. C., Kent, W., Kuroda, M. I., Orr-Weaver, T., Perrimon, N., Pirrotta, V., Posakony, J. W., Ren, B., Russell, S., Cherbas, P., Graveley, B. R., Lewis, S., Micklem, G., Oliver, B., Park, P. J., Celniker, S. E., Henikoff, S., Karpen, G. H., Lai, E. C., MacAlpine, D. M., Stein, L. D., White, K. P., and Kellis, M. (2010). Identification of functional elements and regulatory circuits by drosophila modencode. *Science*.

Montgomery, S., Sammeth, M., Gutierrez-Arcelus, M., Lach, R., Ingle, C., Nisbett, J., Guigo, R., and Dermitzakis, E. (2010). Transcriptome genetics using second generation sequencing in a caucasian population. *Nature*, 464(7289):773–777.

Morishita, R., Imade, H., Ono, I., Ono, N., and Okamoto, M. (2003). Finding multiple solutions based on an evolutionary algorithm for inference of genetic networks by S-system. *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*, 1:615–622 Vol.1.

Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L., and Wold, B. (2008). Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature Methods*, 5(7):621–628.

Murali, T., Pacifico, S., Yu, J., Guest, S., Roberts, G. G., and Finley, R. L. (2011). DroID 2011: a comprehensive, integrated resource for protein, transcription factor, RNA and gene interactions for *Drosophila*. *Nucleic Acids Research*, 39(suppl 1):D736–D743.

Nacher, J. and Akutsu, T. (2006). Sensitivity of the power-law exponent in gene expression distribution to mRNA decay rate. *Physics Letters A*, 360(1):174 – 178.

NCBI (2010). NCBI Sequence Read Archive, Accession number SRP001065. Date accessed: July 2010.

Noman, N. and Iba, H. (2005). Inference of Gene Regulatory Networks using S-System and Differential Evolution. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 439–446, New York, NY, USA. ACM.

Noman, N. and Iba, H. (2006). Inference of genetic networks using S-System: information criteria for model selection. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 263–270, New York, NY, USA. ACM.

Noman, N. and Iba, H. (2007). Inferring gene regulatory networks using differential evolution with local search heuristics. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(4):634–647.

Ono, I., Seike, Y., Morishita, R., Ono, N., Nakatsui, M., and Okamoto, M. (2004). An evolutionary algorithm taking account of mutual interactions among substances for in-

ference of genetic networks. *Evolutionary Computation, 2004. CEC2004. Congress on*, 2:2060–2067.

Ooi, C. H. and Tan, P. (2003). Genetic Algorithms applied to multi-class prediction for the analysis of gene expression data. *Bioinformatics*, 19(1):37–44.

Orlando, D. A., Lin, C. Y., Bernard, A., Wang, J. Y., Socolar, J. E., Iversen, E. S., Hartemink, A. J., and Haase, S. B. (2008). Global control of cell-cycle transcription by coupled CDK and network oscillators. *Nature*, 453(7197):944–947.

Oshima, Y., Ogawa, N., and Harashima, S. (1996). Regulation of phosphatase synthesis in *Saccharomyces cerevisiae* – a review. *Gene*, 179(1):171 – 177.

Ozbudak, E., Thattai, M., Kurtser, I., Grossman, A., and van Oudenaarden, A. (2002). Regulation of noise in the expression of a single gene. *Nature genetics*, 31(1):69–73.

Pal, S., Bandyopadhyay, S., and Ray, S. (2006). Evolutionary computation in bioinformatics: a review. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 36(5):601–615.

Pickrell, J., Marioni, J., Pai, A., Degner, J., Engelhardt, B., Nkadori, E., Veyrieras, J., Stephens, M., Gilad, Y., and Pritchard, J. (2010). Understanding mechanisms underlying human gene expression variation with rna sequencing. *Nature*, 464(7289):768–772.

Pollard, D. (2011). *Drosophila* sequence specific transcription factor binding site matrices, http://www.danielpollard.com/matrices.html. Date accessed: March 2011.

Pramila, T., Miles, S., GuhaThakurta, D., Jemiolo, D., and Breeden, L. L. (2002). Conserved homeodomain proteins interact with MADS box protein Mcm1 to restrict ECB-dependent transcription to the M/G1 phase of the cell cycle. *Genes and Development*, 16(23):3034–3045.

Pramila, T., Wu, W., Miles, S., Noble, W. S., and Breeden, L. L. (2006). The Forkhead transcription factor HCM1 regulates chromosome segregation genes and fills the S-phase gap

in the transcriptional circuitry of the cell cycle. *Genes and Development*, 20(16):2266–2278.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, UK.

Prill, R. J., Marbach, D., Saez-Rodriguez, J., Sorger, P. K., Alexopoulos, L. G., Xue, X., Clarke, N. D., Altan-Bonnet, G., and Stolovitzky, G. (2010). Towards a Rigorous Assessment of Systems Biology Models: The DREAM3 Challenges. *Public Library of Science ONE*, 5(2):e9202.

Przytycka, T. M., Singh, M., and Slonim, D. K. (2010). Toward the dynamic interactome: it's about time. *Briefings in Bioinformatics*, 11(1):15–29.

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

Repsilber, D., Liljenstrm, H., and Andersson, S. G. E. (2002). Reverse engineering of regulatory networks: simulation studies on a genetic algorithm approach for ranking hypotheses. *Biosystems*, 66(1-2):31 – 41.

Robnik-Łikonja, M. and Kononenko, I. (2003). Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning*, 53:23– 69.

Rodrigo, G., Carrera, J., and Elena, S. F. (2010). Network design meets *in silico* evolutionary biology. *Biochimie*, 92(7):746 – 752.

Sakamoto, E. and Iba, H. (2001). Inferring a System of Differential Equations for a Gene Regulatory Network by using Genetic Programming. In *Proceedings of Congress on Evolutionary Computation*, pages 720–726.

Schena, M., Shalon, D., Davis, R. W., and Brown, P. O. (1995). Quantitative Monitoring of Gene Expression Patterns with a Complementary DNA Microarray. *Science*, 270(5235):467–470.

Schlitt, T. and Brazma, A. (2007). Current approaches to Gene Regulatory Network modelling. *BMC Bioinformatics*, 8(Suppl 6):S9.

Shabalin, A. A., Tjelmeland, H., Fan, C., Perou, C. M., and Nobel, A. B. (2008). Merging two gene-expression studies via cross-platform normalization. *Bioinformatics*, 24(9):1154–1160.

Shah, S. C. and Kusiak, A. (2004). Data mining and genetic algorithm based gene/SNP selection. *Artificial Intelligence in Medicine*, 31(3):183 – 196.

Shakya, K., Ruskin, H. J., Kerr, G., Crane, M., and Becker, J. (2009). Comparison of microarray pre-processing methods. In *Advances in Computational Biology, AEMB*. Springer. In press.

Shin, A. and Iba, H. (2003). Construction of genetic network using Evolutionary Algorithm and combined fitness function. *Genome Informatics*, 14:94–103.

Sîrbu, A., Ruskin, H. J., and Crane, M. (2010a). Comparison of evolutionary algorithms in gene regulatory network model inference. *BMC Bioinformatics*, 11(59).

Sîrbu, A., Ruskin, H. J., and Crane, M. (2010b). Cross-platform microarray data normalisation for regulatory network inference. *Public Library of Science ONE*, 5(11):e13822.

Sîrbu, A., Ruskin, H. J., and Crane, M. (2010c). Regulatory network modelling: Correlation for structure and parameter optimisation. In Karim, M., Lee, K., Ling, H., Maroudas, D., and Sobh, T., editors, *Proceedings of The IASTED Technology Conferences (International Conference on Computational Bioscience)*, Cambridge, Massachusetts.

Sîrbu, A., Ruskin, H. J., and Crane, M. (2011a). Integrating heterogeneous gene expression data for gene regulatory network modelling. *Theory in Biosciences - Special Issue ECCS10*. Accepted. DOI 10.1007/s12064-011-0133-0.

Sîrbu, A., Ruskin, H. J., and Crane, M. (2011b). *Stages of Gene Regulatory Network Inference: the Evolutionary Algorithm Role*, chapter 27, pages 521–545. InTech, Rijeka, Croatia.

Smyth, G. K. (2004). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical applications in genetics and molecular biology*, 3(1).

Smyth, G. K. and Speed, T. (2003). Normalization of cdna microarray data. *Methods*, 31(4):265 – 273. Candidate Genes from DNA Array Screens: application to neuroscience.

Souza, B. F. and Carvalho, A. P. (2005). Gene selection based on multi-class support vector machines and genetic algorithms. *Genetics and Molecular Research*, 4(3):599–607.

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., and Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Molecular Biology of the Cell*, 9(12):3273–3297.

Spieth, C., Streichert, F., Speer, N., and Zell, A. (2005a). Clustering-based approach to identify solutions for the inference of regulatory networks. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2005)*, volume 1, pages 660–667.

Spieth, C., Streichert, F., Speer, N., and Zell, A. (2005b). Multiobjective model optimization for inferring gene regulatory networks. In Coello Coello, C. A., Hernndez Aguirre, A., and Zitzler, E., editors, *Proceedings of the Conference on Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science*, volume 3410, pages 607–620.

Spieth, C., Streichert, F., Supper, J., Speer, N., and Zell, A. (2005c). Feedback memetic algorithms for modeling gene regulatory networks. *Proceedings of the 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, 2005. CIBCB '05.*, pages 1–7.

Spieth, C., Streichert, F., and Zell, N. S. A. (2004). Optimizing topology and parameters of gene regulatory network models from time-series experiments. In *Proceedings of*

the Genetic and Evolutionary Computation - GECCO 2004, Lecture Notes in Computer Science, volume 3102 (Part I), pages 461–470.

Spieth, C., Supper, J., Streichert, F., Speer, N., and Zell, A. (2006). JCell–a Java-based framework for inferring regulatory networks from time series data. *Bioinformatics*, 22(16):2051–2052.

SRA (2011). The NCBI Sequence Read Archive (sra, http://www.ncbi.nlm.nih.gov/Traces/sra).

Stekel, D. (2003). *Microarray Bioinformatics*. Cambridge University Press, Cambridge, UK.

Streichert, F. and Ulmer, H. (2005). JavaEvA - A Java Framework for Evolutionary Algorithms. Technical Report WSI-2005-06, Centre for Bioinformatics Tübingen, University of Tübingen.

Sun, Y., Todorovic, S., and Goodison, S. (2010). Local-learning-based feature selection for high-dimensional data analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1610 –1626.

Swain, M., Hunniford, T., Dubitzky, W., Mandel, J., and Palfreyman, N. (2005). Reverse-engineering gene-regulatory networks using evolutionary algorithms and grid computing. *Journal of Clinical Monitoring and Computing*, 19(4-5):329–37.

Tan, K., Tegner, J., and Ravasi, T. (2008). Integrated approaches to uncovering transcription regulatory networks in mammalian cells. *Genomics*, 91(3):219 – 231.

Thieffry, D. (1999). From global expression data to gene networks. *BioEssays*, 21:895–899.

Tian, T. and Burrage, K. (2003). Stochastic neural network models for gene regulatory networks. In *Proceedings of the 2003 Congress on Evolutionary Computation, 2003. CEC '03.*, volume 1, pages 162–169.

Toledano-Katchalski, H., Nir, R., Volohonsky, G., and Volk, T. (2007). Post-transcriptional repression of the drosophila midkine and pleiotrophin homolog miple by how is essential for correct mesoderm spreading. *Development*, 134(19):3473–3481.

Tomancak, P., Beaton, A., Weiszmann, R., Kwan, E., Shu, S., Lewis, S., Richards, S., Ashburner, M., Hartenstein, V., Celniker, S., and Rubin, G. (2002). Systematic determination of patterns of gene expression during *Drosophila* embryogenesis. *Genome Biology*, 3(12). Data can be downloaded from http://fruitfly.org/cgi-bin/ex/insitu.pl, date accessed: Oct 2010.

Tominaga, D., Okamoto, M., Maki, Y., Watanabe, S., and Eguchi, Y. (1999). Nonlinear numerical optimization technique based on a genetic algorithm for inverse problems: Towards the inference of genetic networks. In *GCB99 German Conference on Bioinformatics*, pages 101–111.

Vohradsky, J. (2001). Neural network model of gene expression. *The Journal of the Federation of American Societies for Experimental Biology* , 15:846–854.

Wahde, M. and Hertz, J. (2000). Coarse-grained reverse engineering of genetic regulatory networks. *Biosystems*, 55(1-3):129–136.

Wang, Y., Joshi, T., Zhang, X.-S., Xu, D., and Chen, L. (2006). Inferring gene regulatory networks from multiple microarray datasets. *Bioinformatics*, 22(19):2413–2420.

Wessa, P. (2008). Notched boxplots (v1.0.5) in free statistics software (v1.1.23-r4). http://www.wessa.net/rwasp_notchedbox1.wasp.

Wessels, L. F. A., Reinders, M. J. T., and Backer, E. (2001). Robust genetic network modeling by adding noisy data. In *IEEE - EURASIP Workshop on Nonlinear Signal and Image Processing*.

Xu, R., Hu, X., and Wunsch, D.C., I. (2004). Inference of genetic regulatory networks from time series gene expression data. In *Proceedings of the 2004 IEEE International Joint Conference on Neural Networks.*, volume 2, pages 1215–1220.

Xu, R., Venayagamoorthy, G. K., and Donald C. Wunsch, I. (2007). Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization. *Neural Networks*, 20(8):917–927.

Xulvi-Brunet, R. and Li, H. (2010). Co-expression networks: graph properties and topological comparisons. *Bioinformatics*, 26(2):205–214.

Yeung, M. K., Tegnér, J., and Collins, J. J. (2002). Reverse engineering gene networks using singular value decomposition and robust regression. *Proc Natl Acad Sci U S A*, 99(9):6163–6168.

Zhu, Z., Jia, S., and Ji, Z. (2010a). Towards a memetic feature selection paradigm. *Computational Intelligence Magazine*, 5(2):41–53.

Zhu, Z. and Ong, Y.-S. (2007). Memetic algorithms for feature selection on microarray data. In Liu, D., Fei, S., Hou, Z.-G., Zhang, H., and Sun, C., editors, *Advances in Neural Networks ISNN 2007*, volume 4491 of *Lecture Notes in Computer Science*, pages 1327–1335. Springer Berlin / Heidelberg.

Zhu, Z., Ong, Y.-S., and Dash, M. (2007). Markov blanket-embedded genetic algorithm for gene selection. *Pattern Recognition*, 40(11):3236–3248.

Zhu, Z., Ong, Y.-S., and Zurada, J. (2010b). Identification of full and partial class relevant genes. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 7(2):263 –277.

Part III

Appendices

Appendix A

In depth: comparison of evolutionary algorithms for quantitative model inference

Chapter 3 provided an outline of modelling approaches for gene regulatory networks and described the role of Evolutionary Computation in reverse engineering. In this Appendix, we concentrate on quantitative modelling of gene regulatory networks, as this is more informative than qualitative analysis of biological data. The aim is to analyse different Evolutionary Algorithms in detail, and to provide a comparison framework indicative of the advantages and disadvantages of each approach[1].

Previous (pair-wise) algorithm comparisons for the methods analysed here have been reported [Kikuchi et al., 2003; Noman and Iba, 2005]. However, to provide a valid comparison of existing EAs for continuous models, the algorithms should be applied not only to the same datasets, but also under the same framework. This work aims to achieve this and to provide a consistent evaluation of ideas reported in the literature. The models used are not evaluated here, but only the algorithms that build models from data.

[1]This comparison has been published in BMC Bioinformatics [Sîrbu et al., 2010a].

A.1 Methods

In order to analyse the performance of EAs for model parameter inference, we have implemented seven different approaches and compared them on several datasets. The approaches are (outlined later in this Section): CLGA (Classical GA), MOGA (Multi-Objective GA), GA+ES, GA+ANN , PEACE1 (Predictor by Evolutionary Algorithms and Canonical Equations 1), GLSDC (Genetic Local Search with distance independent Diversity Control) and DE+AIC. These methods use different continuous fine-grained models to represent the GRN and rely on EAs to find the model that best fits the experimental data. The algorithms were implemented using EvA2, a Java framework for EAs [Streichert and Ulmer, 2005] and the implementation and data sets used are available online [Sîrbu et al., 2010a].

The analysis consists of two stages: (i) five hybrid EAs (GA+ES, GA+ANN, PEACE1, GLSDC and DE+AIC) were assessed for scalability, robustness to noise and performance with real microarray data, and (ii) two classical EAs (CLGA and MOGA), were compared in a small-scale setting to evaluate the improvement introduced by the multi-objective approach.

Comparison of different EAs can be performed using several criteria. The most common are the fitness value of best individuals at the end of optimisation and the number of fitness evaluations required for obtaining an observed fitness value. Robustness of fitness values and solutions obtained over multiple runs can also be analysed. Additionally, a problem-dependent criterion was used: the obtained solutions are also compared to the initial models (in the case of synthetic data), or to previous biological knowledge (for real microarray data). Robustness to noise is assessed by maintaining a fixed number of fitness evaluations and other EA meta-parameters (e.g. mutation and crossover operators) and observing the decrease in fitness and solution quality with the addition of noise. Scalability analysis involves reverse engineering of GRNs of increasing size. The number of fitness evaluations was empirically chosen to allow the population to converge towards a stable fitness value (i.e. until only small improvements in fitness are observed). Table A.1 lists the criteria used for comparison of the algorithms implemented.

Table A.1: Evaluation criteria. This table defines criteria used for method evaluation. For detailed definitions, see Appendix C.

Criteria	**Description**
Goodness of data fit (*data MSE*)	Best/average MSE between real and simulated data over a number of runs. This measures the ability of the model to reproduce the experimental data
Identified interactions	Ability of algorithm to qualitatively identify interactions (Sensitivity/Specificity). An interaction is taken to be identified if the corresponding parameter has an absolute value larger than zero. Average values over multiple runs are used for comparison purposes.
Parameter quality (*parameter MSE*)	Best/average MSE between real parameters and algorithm solution over multiple runs. This measures the ability of the algorithm to find the exact parameters of a known model (important especially for underspecified systems.)
Robustness over multiple runs	Average variance of kinetic orders/rate constants over multiple runs
Robustness to noise	Performance of algorithm with noisy datasets: goodness of fit, identified interactions, parameter quality
Performance for real microarray data	Sensitivity/Specificity and goodness of fit when applied to real microarray experiments rather than to synthetic data
Scalability	Performance of algorithms with larger datasets, maximum dimensionality achieved, increase in running time and decrease in goodness of fit and identified parameter quality, (when moving from a smaller to a larger dataset)
Average running time	Over multiple runs.
Function calls	Average number of function calls required for the results obtained.

The rest of this section outlines the seven algorithms implemented.

A.1.1 CLGA

Introduced by [Tominaga et al., 1999], this algorithm optimises parameters for an S-System model (Chapter 3, Section 3.2) using a simple real coded GA. The $n(2n+2)$ parameters $(\alpha_i, \beta_i, g_{ij}, h_{ij})$ of the system are encoded in a $n \times (2n+2)$ matrix of real values, representing the individuals in the algorithm. Classical uniform crossover and mutation operators on vectors of real values are used. The algorithm minimises the difference between the microarray expression values and the expression values generated by the model.

$$fitness = \sum_{i=1}^{n} \sum_{t=1}^{T} \left(\frac{x_i(t) - y_i(t)}{x_i(t)} \right)^2 \tag{A.1}$$

where $x_i(t)$ is the expression value of gene i at time t, observed in real experiments, and $y_i(t)$ is the expression value of gene i at time t generated by the model. In order to evolve

sparse networks[2], the parameters falling below a pre-determined threshold are automatically set to zero in each generation. This was the first attempt to use EAs for this problem, and opened the way for many other approaches. Although its performance, i.e. finding parameters for 2 genes in a GRN, has been improved upon, this algorithm is retained in order to compare it to a multi-objective approach.

A.1.2 MOGA

Multi-objective (MO) optimisation is known to be more effective than combining the set of functions into a single one, as it forces all individual fitness values to be close to the optimum. This approach was applied to GRN S-System model inference by [Koduru et al., 2004], who, rather than adding errors (as in CLGA), used those corresponding to *each* gene expression series as a *different objective*. Furthermore, the concept of *fuzzy domination* was introduced, which gives a continuous aspect to individual domination in multi-objective optimisation. The Simplex algorithm was used to create part of the offspring for each generation. We have implemented this approach, apart from the Simplex algorithm. This was omitted in order to test whether the multi-objective approach *itself* improves the search performance in this problem setting and to determine how fuzzy dominance-based selection affects its performance, without hybridising the search method as well. NSGAII (Non-dominated Sorting Genetic Algorithm II [Deb et al., 2002]) tournament selection (implemented in EvA2) was used for MOGA and fuzzy dominance-based tournament selection for the Fuzzy MOGA. The MO algorithms use an archive of a maximum of 50 individuals belonging to the Pareto front that are fed to the next generation, to implement elitism.

A.1.3 GA+ES

One of the more advanced and recent approaches to applying EAs to GRN modelling divides the search into two phases: structure and parameter search. The S-System model is once again used. The approach is based on the sparsity of the networks: as most of the

[2]GRN are known to be sparsely connected, as each gene has a small number of regulators. This makes the matrix that encodes the solution sparse: many values are null and this fact is used in this approach.

values in the connection matrix are null, there is no need to search for all the parameters, but only for those found to be non-null. Consequently, one phase of the algorithm looks for the non-null parameters, and the other finds the real values for them. These two phases are nested into each other, the second one being similar to a local search performed on the individuals of the first. The first method of this kind we have implemented is the one introduced by [Spieth et al., 2004]. Here, a GA finds the connection matrix with elements in $\{0, 1\}$ and then an ES is employed to find the S-System parameters for the connections represented by the individuals of the GA. The fitness of the GA individuals is given by the best individual in the ES. The method was reported to have achieved parameter determination for a 20-gene artificial GRN.

A.1.4 GA+ANN

The second method implemented, which involves nesting the structure and the parameter search phases, is [Keedwell and Narayanan, 2005]. This is a neural-genetic hybrid that replaces the S-System in GA+ES with an ANN. Here, the structure search uses a different representation: a binary encoded set of real values, rather than the characteristic vector[3] used in [Spieth et al., 2004]. The ANN takes as input a subset of genes and learns the parameters for regulation through Backpropagation, (in order to find regulation strengths from each gene to the gene under analysis), then outputs the error of the resulting model as fitness for the GA individual. Figure A.1 shows the ANN topology used by this method. The fact that it is single layered allows for inference of causal relationship from the topology.

A.1.5 PEACE1

Initially, most methods that took into account the fact that gene networks are sparse, forced parameters that fell below a certain threshold to zero [Tominaga et al., 1999]. An additional way of generating sparse networks was introduced in [Kikuchi et al., 2003], extending CLGA. The fitness function was modified so that the algorithm tries to minimise the squared

[3] A characteristic vector encodes a subset of a finite universe, where, for each element in the universe, a boolean value is used that denotes whether that element is in the subset or not.

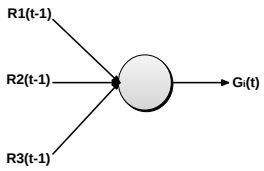


Figure A.1: ANN Topology in GA+ANN: represents how expression value of a gene G at time t is computed from the expression values of its regulators, R_1, R_2 and R_3 at time $t-1$ through a single layered ANN.

error of the model (first term in Equation A.2), while penalising the models with high connectivity (second term):

$$fitness = \sum_{i=1}^{n} \sum_{t=1}^{T} \left(\frac{x_i(t) - y_i(t)}{x_i(t)} \right)^2 + cnT \sum_{j=1}^{n} (|g_{ij}| + |h_{ij}|) \tag{A.2}$$

Here, c is a constant that balances the two terms in the fitness function. Simplex crossover is used and a two-stage gradual optimisation strategy is employed: initially, the GA is run several times to find multiple models that fit the data. Using the resulting models, a new instance of the GA is initialised and evolved towards a model that combines parameters from local solutions into a better global one. During all generations, the parameters that fall below the threshold are set to zero, in order to simplify the simulation process when evaluating an individual. This improved model is analysed and the null parameters are fixed at zero for the next iterations. This *double optimisation* (feeding one GA with the results of other GAs), is needed in order to avoid setting necessary parameters to zero. Afterwards, a new set of GA instances are employed to find local solutions, taking into account the fixed parameters. This procedure is iterated until no more parameters need to be suppressed or a maximum iteration number is reached. The method has been applied on the S-System model for GRNs.

A.1.6 GLSDC

GLSDC [Kimura et al., 2003] evolves parameters for one gene at a time, using the S-System model and a skeletalising term similar to [Kikuchi et al., 2003]:

$$fitness_i = -\sum_{t=1}^{T}\left(\frac{x_i(t) - y_i(t)}{x_i(t)}\right)^2 - c\sum_{j=1}^{n-I}(|G_{ij}| + |H_{ij}|) \tag{A.3}$$

Here, I is the maximum connectivity parameter and G_{ij} and H_{ij} are elements of two sets containing g_{ij} and h_{ij} in ascending order of their absolute values. The approach differs from that of [Kikuchi et al., 2003] in that only the chromosomes that contain more than I activators and I repressors are penalised by the second term of the fitness function and only the network connections with the lowest weights are included in the penalty term (as those with high weights are taken to be the correct interactions).

One of the most costly parts of the EA implementations presented previously is the *evaluation of individuals*, as it requires simulations of the model for each. GLSDC minimises this problem by splitting evolution into two stages: local search and convergence stage. During the convergence stage, crossover and mutation are applied to the population, similar to a classic GA, but no evaluation is performed, so that evolution is faster. During the local search phase, a set of restrictions, (inequalities involving model parameters), derived from data, are tested on the individuals in the current population. In case the conditions are not met for one individual, a local search, based on quadratic programming [Boot, 1964], is performed in order to find the closest parameters, (Euclidean distance), that do satisfy the inequalities. For the rest of the individuals, i.e. those conforming to the restrictions, Powell's local search [Press et al., 1992] is performed, to find a solution with a better fitness.

A.1.7 DE+AIC

DE+AIC [Noman and Iba, 2006, 2007] introduced a further improvement with respect to the penalty expression that forces the algorithm to evolve sparse models. Instead of ordering

g_{ij} and h_{ij} in sets G_{ij} and H_{ij}, as seen in GLSDC, these are combined in only one ordered set K_{ij}. The fitness function no longer measures the closeness of the data generated, by the model, to the experimental data, but uses an information theoretic criterion: Akaike's Information Criterion (AIC). This measures the likelihood of the data under a specific model and penalises those models which describe the data using a large number of parameters. The algorithm evolves parameters for one gene at a time. Considering all the modifications specified above, the fitness function, which the GA needs to minimise, becomes:

$$fitness_i = -2\Lambda + 2\phi_i + c \sum_{j=1}^{2n-I} |K_{ij}| \tag{A.4}$$

where Λ is the likelihood of the data under the model encoded by the current chromosome, ϕ_i is the number of parameters corresponding to gene i and I is the maximum connectivity.

The search heuristic in [Noman and Iba, 2006] is Trigonometric DE. For each generation, a local search is operated using Hill Climbing[4] [Chambers, 2000], to further skeletalise the models (obtain sparser realisations), encoded by the best individual and one selected randomly from the population. The procedure parses the set, K, of sorted kinetic orders,(g_{ij} and h_{ij}), from the smallest to the largest. At each step, it sets the current kinetic order to zero and, if the newly identified individual is better that the one before the change, it replaces the old one. This is repeated until all the kinetic orders have been changed. In a sense, this local search is similar to Powell's method [Press et al., 1992], as it searches each dimension at a time. This approach may miss some improvements that may happen only when two or more parameters are null *at the same time*, but, however, setting one parameter at a time to zero brings no fitness increase. Given that it is used as a local search technique in a more ample search, this is not a major drawback.

[4] An optimisation heuristic that starts with one solution and iteratively modifies it to obtain a better one. The modification is usually done with operators similar to GA mutation. The old solution is replaced by the new one only if the latter is better. The search stops either after a certain number of mutations or when the solution does not improve any more, within a specified tolerance.

A.2 Results and Discussion

In order to be able to evaluate our implementation on the chosen criteria (Table A.1), six datasets generated by S-System models of regulation and five for the ANN model were used. The models for two and five-gene S-System synthetic regulatory networks were taken from the literature [Tominaga et al., 1999], while those for larger systems (10, 20, 30, 50 genes, and for ANNs (5, 10, 20, 30, 50 genes) were randomly generated so that they conform to well known characteristics of real GRNs, i.e. scale-free sparse networks. Real GRNs are also known to display other characteristics such as modularity and feedback mechanisms [Marbach et al., 2009]. However, only sparsity is taken into account by the implemented methods, so using random sparse networks is a good indication of comparative algorithm performance. Nevertheless, we acknowledge that this could be a limitation with regard to relevance of the synthetic experiments in determining algorithm ability to reverse engineer the correct network from real data.

Robustness to noise was tested on the synthetic data for the five-gene networks to which 1%, 2%, 5% and 10% Gaussian noise was added to all values. The assumption of Gaussian noise has been used before in relation to gene expression data [Noman and Iba, 2007; Nacher and Akutsu, 2006] and, although it may not be true in all situations, it provides a good indication of the behaviour of the algorithm with real noisy data.

Ideally, in order to be able to build an S-System model, or to train an ANN, for a large scale network, a large number of measurements (time points) is required. This number increases further when data are noisy, [Mitchell, 1997]. However, in reality, due to the high cost of these experiments, only limited data are available. In order to simulate experiments with real data, we reduced the number of (synthetic) experimental time points used for inference to 60 for the 5-, 10- and 20-gene datasets, 80 for the 30-gene dataset and 125 for the 50-gene dataset.

As EAs are stochastic in nature, multiple runs were performed for each experiment. Multi-objective analysis was performed over 20 runs for each algorithm. The methods analysing the entire system were applied seven times on each dataset, while those using the

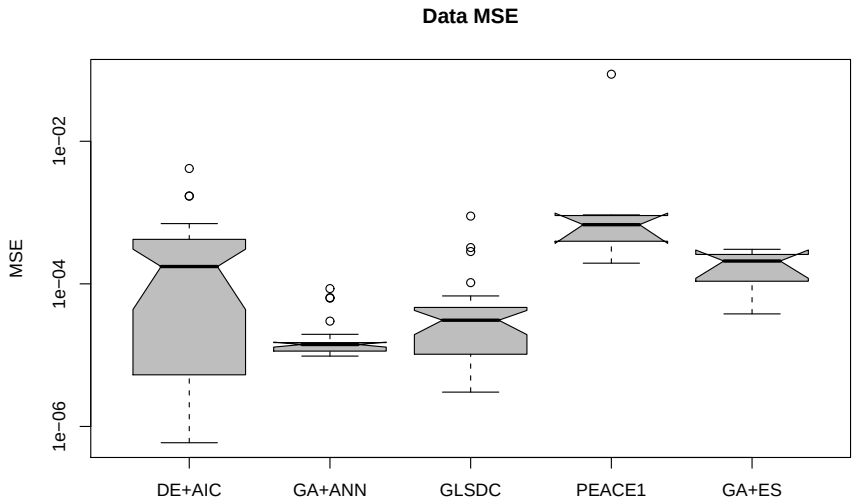


Figure A.2: Small-scale dataset - data fit. Box plot displaying data MSE values for each algorithm with the 5-gene dataset. GA+ANN exhibits significantly better data fit, while PEACE1 has the lowest performance.

Table A.2: Performance of algorithms over multiple runs using the 5-gene synthetic dataset, under three criteria: robustness over multiple runs, qualitative interactions and number of function calls performed.

Criteria	**PEACE1**	**GA+ ES**	**GA+ ANN**	**GLSDC**	**DE+ AIC**
Robustness (Kinetic orders /Rate constants variance)	0.25175/ 4.22818	0.4861/ 3.0170	0.07236	0.08449/ 2.0419	0.21534/ 6.41834
Identified interactions (Sensitivity/ Specificity)	0.55384/ 0.82702	0.6483/ 0.8902	0.74074/ 0.8125	0.72307/ 0.67837	0.58461/ 0.81081
Function calls	1650000	3750000	2500 $\times$ 20000 ANN epochs	100000	275000

divide et impera approach were run five times for each of the first five genes, resulting in 25 runs per dataset. The quantitative results for the algorithms are displayed using notched box plots, Appendix C.

A.2.1 Performance on small scale networks

For a first analysis, we applied five algorithms to the five-gene synthetic dataset from [Tominaga et al., 1999]. We chose this benchmark dataset due to the fact that it has been already used to validate most of the methods we are comparing. At the same time, the reduced dimensionality allows for easier analysis of EA parameters and for multiple runs to be

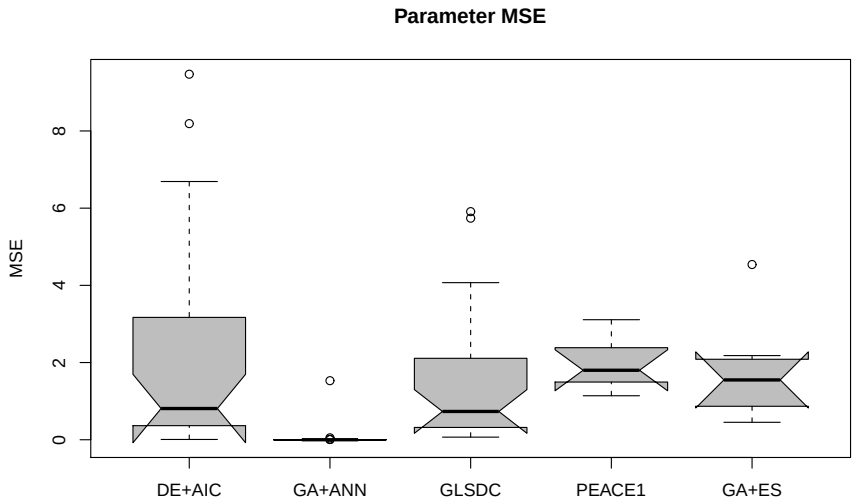


Figure A.3: Small-scale dataset - parameter quality. Box plot displaying parameter MSE values for each algorithm with the 5-gene dataset. GA+ANN finds better parameters, conforming with data MSE values.

performed. Figure A.2 displays the box plots representing the data fit obtained by each algorithm, while Figure A.3 presents the quality of parameters obtained over all runs performed. Table A.2 contains numerical values for three more evaluation criteria (robustness of parameters obtained, sensitivity and specificity and fitness calls). Note that PEACE1 and GA+ES analyse all genes simultaneously, while the others find interactions one gene at-a-time. However, the numerical values for all the genes in the latter type of methods are used, allowing for a direct comparison between them.

As Figure A.2 indicates, all five methods demonstrate good performance in fitting the data (based on *data* MSE as a goodness of fit criterion). However, GA+ANN displays best fitness, followed by GLSDC, while PEACE1 performs least. The fact that the notches around the median do not overlap provides evidence for these differences to be statistically significant at a 5% level. However, these are insufficient alone to choose a specific algorithm, as other options may exist and alternative model parameters may give a good fit to the data. Consequently, we provided (Figure A.3) the *parameter* MSE values that show how close the resulting model parameters are to the real one, (i.e. how much does each parameter deviate, on average, from the real model). These and the values in Table A.2 indicate that GA+ANN appears more robust and better able to identify correct interactions. How-

ever, it should be noted that this model has fewer parameters compared to the others, (25 as opposed to 60), hence reducing the solution space for the EA and, possibly, enhancing algorithm performance.

Although methods using the S-System model display similar average performance, (according to the parameter quality criterion), GA+ES and DE+AIC obtain the best parameters overall, (indicated by minimum values), while, (in sensitivity terms), GLSDC has a higher value, indicating that the latter is more suitable for a quantitative analysis than the two former, which, despite finding parameter values close to the real ones, can miss smaller values.

Table A.2 also supplies the number of function calls needed by the algorithms to achieve the performances above. These indicate the ANN approach to be faster; while each function call represents the training of an ANN, this is not very costly as these are small, due to the connectivity limit. PEACE1 requires a long running time, because of the numerous iterations needed to find all null parameters and, given the low specificity, seems to miss the low ones. GA+ES also requires a large number of function calls, due to the overhead of running a new instance of an ES for each structure evaluation.

A.2.2 Performance on noisy data

An important feature for inferential GRN algorithms, in a real biological setting, is robustness to noise. We have analysed the behaviour of the algorithms implemented on noisy datasets, and the results are displayed in Figures A.4 and A.5, which show the evolution with noise of data fit and parameter quality, using the same type of box plots for significance analysis. Figure A.6 shows average sensitivity and specificity values for the algorithms at the different noise levels.

The sensitivity and specificity criteria allow for a qualitative analysis of results. From the sensitivity point of view, the methods can be divided into three categories: with (1) stable sensitivity values (GLSDC, DE+AIC and GA+ANN), (2) decreasing sensitivity with noise (GA+ES), and (3) increasing sensitivity with noise (PEACE1). Specificity values, on the other hand, decline with noise for all methods, which is explained by the fact that the

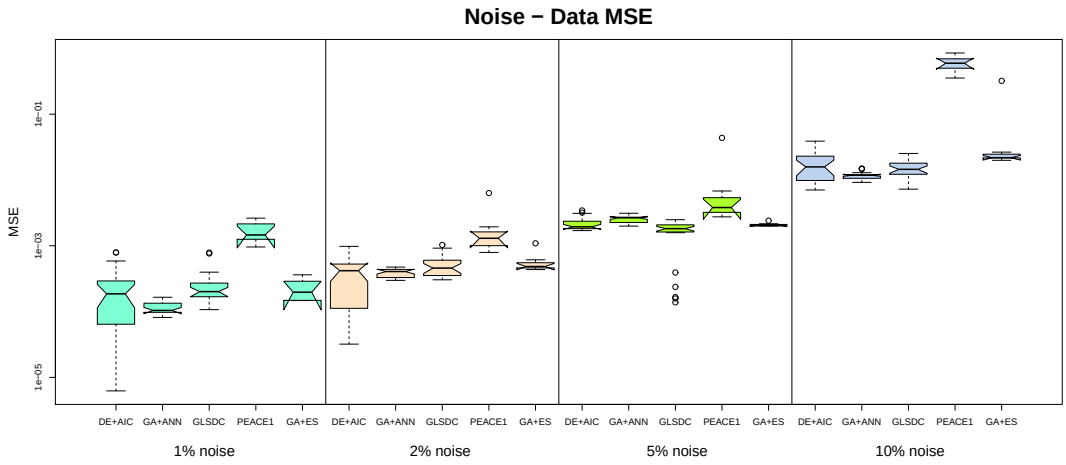


Figure A.4: Small-scale noisy datasets - data fit. Performance of the 5 algorithms with noisy datasets in terms of data fit (data MSE). Algorithms displayed are, from left to right: DE+AIC, GA+ANN, GLSDC, PEACE1, GA+ES. An increase of MSE values with noise can be observed. PEACE1 displays lowest performance, while the rest of the algorithms are comparable under this criterion.

algorithms concentrate on finding null interactions, so the number of true negatives discovered decreases with noise. However, the first two categories seem to exhibit significantly better behaviour than the third. This explains why PEACE1 achieves a maximum sensitivity with maximum noise: a very small proportion of parameters were found to be null, so almost all genes were found to interact. This results in a large number of true positives, however, accompanied by a very large number of *false positives*, which is undesirable.

The quantitative perspective has been analysed using the two criteria in Figures A.4 and A.5. For PEACE1, both data and parameter MSE are inferior to the rest, indicating limited ability to handle noise. However, only data MSE differences are statistically significant at all noise levels. The other four methods are stable and have comparable performance up to 5% noise (favourable behaviour for real microarray data). Concerning the 10% noisy dataset, two trends can be indentified: GLSDC and GA+ANN decrease the data fit but preserve a good parameter quality (parameter MSE), while for DE+AIC and GA+ES both data fit and parameter quality decrease significantly. This means that the former set have the ability to find good parameters in spite of noise, while the latter over-fit the noise in the data, implying low quality solutions. Good performance in the case of GA+ANN may

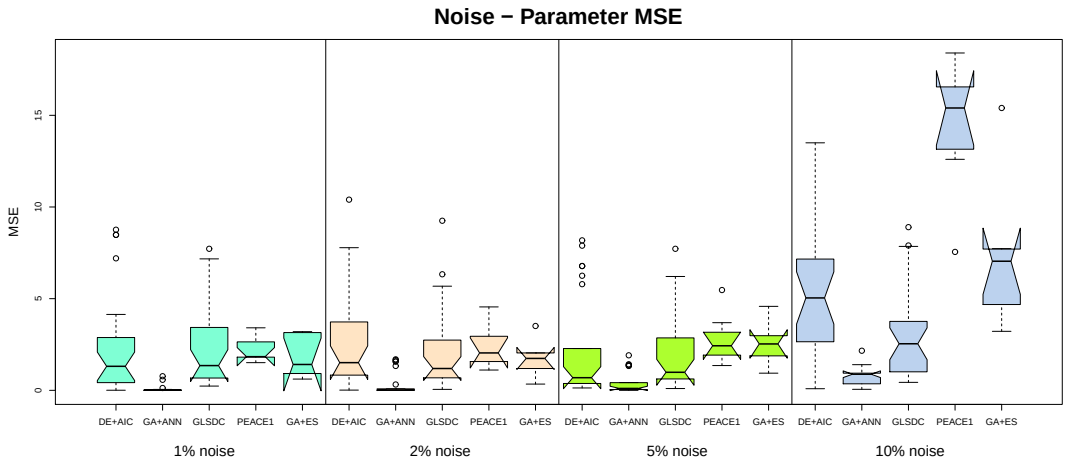


Figure A.5: Small-scale noisy datasets - parameter quality. Performance of the 5 algorithms with noisy datasets in terms of parameter fit (parameter MSE). Algorithms displayed are, from left to right: GA+ANN, GA+ES, DE+AIC, GLSDC, PEACE1. GA+ANN exhibits (statistically significant) better parameters, while the rest of the algorithms display similar behaviour. At high level of noise (10%), GLSDC also performs better compared to the rest.

be due to the nature of the ANN model, which has been proven to cope well with noise in multiple practical applications [Mitchell, 1997], while GLSDC has a mechanism built in the local search phase that specifically handles noise.

In conclusion, the ANN model and the GLSDC mechanism for controlling noise seem to give good quantitative results even with a high noise rate. The best balance for sensitivity-specificity is achieved with GA+ANN, while GA+ES, DE+AIC and GLSDC exhibit the best qualitative behaviour with noise under the S-System model (the former two find more null interactions, but miss some of the real ones and the latter finds most of the real ones but also adds some false positives).

A.2.3 Scalability

Scalability analysis was performed on four synthetic datasets corresponding to four different networks: 10, 20, 30 and 50 genes. For these data, quantitative results using box plots are displayed in Figures A.7 and A.8, while the best qualitative results of all runs are shown in Figure A.9. Given the small sensitivity on the 10 and 20 gene datasets (approximately 0.1), and the dimensionality achieved by the authors themselves, (five genes), no further

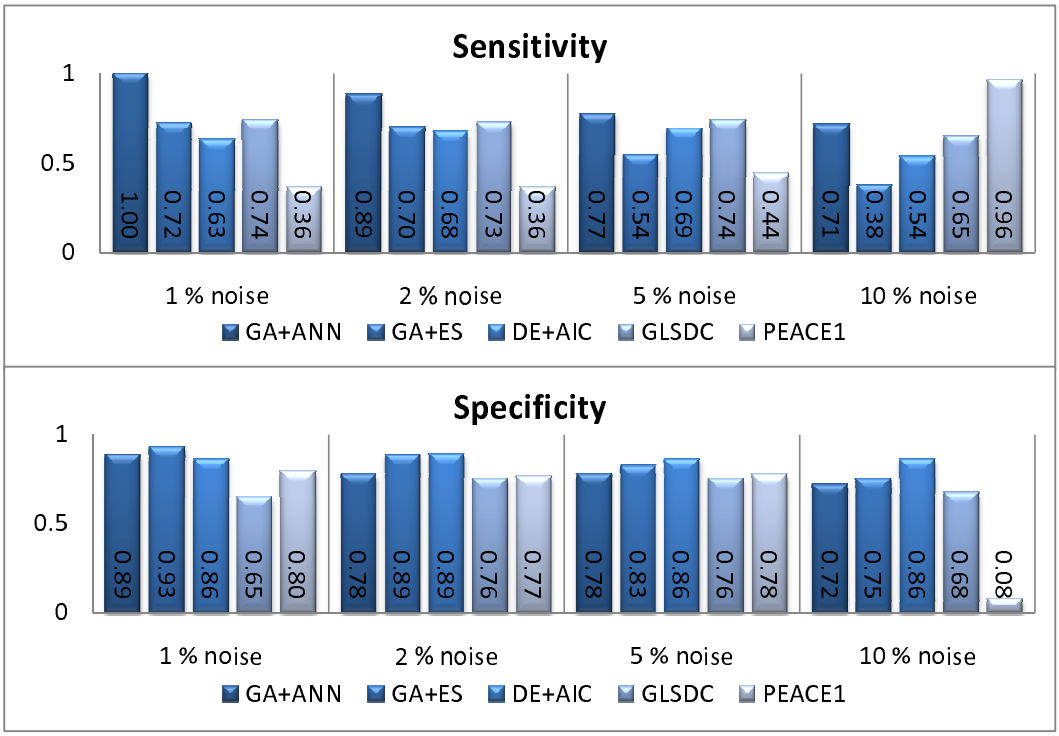


Figure A.6: Small-scale noisy datasets - identified parameters. Performance of the 5 algorithms with noisy datasets in terms sensitivity and specificity. Algorithms displayed are, from left to right: GA+ANN, GA+ES, DE+AIC, GLSDC, PEACE1. GLSDC, DE+AIC and GA+ANN display stable sensitivity values, GA+ES shows decreasing sensitivity with noise while for PEACE1 sensitivity increases with noise. Specificity values decrease with noise for all algorithms.

runs were performed with PEACE1 for the larger datasets. GA+ES was run on the 10-gene dataset with low performance (fitness 25 after 7,500,000 fitness calls, in 170 generations, during 47 hours), while on the 20-gene dataset, having doubled the allocated memory for the Java virtual machine, one generation lasted approximately 3 hours, and, after 35 generations ($\approx$ 109 hours), the best fitness value was 1.4E11. This indicates that this method does not scale very well in a single CPU setting, and was thus discarded from the analysis. For the three methods that analyse one gene at-a-time, we performed experiments on a limited number of genes, (5), and averaged criteria values on them. The results obtained in this way are indicative of the performance of the methods for all the genes in the network. The rest of this section concentrates on these three methods.

Due to the characteristically low connectivity of the networks, all methods analysed

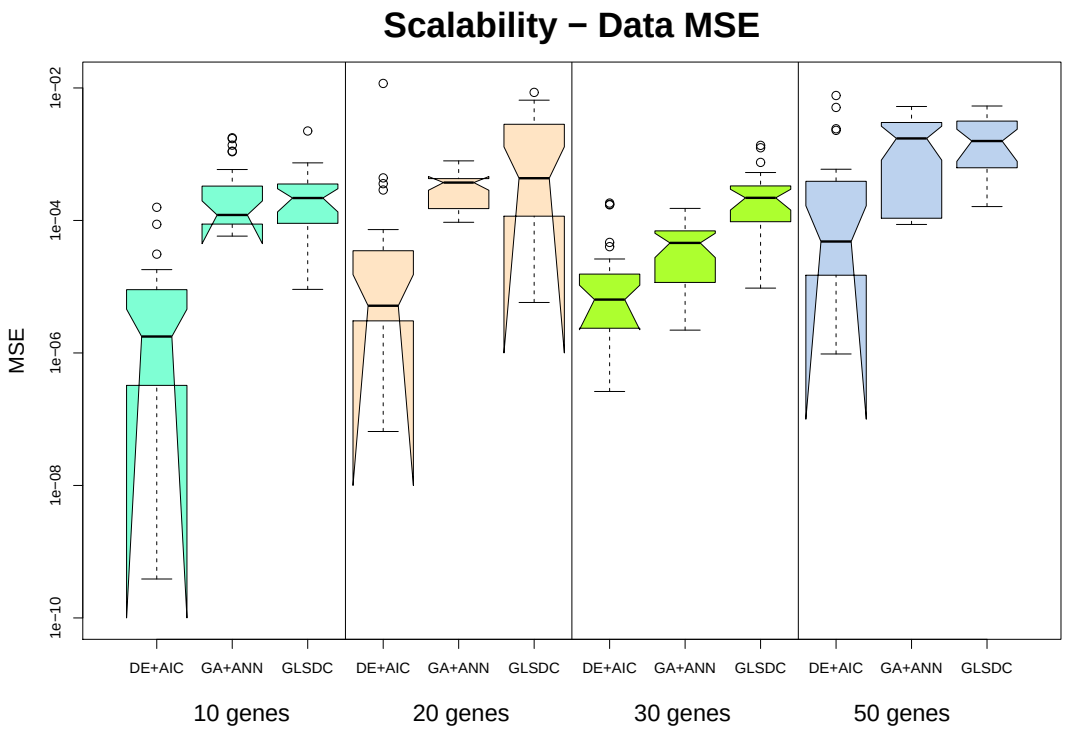


Figure A.7: Scalability - data fit. Box plot representing data MSE with larger datasets. Due to poor performance with the 10- and 20-gene datasets, the values for PEACE1 and GA+ES are not displayed. DE+AIC exhibits (statistically significant at 5% level) better behaviour compared to the rest.

displayed good specificity (preserved for all system scales). However, the sensitivity values tend to decrease with the increase in size, which indicates that, for larger networks, these methods tend to set increasing numbers of parameters to zero, so that more interactions are missed. However, the number of false positives remains small. GA+ANN maintains good qualitative performance up to 30 genes, while DE+AIC and GLSDC display good behaviour with the 10-gene dataset, but do less well as the size of the gene set increases. On the 50 gene dataset, all methods perform poorly in respect of the sensitivity values.

In order to analyse the quantitative behaviour of the methods implemented, values for two criteria were provided: ability to reproduce data (Figure A.7) and parameter quality (Figure A.8). Considering the fact that each benchmark dataset has a different number of parameters to be inferred, of which most are zero, the parameter MSE displayed in Figure A.8 is computed *per gene* rather than *per parameter*. Given the similar connectivity of the

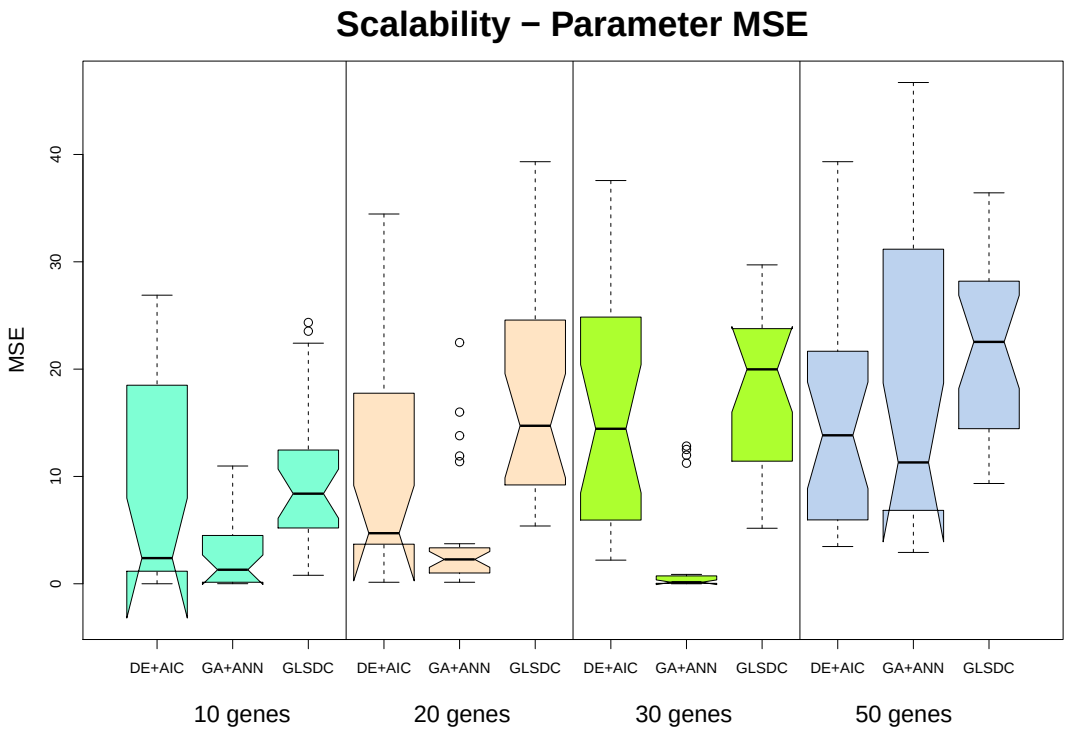


Figure A.8: Scalability - parameter quality. Parameter MSE with larger datasets (computed *per gene* rather than *per parameter*, see text). Due to poor performance with the 10- and 20-gene datasets, the values for PEACE1 and GA+ES are not displayed. Parameters identified by GA+ANN and DE+AIC are better than the rest up to 20 genes while for 30 genes only GA+ANN differs significantly.

four different networks (3 to 5), this offers a good measure of parameter quality that neither depends on the number of genes in the network, (which would have been the case if we had chosen the residual sum of squares), nor is biased by the large number of null parameters usually discovered by the algorithms.

As Figure A.7 indicates, all methods, except for those eliminated from this analysis after the first two experiments (PEACE1 and GA+ES), display good data fit for all datasets. However, DE+AIC exhibits significantly better data fit at all scales.

GA+ANN achieves good parameter quality (parameter MSE, Figure A.8), up to 30 genes, confirming conclusions from the qualitative measures. DE+AIC exhibits a behaviour comparable to GA+ANN up to 20 genes, but displays lower parameter quality for 30 genes, possibly due to the limited data. The superiority of the first method could be partly due to the smaller number of model parameters, compared to the other methods, the resulting

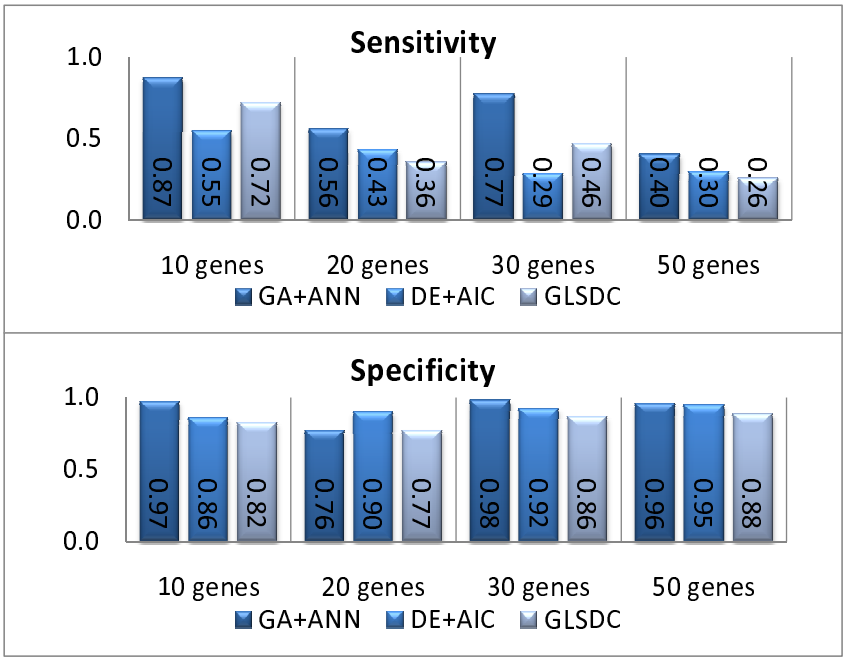


Figure A.9: Sensitivity and specificity for larger datasets. Due to poor performance with the 10- and 20-gene datasets, the values for PEACE1 and GA+ES are not displayed. Specificity values are stable with the increase in scale, but sensitivity values decrease with system size.

system being less markedly under-specified than in the case of S-Systems and the solution space being reduced.

In conclusion, the ANN model displays superior performance again with larger networks, while methods that analyse the whole system at the same time fail to scale up for a single CPU situation. The other two methods behave reasonably well up to 30 genes, identifying the most important interactions to enable them to closely simulate the synthetic time series.

A.2.4 Real DNA microarray data

In order to assess performance of the chosen algorithms on real microarray data, the Spellman dataset [Spellman et al., 1998] was used, which has become a benchmark for validating this type of method. This contains 18 time points measured during two *Saccharomyces cerevisiae* cell cycles. The known interactions between genes and proteins were retrieved from the KEGG database, [Aoki-Kinoshita and Kanehisa, 2007], for validation purposes.

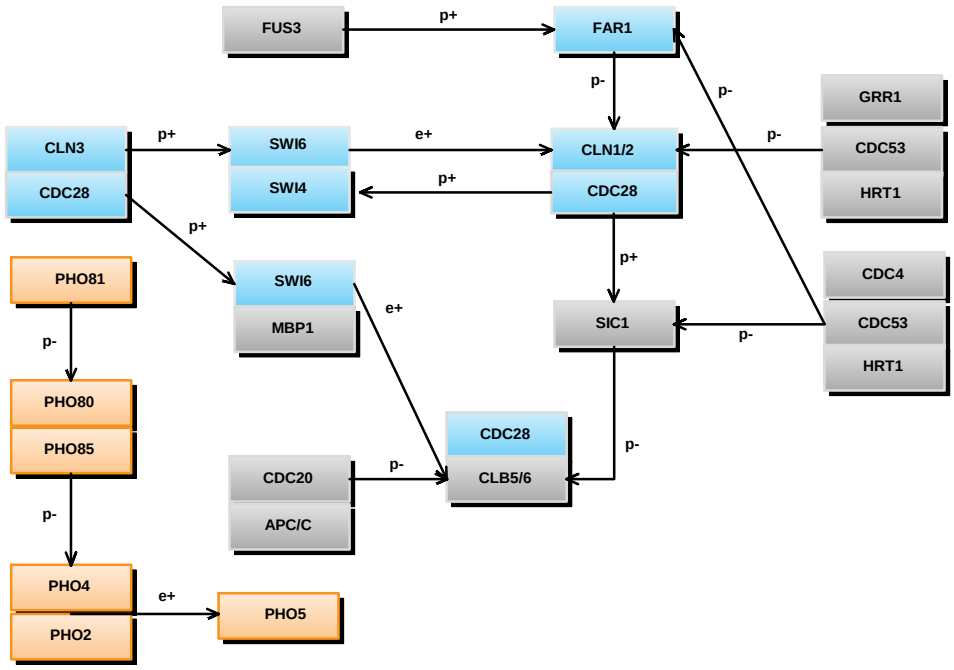


Figure A.10: Pathway for Yeast cell cycle, retrieved from KEGG database, for the 24 genes analysed. The two coloured sets of genes correspond to the two small sub-networks, (6 and 7 genes), analysed separately. The connections between genes labelled with *e* represent known gene regulatory interactions, while the ones labelled with *p* represent known interactions between proteins that can activate or repress the activity of one or several proteins involved.

Three subsystems of this network were analysed; two small-scale (6 and 7 genes) and one medium-scale network, (24 genes), of which the former were subsets (see Figure A.10). The two small-scale networks contain the genes known to be involved, respectively, in the regulation of genes *CLN2* and *PHO5*. The large-scale analysis focused on these two genes as well, in order to investigate how inclusion of additional genes, either not connected or distantly linked to the initial system, influences algorithm performance. The algorithms were applied five times for each gene under analysis.

Figure A.11 displays the ability of each algorithm to reproduce the time series for the two analysed genes (best results obtained in multiple runs), while Figure A.12 provides box plots for data MSE values. Even though, for the *PHO5* experiment, the difference in MSE values, compared to GLSDC and GA+ANN, is not statistically significant, (as extracted from Figure A.12), PEACE1 and GA+ES perform poorly in reproducing behaviour

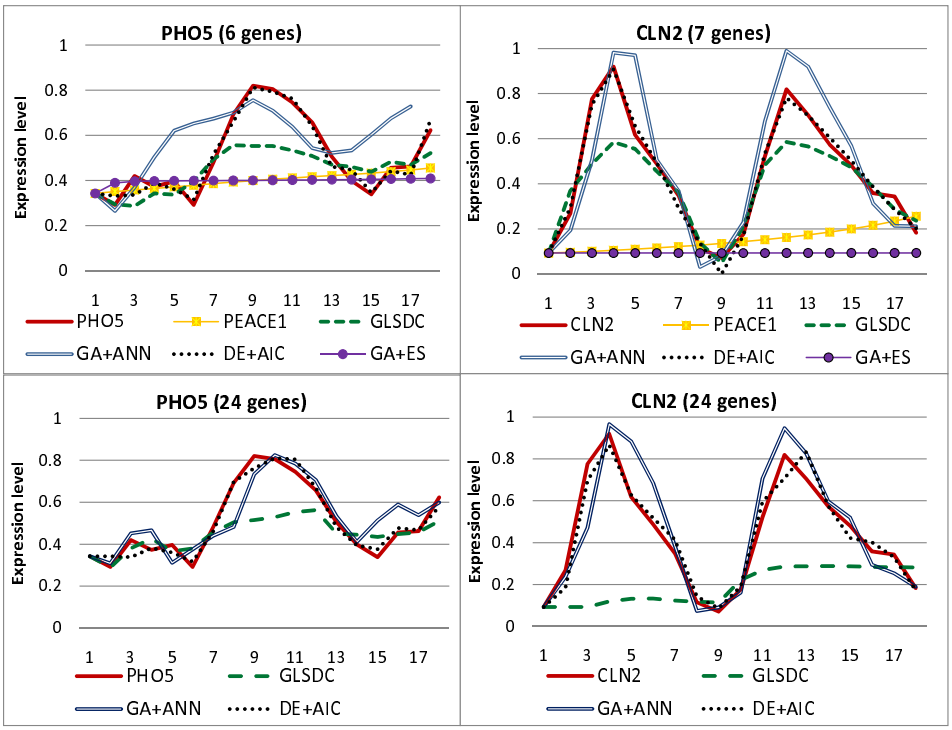


Figure A.11: Ability of algorithms to reproduce real data. The upper graphs display the real and the reproduced time series for the small-scale analysis, and the lower graphs for the medium-scale analysis.

for the small networks (Figure A.11). The small difference in data MSE values is due to the fact that the time series for GLSDC and GA+ANN are slightly shifted for this dataset, although overall behaviour is preserved. For the *CLN2* experiment, both ability to reproduce time series and observed MSE values differ significantly. Given similar unsatisfactory performance on larger synthetic datasets, experiments with the 24-gene real dataset were not pursued with these two methods. Note that DE+AIC displays the best overall ability to reproduce the data, followed by GA+ANN and GLSDC. While GA+ANN and DE+AIC maintain good data fit for the larger dataset on both genes analysed, GLSDC fails to reproduce the data for *CLN2* (Figure A.11) and the MSE values increase significantly (Figure A.12).

Due to noise and the limited number of time points available, it is possible that, although a model is capable of reproducing the experimental data, the connections identified are false positives, and the model invalid. We have analysed the connections obtained,

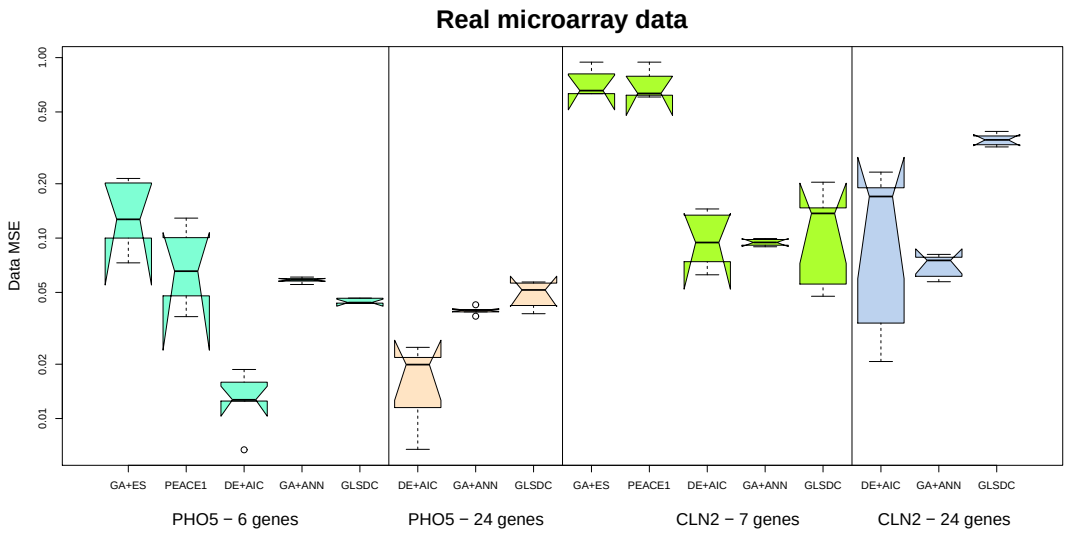


Figure A.12: Real data fit. Box plot representing data MSE for experiments with real microarray data. For the first gene analysed (PHO5), DE+AIC displays best behaviour, while for the second (*CLN2*), both GA+ANN and DE+AIC perform comparably well. Due to scale limitations, experiments with PEACE1 and GA+ES were not performed for the 24-gene network.

using data from the KEGG database and previous descriptions of the cell cycle from the literature [Chen et al., 2000; Oshima et al., 1996]. Table A.3 displays the percentage of known interactions out of the total number of interactions identified by each algorithm in each experiment. The remaining percentage of the interactions predicted are clearly wrong, (either opposite sign or false connection). Both overall values and values corresponding to the fittest individual over multiple runs are presented, in order to facilitate a global view over algorithm performance. These known interactions considered correspond not only to

Table A.3: Percent of interactions identified by each algorithm that are known to exist previously. Average (overall) and best values over multiple runs are displayed.

Experiment	**GA+ANN**	**DE+AIC**	**GLSDC**	**GA+ES**	**PEACE1**
6-gene *PHO5*	overall:92 best:100	overall:80 best:100	overall:41 best:50	overall:59 best:33	overall:25 best:0
24-gene *PHO5*	overall:11 best:0	overall:15 best:14	overall:39 best:40	-	-
7-gene *CLN2*	overall:38 best:50	overall:40 best:40	overall:53 best:60	overall:36 best:40	overall:69 best:75
24-gene *CLN2*	overall:29 best:60	overall:31 best:28	overall:18 best:26	-	-

Table A.4: Average number of overlooked important immediate interactions (from *SWI4/6* for *CLN2* and from *PHO4/2* for *PHO5*)

Experiment	**GA+ANN**	**DE+AIC**	**GLSDC**	**GA+ES**	**PEACE1**
6-gene *PHO5*	1.67	0.4	0.4	1	1.5
24-gene *PHO5*	1.75	1.2	0	-	-
7-gene *CLN2*	0	0	0	0	1
24-gene *CLN2*	0.6	0.6	1.8	-	-

transcriptional activation or repression, but also to protein interactions, (e.g. phosphorylation[5], ubiquitination[6]), that activate or repress transcription factors, hence influencing gene expression. For example, it is known that CLN3 and CDC28 work together to activate, (through phosphorylation), transcription factor SBF, (SWI4 and SWI6), which in turn activates gene *CLN1/2*; hence, *CLN3* and *CDC28* can also be considered as activators of these genes. The methods implemented often identify this type of interaction. Table A.4 presents the average number of previously known direct interactions missed by each algorithm in each experiment.

Note that, for some methods, the fittest individual identifies fewer interactions than the overall value, which confirms that good ability to reproduce data does not necessarily correspond to a model containing biologically relevant connections. Qualitative analysis indicates that, for the small networks, where all the genes are known to interact, the connections identified by the best-fitting methods are mostly correct. For the 7-gene experiment, two of the known interactions, (repression from *FAR1* and activation from *SWI6*), have been consistently assigned parameters with the wrong sign, by *all the methods*, in multiple runs. This indicates noise interference, which explains lower values compared to the similar 6-gene experiment. GLSDC, however, seems to identify a number of interactions comparable to the 6-gene experiment, which confirms that it is more robust to noise than the others. GA+ES and PEACE1 also seem to correctly identify many interactions, but, the fact that the simulated gene values are highly dependent on the rest of the network, means they are unable to reproduce the experimental data.

[5]Addition of a phosphate group to a specific protein, to activate or deactivate it.

[6]Addition of a ubiquitine group to a protein, typically for degradation purposes.

Introducing more genes into the analysis triggers a different response from each method and gene analysed. In the *PHO5* experiment, the percentage of correct interactions identified by GA+ANN and DE+AIC decreases markedly when analysing more genes, while the number of overlooked direct interactions increases, although data fit remains very good or even increases (from Figure A.12, GA+ANN is significantly better in the second experiment compared to the first). This relies on connecting nodes that are not immediately linked in the real network, and, given that many added nodes may not really be connected at all, this leads to a low percentage of true positives. GA+ANN suggests a positive autoregulation of *PHO5*, both with the small and large dataset, which can compensate for other missed interactions, and explain the improvement in data fit for the larger network. On the other hand, GLDSC maintains both quality of data fit, (though poorer than for the other two algorithms), and percentage of interactions, and adds fewer false interactions outside the *PHO* gene family (connections from *SIC1* and *APC/C*). This suggests that, when the added nodes are not connected to the existing ones, the algorithm is better at finding correct qualitative interactions, although fit obviously suffers.

In the second experiment, where most of the new nodes are connected to the initial network, GA+ANN and DE+AIC perform better both in terms of from the data fit and validity of interactions. However, the number of false positives increases when moving to the larger dataset. GLDSC finds many effects of *PHO* genes on *CLN2*, but these are not biologically plausible. At the same time, when moving to the larger dataset, it correctly adds a positive effect from *FUS3*, that affects the gene through *FAR1*, but fails to identify the *SBF* complex (*SWI4/6*) as an activator. The fact that it does not succeed in identifying the main activation link explains the poor performance when reproducing the data. DE+AIC and GA+ANN preserve the connections from *SWI4*, *SWI6* and *CLN3* from one analysis to the other, but at the same time add some false connections to *PHO80*, *PHO4* and *APC/C*.

All in all, the results indicate GA+ANN and DE+AIC as better choices when a continuous simulation of the system is required, with less concern for qualitative analysis of connections (i.e. a black box approach). GLDSC seemed to identify correct interactions in most experiments, but is not able to reproduce the data as well as the other two methods.

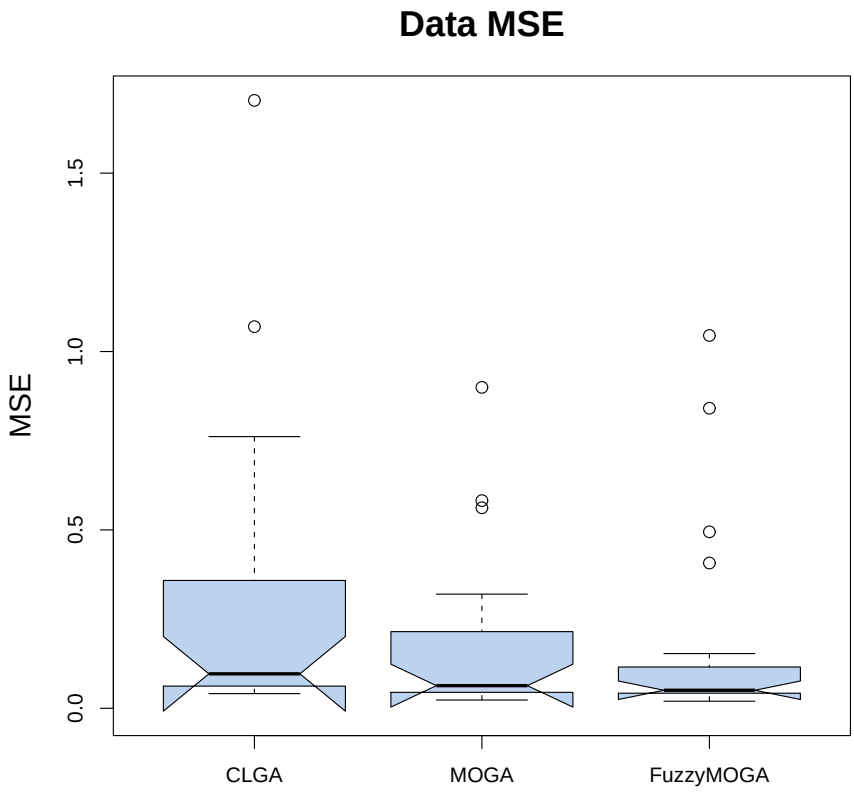


Figure A.13: Final data MSE for CLGA, MOGA and Fuzzy MOGA on the 2-gene dataset. The difference observed is not statistically significant at a 5% level.

Methods analysing all genes simultaneously displayed very poor performance in terms of reproducing the data, but succeeded in qualitatively identifying some correct interactions for the small-scale datasets.

A.2.5 Single versus multi-objective optimisation

As CLGA ([Tominaga et al., 1999]) and MOGA ([Koduru et al., 2004]) were found not to be suitable for large networks, these were compared only for a small network, i.e. a two-gene GRN. The approach used in MOGA is to split the squared error fitness of CLGA into separate objectives for each gene. Hence, in our experiments, we had 2 objective functions to minimise. The aim of this experiment is to compare CLGA with this MO approach and to identify the benefits of introducing *fuzzy domination*. The results of this experiment should be indicative of the improvement of other, more advanced EA approaches, when using MO

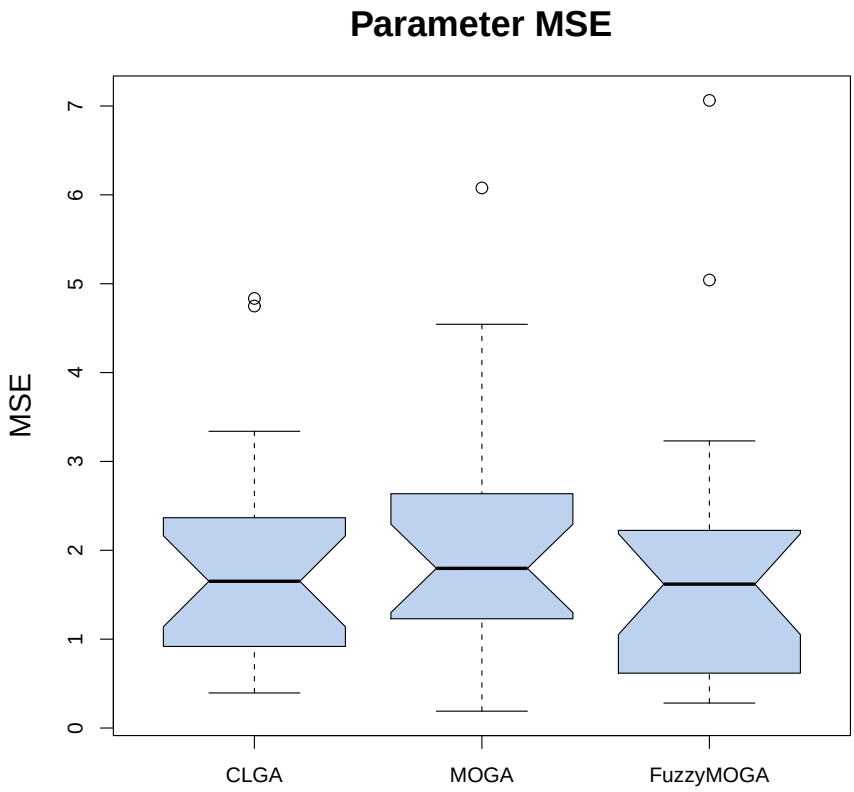


Figure A.14: Parameter MSE for CLGA, MOGA and Fuzzy MOGA on the 2-gene dataset.

optimisation.

In order to ensure the validity of our comparison we performed twenty 100,000-fitness call runs for each of the three algorithms and the results are summarised in Table A.5 and Figures A.13 and A.14. The averaged values in the table have been computed after eliminating the two worst and best two results for each algorithm.

Figure A.15, which shows the average, minimum and maximum squared error between the data and the model for the 20 best individuals in each generation, (one for each run), indicates that the MO algorithms perform better in terms of goodness of fit (the models found simulate the time series better than the CLGA). However, Figure A.13 indicates that this difference is not statistically significant at a 5% level. Indeed, a t-test shows this is likely to occur by chance 15% of the time. Similarly, although minimum values found for parameter MSE are better for the multi-objective approaches, the differences are not statistically significant at any meaningful level. Note, however, (Figure A.15), that the two MO

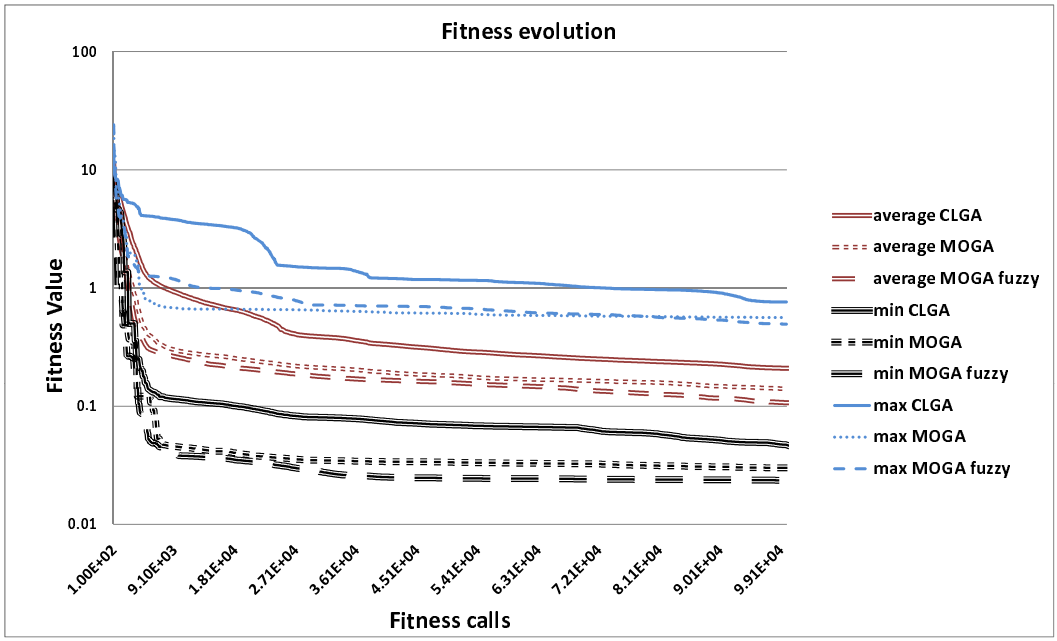


Figure A.15: Multi objective optimisation - fitness evolution. Comparison of CLGA, MOGA and Fuzzy MOGA on the 2-gene dataset during all generations.

approaches converge faster. This observed difference is confirmed by a t-test performed on fitness values obtained after 20,000 iterations (a fifth of total optimisation), that resulted in a p-value of 0.02 when comparing the single with the multi-objective approaches. However, no significant improvement is introduced by fuzzy dominance selection in this case.

A more general observation is that, if we perform two rankings of the 20 solutions obtained, (by goodness of fit and parameter quality, respectively), results differ, for all three methods. So, improved fitness does not necessarily mean better parameters. This suggests

Table A.5: Performance of classical vs multi objective real-coded GA over 20 runs using the 2-gene synthetic dataset.

Criteria	**CLGA**	**MOGA**	**Fuzzy MOGA**
Goodness of data fit (Best/ Average SE)	0.0411/ 0.2091	0.0232/ 0.1400	0.0198/ 0.1070
Parameter quality (Best/ Average SE)	2.3689/ 10.2550	1.1388/ 11.2255	1.6858/ 9.6762
Robustness (Kinetic orders /Rate constants variance)	0.3248/ 1.1070	0.3207/ 4.0854	0.2793/ 1.5181
Average running time	187.6s	302.8s	300.6s

that some parameters may be more important than others. In consequence, a slight perturbation of the more meaningful ones strongly influences the ability of the model to simulate the data. Another argument for this is the observed difference between the robustness of kinetic orders and that of rate constants, which suggests that variability in the latter impacts less on the goodness of fit. These observations also suggest that alternative models are possible, so that more precise discrimination is needed, as the MSE criterion not powerful enough, even for small systems.

In conclusion, we have shown that, splitting the squared error objective into smaller sub-objectives, for a MO approach, significantly speeds up convergence for EAs. Nevertheless, after a large number of iterations, final results are comparable. This could be due to the fact that this approach forces the algorithm to fit all parts of the time series at the same time, instead of allowing it to converge more slowly by improving only some of the objectives, which is an advantage, especially when dealing with high-dimensional problems. This suggests that, even when analysing only one gene at-a-time, we can still split the time series into shorter parts, to speed up convergence in a MO setting. Of course this is limited by the length of the time-series, so further analysis, to investigate to what extent this objective division is useful and at what point the overhead becomes greater than the gain, would be valuable.

A.2.6 Divide et impera?

The argument found in the literature *in favour of division* of the optimisation process into subproblems corresponding to each gene is *increased scalability*. This is due to a decrease in number of parameters (linear instead of quadratic dependency on the number of genes in the network [Noman and Iba, 2006]), and ease of solution evaluation, as only the time series for the current gene needs to be simulated. However, these arguments do not take into account the fact that this method has to be iterated for all genes, so, ultimately, the number of parameters and the number of simulated time series is the same (no significant increase in running time or computational power needed). Also, when simulating one series at-a-time, the values of the rest of the genes are considered to be those of the experimental data.

However, the effect of the current gene on the others is not taken into account and this can give the impression of finding a good solution when, in reality, the difference between the data and the simulation in a whole system setting could be larger. This effect is exacerbated for real (noisy) data. In order to compensate for this disadvantage, a complete network analysis can be performed, to fine tune the parameters obtained for each gene in each sub-problem.

In order to avoid resource issues and enable scale-up even when analysing the entire network simultaneously, parallelisation is clearly desirable. In a parallel setting, division loses its advantages, becoming less viable than the complete network analysis, which can be parallelised in a more convenient way, to avoid simulating only part of the network when evaluating individuals.

During our experiments, division proved to be more useful when analysing real data, statistically significant differences being observed in one of the small scale experiments. Nevertheless, in both of these experiments, probably due to noise, the two methods analysing the complete networks failed to reproduce the time series, even for a small number of genes. However, a more detailed analysis, in a multi-CPU setting, is required with respect to their behaviour with real microarray data.

Appendix B

Datasets

B.1 DREAM 4 data

DREAM (Dialogue for reverse engineering assessments and methods)[Prill et al., 2010] is an initiative to bring together research efforts for biological network inference, by compiling a set of publications and providing a platform to validate reverse engineering methods. A research competition is organised annually, where a set of data is published and the challenge is to predict the interactions within the system that produced the data. The DREAM4 competition for GRN inference concentrated on *in-silico* networks of two sizes (10 and 100 genes) [Marbach et al., 2010], and we have used the data available for two of the networks to validate the experiments presented in this work (Chapter 6).

Validation with synthetic data has the advantage that the system under analysis is known beforehand, so the interactions retrieved can be validated easily, unlike real systems where the underlying network is unknown. DREAM4 data have a further advantage, in that the *in-silico* networks have been designed to display the same connectivity patterns and modules seen in real-life GRNs [Marbach et al., 2009]. This increases the relevance of evaluation using these data.

The DREAM4 data used here consists of time-series data (5 experiments for the 10-gene network and 10 for the 100-gene network, each experiment consisting of 11 time points). Additionally, steady-state knockout measurements are provided after the knockout

of every gene in the network. These two data types have been used here as training data for the EGIA framework. For validation, the known gold-standards and a set of steady state dual-knockout experiments were used as test datasets. The results obtained on these by the EGIA framework have been compared to those obtained by the teams participating in the competition, which are available on the DREAM website [Prill et al., 2010].

B.2 *Saccharomyces cerevisiae* data

Yeast data used in this work consists of four time series datasets measured on different microarray platforms. These have been used together for analysis of data integration in Chapter 4, while the Spellman dataset has been also employed for algorithm validation in Appendix A.

Spellman dataset The Spellman dataset [Spellman et al., 1998], GEO accession number GSE22, is a dual-channel microarray time series measuring the gene expression pattern during the Yeast cell cycle. Two cell cycles are analysed every 7 minutes resulting in 18 time points.

PramilaL dataset The PramilaL dataset, [Pramila et al., 2006], GEO accession number GSE4987, measures two yeast cell cycles at time intervals of 5 minutes, resulting in 25 different samples. The experiments are performed with dual-channel microarrays, FHCRC Yeast Amplicon v1.1. The dataset contains a technical replicate, which has been used in this work as an additional time series.

PramilaS dataset This dataset [Pramila et al., 2002] has been obtained on the same platform and by the same authors of that in the previous paragraph, but contains only 13 time points measured every 10 minutes, and no replicate. The accession number of the dataset is GSE3635.

Table B.1: Set of 27 genes selected for network analysis for the *Drosophila melanogaster* dataset.

arm	bcd	cad	CrebA	Egfr	en	eve	ftz	fz
gt	hb	hkb	how	ken	Kr	L	Mef2	mxc
noc	os	pnr	ras	smo	sna	Tl	tor	twi

Hasse dataset The Hasse dataset ([Orlando et al., 2008], GEO accession GSE8799) describes the same cell-cycle process in yeast, containing time series data sampled with single-channel arrays (Affymetrix Yeast Genome 2.0) every 16 minutes. This results in 15 time points available. A technical replicate is also present and is again used as a separate time series during inference.

B.3 *Drosophila melanogaster* data

Drosophila melanogaster data have been used as a test case for the EGIA integrative platform (Chapters 6 and 7), so several types of data have been retrieved from publicly available databases. These include time series data from three platforms (retrieved from the Gene Expression Omnibus database [Barrett et al., 2011]), a set of knockout microarray experiments, PSWMs, known cis-regulatory modules and Gene Ontology annotations. For model validation, a set of previously known interactions has been used. A subnetwork of 27 genes involved in embryo development, listed in Table B.1, has been analysed.

Dual-channel (DC) dataset This time-course dataset analyses gene expression during Fly embryo development, using dual-channel microarrays (GEO accession GSE14086, [Liu et al., 2009a]). The dataset contains seven times points sampled at 1 and 2 hours intervals, up to 10 hours after egg laying. Three biological replicates are available, resulting in three time series in total.

Single-channel (SC) dataset The single-channel dataset [Tomancak et al., 2002], measured with Affymetrix arrays, contains gene expression measurements for 12 time points during *Drosophila m.* embryo development. Samples have been taken every hour up to 12

and a half hours after egg laying. Three biological replicates are present.

RNA-Seq (NGS) dataset A time-course dataset [NCBI, 2010] measured with the RNA-Seq technology (Illumina Genome Analyzer II) for the same process of embryo development in the Fruit Fly has been retrieved in order to analyse the applicability of the EGIA framework on this data type, and identify overlapping features with microarray data. This dataset contains 12 time points measured every 2 hours up to 24 hours after egg laying. Three technical replicates are available. The data have been retrieved from the NCBI Sequence Read Archive database [SRA, 2011] (accession number SRP001065).

Previously known interactions For validation purposes, a set of known interactions have been retrieved from DROID (DROsophila Interactions Database, [Murali et al., 2011]), version 2010_10. This consists of 16 pair-wise interactions between transcription factors and their target genes, for the 27-gene network under analysis.

Knockout datasets Five knockout microarray datasets have been retrieved form the Gene expression omnibus database, which contain knockout experiments for 8 genes and the corresponding wild-type measurements. The accession numbers for the datasets are GSE23346 ([Fox et al., 2010], Affymetrix Drosophila Genome 2.0 Array), GSE9889 ([Elgar et al., 2008], Affymetrix Drosophila Genome Array), GSE7772 ([Toledano-Katchalski et al., 2007], Affymetrix Drosophila Genome Array), GSE3854 ([Estrada et al., 2006], Affymetrix Drosophila Genome Array) and GSE14086 ([Liu et al., 2009a], dual-channel array). For these, the log-ratios between knockout and wild-type expression values have been used within the EGIA framework, as described in Chapter 5.

Binding site affinities A set of PSWMs (Section 2.2.2) for 11 transcription factors have been retrieved from [Pollard, 2011]. These matrices have been computed using DNA footprinting data from [Bergman et al., 2005]. Using promoter sequence information we used these matrices to compute binding site affinities, which were integrated in the EGIA framework as described in Chapter 5.

Cis-regulatory modules In order to compute binding site affinities using PSWMs, the promoter sequence for each gene is required. For the Drosophila genome, the RedFly database [Gallo et al., 2010] provides a set of known cis-regulatory modules (CRMs), which have been used for this purpose here. CRMs for 16 genes have been retrieved, while for the other genes the upstream 2Kbp sequence has been used to assess binding affinity.

Gene Ontology (GO) annotations GO [Ashburner et al., 2000] is a database of genes that have been annotated to have a specific function or to be involved in specific processes. These annotations come from various sources, and have been determined using technologies ranging from those in wet-lab experiments to computational methods. The database is a valuable source of meta-information that can be used in different ways. Here, we have used the GO platform to identify which of the gene products involved in the network analysed have been previously shown to display transcription factor activity. This information was used in the EGIA framework as described in Chapter 5.

Appendix C

Evaluation criteria and basic definitions

C.1 Quantitative evaluation criteria

Typical quantitative evaluation of models computes a distance or error measure between simulated and real data. This gives an indication of the extent of simulation abilities that the model displays. Several such criteria can be employed, depending on the evaluation requirements.

The Residual Sum of Squares (RSS) is defined as:

$$RSS = \sum_{i=1}^{T} (x_{t_i} - x_{s_i})^2 \tag{C.1}$$

where T is the number of data points available, x_{t_i} is the value of point i in the real data, while x_{s_i} is the corresponding simulated value. This distance measure is useful to compare models on the same dataset; however, due to the dependence on the number of time points, a comparison across datasets cannot be performed. In consequence, the Mean Squared Error (MSE) can be used for such cases:

$$MSE = \frac{RSS}{T} \tag{C.2}$$

where RSS is defined in equation C.1. This offers a measure independent of the dataset used for evaluation. However, the measurement unit is different than the data itself, so, in order to provide a more insightful criterion of the error magnitude, the Root Mean Squared Error (RMSE) can be used:

$$RMSE = \sqrt{MSE} \tag{C.3}$$

A normalised version of the RMSE, which divides this by the Mean of the sample, also exists. This is useful to compare algorithm behaviour between multiple datasets that may have different expression ranges.

C.2 Qualitative evaluation criteria

In this work, qualitative evaluation of models consists of determining the amount of previously known interactions that the model is able to identify. This has been performed either for one single model or, due to the stochasticity of the optimisation strategy used, by combining results from multiple runs, through voting and sorting the interactions descending by the number of votes obtained (i.e. the interactions contained in more models are more likely to be actually true). Several statistical measures have been used to evaluate a set of interactions predicted.

Sensitivity (also known as true positive rate (TPR) or recall), represents the fraction of all the previously known interactions that have been uncovered by the algorithm.

$$Sensitivity = \frac{TP}{P} \tag{C.4}$$

where TP represents the number of known interactions correctly uncovered by the algorithm, while P represents the total number of known interactions.

Specificity is a complementary measure to sensitivity, which describes the algorithm performance on the negative set of interactions (N, the number of interaction that are not

known previously).

$$Specificity = \frac{TN}{N} \tag{C.5}$$

where TN represents the number of interactions correctly identified by the model not to exist.

The false positive rate (FPR) represents the proportion of N that are wrongly predicted to be correct interactions.

$$FPR = \frac{FP}{N} \tag{C.6}$$

where FP represents the amount of interactions wrongly predicted to exist.

Precision is a quantity describing the quality of the predicted interaction as the fraction of predicted interactions that are actually present in the real system.

$$Precision = \frac{TP}{TP + FP} \tag{C.7}$$

These measures, used alone, give only partial indication of the performance of the algorithm in terms of the goodness of the predicted set of interactions. Hence, typically, at least two of these are employed together to describe the performance more accurately. Examples are the TPR and FPR. By plotting these two measures (i.e. TPR vs. FPR), after each individual interaction prediction, the ROC (Receiver Operating Characteristic) curve is obtained. This shows how the two measures evolve during the prediction process, assuming that in the end all possible interactions will be predicted as positive. Ideally, the FPR should be null until the TPR reaches the value of 1, i.e. all the positive interactions are predicted first. As a combined measure of quality of predicted interaction, the area under the ROC curve (AUROC) can be computed. A value of 1 for this area indicates a perfect prediction, while a value of 0.5 represents a prediction close to random.

A combined measure similar to AUROC, using precision vs. recall, is the AUPR (Area Under the Precision Recall curve). This plots the precision and recall of the prediction and computes the area under this curve. Ideally, the precision should maintain a value of 1 until the recall reaches the same value, resulting in an AUPR value of 1. A value that represents

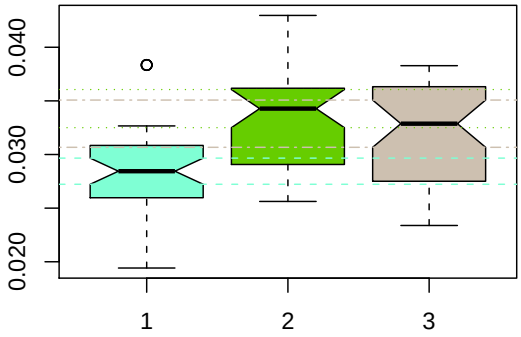


Figure C.1: Notched box-plot example. Three different distributions are displayed, the first one with values significantly lower than the other two. The horizontal lines show the intervals defined by the notches for each distribution, which overlap for the last two plots.

the AUPR achievable by a random classification of interactions depends on the distribution of N and P:

$$\text{AUPR}_{random} = \frac{P}{P+N} \tag{C.8}$$

C.3 Displaying results

Given that the platform presented in this work is based on stochastic optimisation, in order to provide a consistent analysis of results, multiple runs of the algorithm have been performed. The performance achieved (in qualitative or quantitative terms, i.e. MSE, RMSE, etc.) has often been displayed as box plots, with their notched variant [McGill et al., 1978]. Box plots display the distribution of values over the multiple runs, i.e. minimum, maximum and quartiles. Notched box plots allow for significance analysis of results, hence, in this work, enable identification of significant changes in algorithm performance. This is performed by defining intervals around the median, displayed as notches, where two distributions should not overlap if they are significantly different. The significance level typically used in box plots is 5%. For example, Figure C.1 shows three such distributions, with the notches defining an interval around the medians. The last two do not display a sig-

nificant difference, although medians are different. However, the first distribution displays significantly lower values that the other two. Throughout the thesis, box-plots have been generated using R software, while for the results presented in Appendix A, they have been created using the Free Statistics Software from Wessa.net [Wessa, 2008].

C.4 Wavelet analysis

Wavelets [Kaiser, 1994] are a mathematical tool for time-scale signal analysis: at large scale, low frequencies present in the signal can be readily extracted, while small scale analysis detects high frequency components. In signal decomposition in general, high frequencies correspond to noise, while low frequencies correspond to the signal itself [Kaiser, 1994]; this also applies to time series gene expression measurements. Here, we have used discrete wavelet decomposition to obtain wavelet coefficients for gene signals at different scales, (also known as *levels*). This type of decomposition uses a set of functions, called wavelets, which are generated by contracting and dilating a base function, i.e. the *mother wavelet*, in discrete steps [Kaiser, 1994]:

$$\Psi_{j,k}(t) = \frac{1}{\sqrt{s^j}}\Psi(\frac{t}{s^j} - k\tau) \tag{C.9}$$

Here, $\Psi_{j,k}$ is the wavelet obtained from Ψ, the mother wavelet, by using s, a fixed scaling step, which is usually 2, and τ, a translation factor, usually 1. This results in a discrete sampling of the time-scale space. The resulting wavelets are used to represent the signal as a discrete superposition:

$$f(t) = \sum_{j,k} w_{j,k}\Psi_{j,k}(t) \tag{C.10}$$

where $w(j,k)$ represent the wavelet transform coefficients, which describe components of the signal, corresponding to scale window j and time window k. In practice, to obtain these coefficients, an iterative approach is used, which builds coefficients for the upper half of the frequency spectrum (considering $s = 2$ and $\tau = 1$), filters these frequencies out and repeats the process for the lower half, after sub-sampling the signal by 2. This results

in different *levels* for coefficients, with low levels corresponding to small scale i.e. high frequencies, and high levels to high scale i.e. low frequencies. Having 2^k time points in the data, 2^{k-1} level 1 coefficients are computed, for short time windows (total time divided by number of coefficients), 2^{k-2} level 2, for double-sized time windows, while the last level, k, contains just 2 coefficients, for large time windows. Each of these coefficients indicates the importance of the current frequency level present in the signal in the current time window, i.e. what part of the variation in the data is attributable to low/high frequency effects. This results in high time resolution and low frequency resolution at small scale and high frequency resolution and low time resolution at large scale.

Throughout the thesis, wavelet decomposition was performed using the MATLAB toolbox WaveLab [Donoho et al., 1995].

Appendix D

List of publications

2011

Sîrbu, A., Ruskin, H. J. and Crane, M. Stages of Gene Regulatory Network Inference: the Evolutionary Algorithm Role, *Evolutionary Algorithms*, pages 521-545, InTech, ISBN 978-953-307-171-8.

Sîrbu, A., Ruskin, H. J. and Crane, M. Integrating Heterogeneous Gene Expression Data for Gene Regulatory Network Modelling, Presented in *ECCS 2010*, Accepted for publication in the *Special Issue ECCS10 Theory in Biosciences*.

Sîrbu, A., Kerr, G., Ruskin, H. J. and Crane, M. NGS vs dual- and single-channel microarray data: sensitivity analysis for differential expression and clustering, In preparation

2010

Sîrbu, A., Ruskin, H. J. and Crane, M. Comparison of evolutionary algorithms in gene regulatory network model inference, *BMC Bioinformatics*, 11(59).

Sîrbu, A., Ruskin, H. J. and Crane, M. Cross-platform microarray data normalisation for regulatory network inference, *PLoS ONE*, 5(11): e13822.

Sîrbu, A., Ruskin, H. J. and Crane, M. Regulatory network modelling: Correlation for structure and parameter optimisation, In *Proceedings of The IASTED International Conference on Computational Bioscience*, Cambridge, Massachusetts, 1-3 Nov.

Sîrbu, A., Ruskin, H. J. and Crane, M. Modelling Gene Regulatory Networks - An Integrative Approach, *ERCIM News*, 81(36-37), Special Issue: Computational Science/Scientific Computing. Simulation and Modelling for Research and Industry.

Abstracts

Sîrbu, A., Ruskin, H. J. and Crane, M. (2011). *Stages of Gene Regulatory Network Inference: the Evolutionary Algorithm Role*

`http://www.intechopen.com/articles/show/title/stages-of-gene-regulatory-network-inference-the-evolutionary-algorithm-role`

In all genetic material, the DNA information encoded is required by the cell to create proteins that are vital for its mechanisms . Each cell has access to the same information, but behaviour depends on tissue type. Hence, regulatory mechanisms exist, which control protein level and function, dependent on the environment. One such regulatory mechanism, transcriptional regulation, is controlled by proteins called transcription factors (TFs). These TFs bind to the region upstream of the gene that needs to be expressed and regulate its transcription in a positive or negative way (up- or down-regulation). Such interactions contribute to create a complex network of regulation, known as gene regulatory network (GRN).

Uncovering interactions between genes and their products has been a major aim of Systems Biology over recent years. The objective is to gain a better understanding of the functioning of different organisms, together with discovery of disease markers and new treatments . The class of computational methods known as Evolutionary Algorithms (EAs) have demonstrated relevance at different stages of these investigations. This chapter, in con-

sequence, presents an overview of approaches and issues in GRN modelling and inference, and discusses the role of EAs in this regard.

GRN analysis has been facilitated by the advent of technologies for measuring gene expression. These include mature technologies such as *qrtPCR*, , suitable for a limited number of genes, and *microarrays*, which allow for high-throughput measurement of thousands of genes at the same time. More recently, *RNA-Seq* measurements have become available, due to advances in high throughput sequencing technology. However, these data are still scarce, due to high experimental costs. Characterised as they are by high dimensionality and noise levels, analysis of these data types is far from trivial.

Three different analysis stages can be identified for GRN inference: (i) expression pattern analysis, (ii) mathematical modelling from expression data and (iii) integrative modelling. At each of these, and most particularly at the latter stage, EAs have an important role to play, due to the strength and flexibility of these search methods.

Expression pattern analysis is largely concerned with application of classification and clustering methods to gene expression data. Clustering, as a first step towards GRN modelling, together with classification, typically used to distinguish between tissue types (e.g. control/treatment or healthy/infected), give valuable insight on gene involvement in different processes. EAs are typically employed at this stage, with some success, for feature selection and clustering.

At the second stage, a GRN model is created to explain the data, which can be used for *in silico* simulation and process analysis under various criteria. Such a model is built by reverse engineering from available time course expression data, with inferential algorithms used to fit model parameters to the data, using evolutionary optimisation. Different EA approaches are presented here, for inferences on *discrete qualitative* to *continuous quantitative* models. Consequently, the discussion includes *classical* to *hybrid* EAs, (hybridisation by local search, divide-and-conquer and nested optimisation), and identification of strengths and weaknesses. Earlier work on a comparison framework, will also be described in this context.

A general limitation in GRN modelling is that, although qualitative models can be built

for entire GRNs, quantitative analysis is still restricted to sub-networks, due to limited data available and the large number of parameters to be optimised. Quantitative models allow for a better representation of interaction links, and for continuous simulation of dynamical behaviour, but the limitations in size and accuracy has impeded their use in real-world scenarios. Consequently, a third stage in network inference, integrative analysis, aims at reconciling different sources for the large amount of biological data available, in order to improve reliability of the inferential process, and realism of the models. Additional data types, which can contribute to synthesis, include DNA-protein interactions, knockout/knockdown experiments, binding site affinities, as well as known TFs and RNA interference measures.

To date, integration efforts are sparse. Nevertheless, examples of approaches based on EAs are presented here, although these typically combine only one additional data type with expression measurements. Ideally, all such related data should contribute to the inferential process. With this aim, a novel algorithm, based on evolutionary computation, that aims at large scale data integration for quantitative modelling, is also outlined, and the advantages and disadvantages of EAs for data unification discussed.

Sîrbu, A., Ruskin, H. J. and Crane, M. , *Integrating Heterogeneous Gene Expression Data for Gene Regulatory Network Modelling*

Gene regulatory networks are complex biological systems that have a large impact on protein levels, so that discovering network interactions is a major objective of Systems Biology. Quantitative GRN models have been inferred, to date, from time series measurements of gene expression, but at small scale, and with limited application to real data. Time series experiments are typically short, (number of time points of the order of 10), while regulatory networks can be very large, (containing hundreds of genes). This creates an under-determination problem, which negatively influences the results of any inferential algorithm. Presented here is an integrative approach to model inference, which has not been previously discussed, to the authors' knowledge. Multiple heterogeneous expression time series are used to infer the same model, and results are shown to be more robust to noise and

parameter perturbation. Additionally, a wavelet analysis shows that these models display limited noise over-fitting within the individual datasets.

Sîrbu, A., Kerr, G., Ruskin, H. J. and Crane, M. , *NGS vs dual- and single-channel microarray data: sensitivity analysis for differential expression and clustering*, In preparation.

With the fast development of high throughput sequencing technologies, a new generation of genome-wide gene expression measurements is under way. Based on mRNA sequencing, which complement the already mature technology of microarrays, this is expected to overcome some of the latter's disadvantages. These data pose new challenges, however, as strengths and weaknesses have yet to be fully identified, while very few study analyses have been reported to date. Ideally, Next Generation Sequencing measures can be integrated for more comprehensive gene expression investigation, to facilitate analysis of whole regulatory networks. At present, however, the nature of these data is not well understood.

In this paper, we study three alternative gene expression time series datasets for the *Drosophila melanogaster* embryo development, in order to compare three measurement techniques: RNA-seq, single-channel and dual-channel microarrays. The approach consists of a sensitivity analysis for differential expression and clustering, and aims to highlight different features of the three datasets.

In general, the RNA-seq dataset displayed highest sensitivity to differential expression and robustness to stringent thresholds. The single-channel data performed similarly for the differentially expressed genes common to gene sets considered. Cluster analysis was used to identify different features of the gene space for the three datasets, with similarities found for the RNA-seq and single-channel dataset at fine-grained level, and complementary information from the RNA-seq dataset at coarse-grained clustering.

Sîrbu, A., Ruskin, H. J. and Crane, M. (2010). ***Comparison of evolutionary algorithms in gene regulatory network model inference***

`http://www.biomedcentral.com/1471-2105/11/59`

The evolution of high throughput technologies that measure gene expression levels has created a data base for inferring GRNs (a process also known as *reverse engineering* of GRNs). However, the nature of these data has made this process very difficult. At the moment, several methods of discovering *qualitative* causal relationships between genes with high accuracy from microarray data exist, but large scale *quantitative* analysis on real biological datasets cannot be performed, to date, as existing approaches are not suitable for real microarray data which are noisy and insufficient.

This paper performs an analysis of several existing evolutionary algorithms for *quantitative* gene regulatory network modelling. The aim is to present the techniques used and offer a comprehensive comparison of approaches, under a common framework. Algorithms are applied to both synthetic and real gene expression data from DNA microarrays, and ability to reproduce biological behaviour, scalability and robustness to noise are assessed and compared.

Presented is a comparison framework for assessment of evolutionary algorithms, used to infer gene regulatory networks. Promising methods are identified and a platform for development of appropriate model formalisms is established.

Sîrbu, A., Ruskin, H. J. and Crane, M. (2010). ***Cross-platform microarray data normalisation for regulatory network inference***

`http://www.plosone.org/article/info:doi/10.1371/journal.pone.0013822`

Inferring Gene Regulatory Networks (GRNs) from time course microarray data suffers from the dimensionality problem created by the short length of available time series, compared to the large number of genes in the network. To overcome this, data integration from diverse sources is mandatory. Microarray data from different sources and platforms are

publicly available, but integration is not straightforward, due to platform and experimental differences.

We analyse here different normalisation approaches for microarray data integration, in the context of reverse engineering of GRN quantitative models. We introduce two pre-processing approaches based on existing normalisation techniques, and provide a comprehensive comparison of normalised datasets.

Results identify a method based on a combination of *Loess* normalisation and *iterative K-means* as best for time series normalisation for this problem.

Sîrbu, A., Ruskin, H. J. and Crane, M. (2010). ***Regulatory network modelling: Correlation for structure and parameter optimisation***

`http://www.actapress.com/Abstract.aspx?paperId=41573`

Due to the limitations of available gene expression data, (i.e. noise and size of time series), modelling gene regulatory networks is still restricted, especially in terms of their quantitative analysis. To date, the only criterion used for model evaluation is the residual error between observed and simulated data. This does not assign good fitness to models that can simulate the general oscillation, but are shifted with respect to observed data. Given that oscillatory behaviour of such complex systems is mostly driven by the topology of regulatory networks, these models may contain important information on network structure, which can shed light on evolutionary parameter optimisation. In consequence, a second model evaluation criterion is introduced here, namely the Pearson correlation coefficient between simulated and observed time series, which enables good fit to be assessed for candidate solutions able to approximate the general behaviour seen in the data. This is employed in a nested optimisation algorithm, which separately analyses the structure and parameters of the models. The method is evaluated using both synthetic and real microarray gene expression data, (Yeast cell cycle), and results show that models obtained in this way display more plausible connections, also contributing to simulation of quantitative behaviour.