Department of Computer Languages
University of Seville

Evolutionary Biclustering of Gene Expression Data: Shifting and Scaling Pattern-based Evaluation.

Beatriz Pontes Balanza, 28782976V
bepontes@us.es

Supervised by
Prof. Dr. Raúl Giráldez Rojo
Prof. Dr. Jesús S. Aguilar Ruiz

European Doctoral Dissertation

Contents

I Preamble **15**

1 Introduction **17**
1.1 Motivation 17
1.2 Hypotheses 20
1.3 Summary of Contributions 21
1.4 Document Structure 24

2 Bioinformatics Background **27**
2.1 The Principles of Life 28
2.1.1 Structural Cell Biology 29
2.1.2 Molecular Genetics 32
2.2 Bioinformatics Research Fields 34
2.2.1 Sequence Analysis and Comparison 34
2.2.2 Comparative Genomics 36
2.2.3 Gene Expression Analysis 39
2.2.4 Proteomics 41
2.2.5 Systems Biology 44
2.2.6 Other Related Research Fields 45
2.3 Data Mining in Bioinformatics 47
2.3.1 Supervised methods 48
2.3.2 Unsupervised methods 51
2.4 Summary 54

II Foundations **55**

3 Microarray: Technology and Analysis **57**
3.1 Microarray Technology 57
3.1.1 Applications 59
3.1.2 Experiment Cycle 61
3.2 Gene Expression Microarray Analysis 62

3.2.1 Identification of Differentially Expressed Genes 63
3.2.2 Classification and Clustering Analyses 65
3.2.3 Other Analyses 67
3.2.4 Biological Databases for Verification and Interpretation 68
3.3 Summary 72

4 Biclustering Algorithms 73
4.1 Bicluster Unified Notation 74
4.2 Gene Expression Patterns 75
4.2.1 Shifting and Scaling Expression Patterns 76
4.3 Evaluation Measures 78
4.3.1 Variance 79
4.3.2 Mean Squared Residue 80
4.3.3 Scaling Mean Squared Residue 80
4.3.4 Relevance Index 81
4.3.5 Correlation-based Measures 81
4.3.6 Similarity Score 84
4.4 Biclustering Approaches Based on Evaluation Measures 85
4.4.1 Iterative Greedy Search 86
4.4.2 Stochastic Iterative Greedy Search 89
4.4.3 Nature-inspired Meta-heuristics 92
4.4.4 SVD and Clustering-based Approaches 97
4.5 Non Metric-based Biclustering Algorithms 98
4.5.1 Graph-based Approaches 98
4.5.2 One-way Clustering-based Approaches 101
4.5.3 Probabilistic Models 103
4.5.4 Linear Algebra 106
4.5.5 Optimal Reordering of Rows and Columns 107
4.6 Biological Validation for Biclusters 109
4.7 Summary 110

5 Evolutionary Computation 113
5.1 Common Features of EC 113
5.1.1 Individual Encoding and Initialization 115
5.1.2 Evaluation 116
5.1.3 Selection 116
5.1.4 Reproduction: Genetic Operators 117
5.2 EC Paradigms 118
5.2.1 Evolution Strategies 118
5.2.2 Evolutionary Programming 119
5.2.3 Genetic Algorithms 119

5.2.4 Genetic Programming 120
5.3 Evolution Models 120
5.3.1 Generational EAs 120
5.3.2 Steady State EAs 121
5.3.3 Generational Gap EAs 122
5.4 Advanced Evolutionary Algorithms 122
5.4.1 Multi-Objective EAs (MOEAs) 122
5.4.2 Memetic EAs 125
5.5 Summary 127

III Proposals 129

6 Standardization-based Evaluation Measures 131
6.1 Standardization Procedure 131
6.2 Maximal Standard Area (MSA) 132
6.3 Virtual Error (VE) 134
6.3.1 Virtual Error Analysis 137
6.4 Evolutionary Biclustering with Virtual Error 138
6.4.1 Sequential Multi-objective Biclustering 139
6.4.2 Experimental Results on Real Data Microarrays 143
6.5 VE^t: Transposed Virtual Error 150
6.5.1 Analytical Analysis 151
6.5.2 Noise Robustness Analysis 154
6.6 Conclusions 156

7 Evolutionary Biclustering based on Expression Patterns 159
7.1 Biclusters Evaluation in Evo-Bexpa 160
7.1.1 Bicluster Volume 160
7.1.2 Overlapping among Biclusters 162
7.1.3 Gene variance 163
7.2 Evolutionary Algorithm 164
7.2.1 Individual Encoding and Initialization 166
7.2.2 Generational Change 168
7.2.3 Fitness Function 170
7.3 Experimental Results and Discussion 171
7.3.1 Analysis of Parameters 172
7.3.2 Synthetic Data Experiments 178
7.3.3 Experiments on Real DataSets 181
7.3.4 Biological Assessment 184
7.4 Conclusions 188

8 Conclusions and Future Works **189**
8.1 Conclusions . . . 189
8.1.1 Bicluster Evaluation Measures . . . 190
8.1.2 Evolutionary Biclustering of Gene Expression Data . . . 192
8.2 Future Works . . . 193
8.2.1 Memetic Evo-Bexpa . . . 193
8.2.2 Integrated Biological Validation and Interpretation of Biclusters . . . 194

IV Appendices **219**

A Improved Biclustering on Expression Data through Overlapping Control **221**
A.1 Overlapping Control Mechanisms . . . 222
A.1.1 Random Replacement . . . 222
A.1.2 Overlapping Control with a Matrix of Weights . . . 223
A.2 CC Algorithm . . . 224
A.3 Re-adaptation of Cheng & Church Algorithm . . . 226
A.4 Experimental Results . . . 229
A.4.1 Yeast *Saccharomyces Cerevisiae* Cell Cycle Expression Dataset . . . 233
A.4.2 Human B–cells Expression Data . . . 235
A.4.3 *Colon Cancer* dataset . . . 235
A.4.4 Comparison . . . 237
A.5 Conclusions . . . 237

List of Figures

2.1 Structure of a prokaryotic cell 31
2.2 Structure of an eukaryotic cell 32
2.3 Phylogenetic tree 38
2.4 Structure levels of proteins molecules 42
2.5 Knowledge discovery in databases 47

3.1 The first DNA microarray 58
3.2 Microarray production process 59
3.3 Microarray experiment cycle 62

4.1 Graphical representation of a bicluster containing a perfect shifting pattern. 77
4.2 Graphical representation of a bicluster containing a perfect scaling pattern. 78
4.3 Graphical representation of a bicluster containing perfect shifting and scaling patterns. 79

6.1 Example to illustrate the Maximal Standard Area (MSA). In (a) a bicluster example; in (b) its standardization; in (c) the MSA. 133
6.2 Virtual Error computation diagram. 136
6.3 Example to illustrate the Virtual Error. 136
6.4 VE^t behaviour in biclusters with errors. 155
6.5 VE behaviour in biclusters with errors. 156
6.6 MSR behaviour in biclusters with errors. 156

7.1 Genes size term in volume evaluation. 162
7.2 Binary string for individuals representation. 166
7.3 One-point crossover. 169
7.4 Two-points crossover. 169
7.5 w_g influence over the different bicluster features. 175
7.6 w_c influence over the different bicluster features. 175

7.7 w_s influence over the different bicluster features. 176
7.8 w_{ov} influence over the different bicluster features. 177
7.9 w_{var} influence over the different bicluster features. 178
7.10 Gene and condition match scores for ISA, xMotifs, OPSM, BIMAX, CC and Evo-Bexpa in synthetic experiments. 180
7.11 Number of significant biclusters for different w_g values. 185
7.12 Number of significant biclusters for different w_c values. 185
7.13 Number of significant biclusters for different w_s values. 186
7.14 Number of significant biclusters for different w_{ov} values. 187
7.15 Number of significant biclusters for different w_{var} values. . . . 187

A.1 Example of a bicluster containing a random element (showed in bold). The original value of the element was 153. 226
A.2 Flow chart representing the re-adapted multiple node deletion phase. 229
A.3 Flow chart representing the re-adapted addition phase. 230
A.4 Examples of biclusters found on the Yeast dataset. Each column shows three pictures of the same bicluster: full bicluster, 20% and 10% of the genes, respectively. X-axis represents each of the experimental conditions in the bicluster, while Y-axis represents the expression level of the genes in each bicluster. 234
A.5 Examples of biclusters found on the Human dataset. X-axis represents each of the experimental conditions in the bicluster, while Y-axis represents the expression level of the genes in each bicluster. 236
A.6 Examples of biclusters found on the Colon dataset. X-axis represents each of the experimental conditions in the bicluster, while Y-axis represents the expression level of the genes in each bicluster. 236

List of Tables

6.1 Datasets used in the experimentation. 144
6.2 Values of δ used in the settings SMOB-δ and SMOB-Δ. 144
6.3 Parameter settings for the algorithm. 145
6.4 Average MSR obtained on each dataset. 146
6.5 Average VE obtained on each dataset. 146
6.6 Average gene variance and volume obtained on each dataset. . 147
6.7 Average GFD values for each of the three sub-ontologies of GO. 149
6.8 Number of significant biclusters for the three GO ontologies, at two different levels. 149

7.1 Evolutionary parameter setting for Bexpa. 171
7.2 Experimental values for configuration parameters 174
7.3 Qualitative influence of the configuration parameters over the different objectives. 177
7.4 Datasets used in the experimentation. 181
7.5 Summary of experimental results for the microarrays in table 7.4. 182
7.6 Validation results with GO hierarchy level limitations. 188

A.1 Main information for each dataset. 231
A.2 CC average results for each dataset. Standard deviation is given between brackets. 231
A.3 CC-R average results for each dataset. Standard deviation is given between brackets. 232
A.4 SEBI average results for each dataset. Standard deviation is given between brackets. 237

List of Algorithms

1 A general scheme of an EA 114
2 Local search scheme 126

3 **SMOB** for sequential covering 140
4 Procedure **MOEB** 140

5 Function **Evo-Bexpa** 165
6 Function **Bexpa** 165

7 Cheng and Church's original algorithm (CC). 225
8 Cheng and Church's re-adapted algorithm (CC-R). 228

Acknowledgments

I would here like to express my gratitude to the many people who have helped and supported me during the time it took me to develop this PhD Thesis.

To start with, I am very thankful to Professor Eduardo F. Camacho, Dr. Alfonso Cepeda and Dr. David Muñoz for introducing me into the research world. They are my former college research advisers, and the reason why I decided to pursue a career in research.

Secondly, I would like to thank my mentors, Professor Jesús S. Aguilar and Dr. Raúl Giráldez, for their guidance and assistance in this work. Jesús wrote a research proposal that supposed the start point in my investigation. Raúl, as my daily supervisor, has been encouraging, patient and supportive, and has given me the freedom to choose different exploration paths in our investigation. Both of them have always provided insightful discussions about our research.

My gratitude is also extended to Professor José C. Riquelme, who has provided me with expert advices and instructive comments. It is very comforting to know that I am in such good hands in the *Computer Languages Department*.

A big thank to those authors who have occasionally collaborated in this work. I would like to specially thank Dr. Federico Divina, I truly appreciate his help in giving shape and polishing my work. Thanks to Dr. Fermin L. Cruz for being so in tune with this research and helping me find the best ways to finish it. Thank you to Dr. Eloisa Andújar and Dr. Mónica Pérez, from the genomic unit in the *Andalusian Center for Molecular Biology and Regenerative Medicine* (CABIMER). They have significantly contributed to giving biological insights to our research results.

I can't mention all the members in the Bioinformatics Research Group, both at the University of Seville and Pablo de Olavide, but thank you very much to each of you for your support and belief in the potential of my research. In particular, thank you to Dr. Cristina Rubio, my tutor and also my office mate, for her invaluable inputs and comments.

I am also very grateful to Professor Elena Marchiori and Pavol Jancura for providing me with a rich and fertile environment to study and explore new

ideas. I have been very lucky to be a part of their research team during my stay at the *Department of Information and Knowledge Systems*, at Radboud University.

I also thank the wonderful staff in the *Computer Languages Department*, for always being so helpful and friendly.

And last, but not least, I would like to express an ocean of thanks to my friends and family, for all their incomparable love and support.

Abstract

Biclustering has become a very popular data mining technique due to its ability to explore at the same time two different dimensions, as opposed to clustering techniques, that make use of only one dimension. Gene expression data offer a suitable framework for the application of biclustering algorithms, where enormous amount of information are being produced due to technological advances. Microarray technology offers the possibility of quantifying the expression levels of thousand of genes simultaneously. Furthermore, several experimental conditions may be taken into account, producing thus numerical matrices of expression in which one dimension refers to genes (rows) and the other refers to samples or experimental conditions (columns). This constitutes an ideal scenario for applying biclustering since exploring both dimensions simultaneously would provide the analyst with useful knowledge (subset of genes showing a common tendency under a subset of conditions). Different heuristics have been proposed in order to discover interesting biclusters in data. Many of them are guided by a measure that determines the quality of biclusters. Thus, defining a quality measure represents a key factor in the search of biclusters. The most widespread measure for biclustering of gene expression data has been the *mean squared residue* (MSR), that can identify correctly some types of patterns, but fails at discovering others. In this PhD Thesis we plan to first make an overview of the most used biclustering techniques for gene expression data, and second to propose a both effective and efficient quality bicluster measure, together with a fully customizable evolutionary biclustering technique. The evaluation measures we propose (named VE and VE^t) are based on the use of a standardization procedure in order to perform a comparison among the genes and conditions tendencies, with independence of their specific expression values. On the other hand, our search heuristic (called Evo-Bexpa) offers the possibility of guiding the search towards certain desired characteristics in the biclusters, by adjusting several weights that give preference to some bicluster features over others. Also, new objectives can be easily incorporated into the search. Our work also include a wide set of experimental tests which helps us to validate our proposal, both statistically and biologically.

Part I

Preamble

Chapter 1

Introduction

1.1 Motivation

DNA Microarray technologies are used to analyse the expression level of many genes in a single reaction quickly and in an efficient manner. Different types of microarray chips have been designed for different investigations, being expression chips the most common application. They are used to determine the expression patterns of genes that correspond to different samples or experimental conditions, where the samples may vary according to experimental conditions and/or physiological states. They may even be extracted from different individuals, tissues or developmental stages (Lesk (2008)). The applications of this kind of microarrays involve determine genes functions, find new genes, study genes regulation and assess how they have evolved over time.

The raw data of a microarray experiment is an image, in which the colours and intensities reflect the expression level of each gene and each sample. This image is processed in order to obtain a numerical gene expression matrix, in which rows correspond to the genes under study and the columns refers the different samples or experimental conditions. A special characteristic of these expression matrices is that they are very unbalanced, in the sense that the number of genes is much larger (usually thousands of genes) than the number of samples (usually less than a hundred) (Watson (2003)). Therefore, analysing these kind of matrices implies understand the relationships of a space of lots of variables (genes) from only a few measured points (experimental conditions).

In order to obtain relevant knowledge from microarray data, similarities among genes and samples need to be carried out in many different ways, depending on the specific application. Focussing the analysis on the genes,

one of the most studied goals is to extract information on how gene expression patterns vary among the different samples, finding groups of co-expressed genes. If two different genes show similar expression patterns across the samples, this suggests a common pattern of regulation, possibly reflecting some kind of interaction or relationship between their functions (Baldi & Hatfield (2002)).

Within data mining techniques it is possible to differentiate two main sets of algorithms, depending on the use (supervised learning) or not (non-supervised learning) of previous knowledge on the data. Classification has been extensively studied within gene expression data as a supervised technique [Golub et al (1999); Ben-Dor et al (2000); Asyali et al (2006); Schachtner et al (2008); Buness et al (2009)], where labelled data is used to create an algorithm able to assign any new input data to its proper class.

On the other hand, non-supervised learning is used when no previous assignations are available; the goal is to divide the data into clusters of samples and to identify the differences between the genes that characterize such groups. The application of clustering techniques to gene expression data has also been broadly studied in the literature [Jiang et al (2004); Xu et al (2005); Handl et al (2005)]. Nevertheless, there exists two main restrictions in the use of clustering algorithms: (1) genes are grouped together according to their expression patters across the whole set of samples, and (2) each gene must be clustered into exactly one group. This last limitation is two-fold: firstly, it means that a certain gene cannot be present in different groups, thus forbidding overlapping among clusters; secondly, it confines each gene to be a member of any cluster, even if it is not co-regulated with any of the other genes in the cluster.

However, genes might be relevant only for a subset of samples. This is essential for numerous biological problems, such as the analysis of genes contributing to certain diseases, assigning biological functionalities to genes or when the conditions of a microarray are diverse (Wang et al (2002)). Thus, clustering should be performed on the two dimensions (genes and conditions) simultaneously. Also, many genes may be grouped into diverse clusters (or none of them) depending on their participation in different biological processes within the cell (Gasch & Eisen (2002)). These characteristics are covered by biclustering techniques, which have also been largely applied to microarray data [Madeira & Oliveira (2004); Tanay et al (2005); Busygin et al (2008)]. The groups of genes and samples found by biclustering approaches are called biclusters.

Finding significant biclusters in a microarray is a much more complex problem than clustering (Divina & Aguilar-Ruiz (2006)). In fact, it has been proven to be a NP-hard problem (Tanay et al (2002)). Consequently, the

majority of the proposed techniques are based on optimization procedures as the search heuristic. The development of both a suitable heuristic and a good cost function for guiding the search is essential for discovering interesting biclusters in an expression matrix. In order to design an effective evaluation measure for biclusters, we have focused our research on the study of the different types of expression patterns in the literature, being the most general situation given by the so-called shifting and scaling pattern, which combines both the additive and multiplicative models (Aguilar-Ruiz (2005)). This combined pattern summarises in a mathematical formula the different type of bicluster formulations in the literature. This way, every expression value in a bicluster would be the result of applying both multiplicative and additive factors per sample to a characteristic value of each gene. In section 4.2 we give a formal definition of a shifting and scaling pattern for gene expression data.

In this PhD Thesis we present an efficient evaluation measure for biclusters based on the concept of behavioural patterns, as well as an evolutionary heuristic with combines several objectives for the search of biclusters in gene expression microarray data. The main novelty of our approach regarding the existing ones is its capability of adjusting the system behaviour depending on desired characteristics of the results.

1.2 Hypotheses

Our work has been mainly based on three hypotheses, regarding two different existing problems within biclustering of gene expression data.

First hypothesis: *It is possible to develop effective quality measures for biclusters capable of capturing both shifting and scaling patterns simultaneously.*

This hypothesis covers the problem that supposes the lack of a quality metric for biclusters able to recognize all the possible kind of patterns in gene expression data. In order to fully comprehend this problem we first drew up an in-depth description of all the possible patterns in gene expression data, according to the literature. Afterwards, we studied different possibilities of grouping all these patterns into a single mathematical formula or procedure, consisting thus as our start point for developing a quality measure for biclusters.

In order to conduct our research for finding a both effective and efficient quality metric for biclusters based on the patterns concepts, we started studying the equation for shifting and scaling combined pattern presented in a previous work of Aguilar-Ruiz (2005). We also analysed some biclusters obtained by several authors (Cheng & Church (2000); Bleuler et al (2004); Divina & Aguilar-Ruiz (2006)), observing that the range of expression values assumed by genes can vary substantially depending on the specific microarray. Therefore, in order to do an appropriate comparison between each gene and the pattern, it would be desirable to define a mechanism for moving the expression levels to a common range. This mechanism would also be responsible for softening every gene behaviour, since the most important aspect is to characterize their tendency rather than their numerical values. For these reasons, we focused our research in the application of a standardization based procedure in order to assess the similarity of the data in a bicluster to its closest perfect combined pattern.

The use of an appropriate bicluster evaluation measure together with a search heuristic made us think in a second hypothesis:

Second hypothesis: *The use of a quality measure with the characterictics presented in the first hypothesis within an evolutionary strategy would guide the search towards significant biclusters.*

This second hypothesis states that the combination of a good metric together with an evolutionary heuristic would produce quality biclusters. Once the metric to be used in the procedure has been defined, the problem

consists in finding the best configuration of an evolutionary based algorithm for the search of biclusters in a microarray. The main reason for focusing our research on evolutionary computation is the fact that biclustering has been proven to be a NP-hard problem by Tanay et al (2002). Evolutionary algorithms suit very well to this kind of problems, where a group of candidate solutions are iteratively improved with regard to a given measure of quality (fitness function) (Floreano & Mattiussi (2008)).

Third hypothesis: *It is possible to desing an effective parametrization of an evolutionary biclustering algorithm which would guide the search towards biclusters with certain features specified by the user.*

Evolutionary environments have already been used in biclustering, due to its appropriateness to the problem, where populations of potential solutions allow the exploration of a greater portion of the search space. In section 4.4 we review the most important evolutionary-based biclustering approaches. Nevertheless, none of them allow the user to choose the objectives involved in the search. In this sense, this hypothesis made us work towards the developing of a customizable evolutionary biclustering approach, where different objectives might be weighted by the user depending on the desired results. This objectives include volume, gene variance or overlapping level among biclusters. Furthermore, it would also be very easy for the user to incorporate new objectives into the search.

1.3 Summary of Contributions

The main contributions of this PhD Thesis correspond to the problems summarized in the hypotheses: the need for an efficient evaluation measure for biclusters capable of capturing shifting and scaling patterns simultaneously, and the inclusion of a metric with these characteristic in an evolutionary environment.

The research in this PhD Thesis has been supported by grants TIN2004-00159 and TIN2007-68084-C02-00 from the Ministry of Science and Tehnology, and also by grant TIN2011-28956-C02 from the Spanish Ministry of Economy and Competitiveness, which is currently an ongoing project.

During the development of this PhD Thesis work, our first tasks were oriented towards the design of an effective bicluster evaluation measure. This way, *Maximal Standard Area* (MSA) and *Virtual Error* (VE) were our first approximations, being both of them based on standardization procedures. Finally, applying matrix transpositions before computing VE we came up to *Transposed Virtual Error* (VE^t), which has been proven to identify shifting

and scaling patterns simultaneously in biclusters. These works have been backed up by the publication of several conference and journal papers, in which these evaluation measures were included in other author heuristics.

After having developed VE^t, we focused our efforts upon the design of a fully customizable evolutionary approach for the search of biclusters in gene expression microarrays named Evo-Bexpa, from ***Evo****lutionary* ***B****iclustering based on* ***Ex****pression* ***Pa****tterns*, using VE^t as the main objective. Furthermore, Evo-Bexpa includes a guide on how the algorithm can be guided towards different kind of solutions, by using configuring parameters. Through the manipulation of several input parameters, the user can change the system behaviour, producing thus biclusters with different volumes, overlapping levels or gene variances, among others. It is even possible to add new user-defined objectives to the search, if available.

Our contributions can therefore be enumerated as a list of three different metrics for the evaluation of biclusters and a biclustering heuristic based on evolutionary computation, configurable with parameters. Furthermore, other related works to this PhD Thesis include a study on two overlapping control mechanisms for biclusters and a parallel work on clustering over protein interaction networks, a piece of research carried out during the leave of absence required to obtain the European doctoral dissertation.

We summarize in the following the list of conference and journal publications mentioned above:

- **Evaluation measures for biclusters based on the concept of expression patterns:**
  - **Maximal Standard Area (MSA).** This evaluation measure was published in the proceedings of the IEEE International Fuzzy Systems Conference of 2007 (Giraldez et al (2007)), and also in the *II Congreso Español de Informática* (CEDI) (Pontes et al (2007c)).
  - **Virtual Error (VE).** VE was first published in the international journal *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics* (Pontes et al (2007a)) and also in the national workshop *IV Taller Nacional de Minería de Datos y Aprendizaje* (TAMIDA) Pontes et al (2007b)), within the *II Congreso Español de Informática* (CEDI) national conference. A further study on VE and its inclusion into a multi-objective approach has been finally published in the *Computers in Biology and Medicine* international journal (Divina et al (2012)). This work contains an in-depth comparative of VE and MSR with seven different real datasets, also including biological validation of the results. This

validation proves that significant results are obtained, covering thus the second hypothesis.

- **Transposed Virtual Error (VE^t).** VE^t was originally presented in the international journal *Pattern Recognition in Bioinformatics* (Pontes et al (2010a)), together with an experimental comparison in which two other evaluation measures were included. It has also been published in the *VII Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados* national conference (Pontes et al (2010b)). The definition of VE^t in these works covers our first stated hypothesis.

- **Customizable evolutionary biclustering algorithm:**

  - **Evolutionary Biclustering based on Expression Patterns (Evo-Bexpa).** This work present our final contribution to this PhD Thesis, since it makes use of VE^t at the same time as presents a fully customizable evolutionary biclustering algorithm, which covers our third hypothesis. Biological validation is also including, which also proves the second hypothesis. A preliminary version of this work was presented in the 11th international conference on *Intelligent Systems Design and Applications* (ISDA) (Pontes et al (2011)). The complete work presented in chapter 7 has been sent to the international journal *Algorithms for Molecular Biology* and has already been accepted for publication.

- **Other related works:**

  - **Improved Biclustering on Expression Data through Overlapping Control.** In this work we present a overlapping control strategy for biclustering based on the use of a matrix of weights. This strategy has been compared with the random replacement in a greedy approach. A discussion on this topic together with the obtained results have been published in the eight IEEE international conference on *Hybrid Intelligent Systems* (Pontes et al (2008)) and also in the international journal of *Intelligent Computing and Cybernetics* (Pontes et al (2009)).
  - **Describing the Orthology Signal in a PPI Network.** This work represents the results of a collaboration with professor Elena Marchiori and her research group at the *Department of Information and Knowledge Systems* (IRIS), in which we proposed a

clustering-based methodology for protein-protein interaction networks. This work has been published in the international journal of *Bioinformatics Research and Applications* (Jancura et al (2011)).

1.4 Document Structure

The contents of this PhD Thesis dissertation are organised into four different parts, as follows:

- **Part I: Preamble.**
  - **Chapter 1: Introduction.** This chapter corresponds to the current one, in which the motivation, hypothesis and a summary of the contributions of this PhD Thesis have already been presented.
  - **Chapter 2: Bioinformatics Background.** An overview on bioinformatics background is presented in this chapter. It starts by describing some basic biological concepts needed to understand the main research fields on bioinformatics, which are exposed afterwards. Finally, a summary on the most relevant data-mining techniques used in this research field is also presented.
- **Part II: Foundations.** An study of the state of the art on the central topics of this PhD Thesis is presented in this part. It has been divided into three different chapters in the following way:
  - **Chapter 2: Microarray: Technology and Analysis.** This chapter exposes microarray technology and its applications, giving an special attention to gene expression microarrays. Regarding microarray analysis, the focus has been put upon high-level analysis based on data-mining techniques. Biological databases for verification and interpretation of microarrays analyses output have also been overviewed.
  - **Chapter 4: Biclustering Algorithms.** An extensive survey on biclustering algorithms for gene expression data is presented in this chapter. Focus has been put on gene expression pattern, describing the different kind of bicluster patterns in the literature and also the most relevant bicluster evaluation measures capable of identifying them. Biclustering approaches have been divided into two categories, depending on being based on any bicluster evaluation metric or not. Finally, the most common biological

validation strategy for biclusters has also been introduced in this chapter.

- **Chapter 5: Evolutionary Computation.** In this chapter the basic notions of *Evolutionary Computation* (EC) are given. We begin by describing the common features of EC, individuating afterwards four different paradigms. Three distinct evolution strategies are also presented, as well as two advanced evolutionary algorithms, the multi-objective and memetic approaches.

- **Part III: Proposals.**

  - **Chapter 6: Standardization-based Evaluation Measures.** This chapter gives formal definitions of our proposed evaluation measures for biclusters. Since we have based all of them on a previous bicluster standardization procedure, we begin the chapter by defining the standardization procedure applied. Afterwards, we describe three different evaluation metrics for biclusters, the latest of which (VE^t) has been proven to be effective for capturing the combined patterns.

  - **Chapter 7: Evolutionary Biclustering based on Expression Patterns.** We present in this chapter a fully customizable evolutionary biclustering algorithm named Evo-Bexpa. VE^t has been used as the bicluster coherence evaluation measure, together with other objectives such as the bicluster volume, gene variance or overlapping level. The algorithm can be easily configurable towards obtaining results with the desired characteristics, according to the user preferences. Furthermore, new user-defined objectives can also be incorporated into the search without any difficulties, which makes our algorithm fully customizable. Experiments on both synthetic and real datasets have also been conducted, demonstrating Evo-Bexpa abilities to obtain meaningful biclusters.

  - **Chapter 8: Conclusions.** This chapter summarizes the main conclusions of the work carried out during the development of this PhD Thesis. Furthermore, future works that we think will be a very interesting continuation of the ones carried out are also described.

- **Part IV: Appendices.** This part includes a related work carried out during the development of this PhD Thesis, and which has already been commented in the former section.

– **Appendix A: Improved Biclustering on Expression Data through Overlapping Control.** This work presents a study on two different overlapping among biclusters control strategies: random replacement and by using a matrix of weights. We propose a modified version of a very popular bicluster algorithm that improves the original one by using a weight of matrix for the overlapping control.

Chapter 2

Bioinformatics Background

Bioinformatics is a relatively young multidisciplinary field, consisting of the application of computer science techniques to solve diverse problems in biology. In spite of being a young area of study, its pace of research is rapidly increasing due to the large amounts of complex data generated by high-throughput technologies in laboratories. The scope of bioinformatics research moves beyond the study of individual biological components (genes or proteins), allowing also the study of how different individual parts cooperate in a more general scenario. In this context, systems biology integrates different kind of biological data together, building up system models, seeking for the interpretation of biological complexes (Lesk (2008)).

Due to the great complexity inherent to biological data, and also to the enormous amount of information available, the application of computer science technologies is essential for the discovery of meaningful knowledge. The ultimate goal of bioinformatics is the design and development of new algorithms and tools which help in the interpretation and analysis of many types of biological data (Cohen (2004)). In fact, there exist a great variety of areas in molecular biology in which computer science methodologies can be applied in order to extract useful knowledge from raw data, including assembly of sequence fragments, DNA or RNA analysis, prediction and analysis of protein sequence and function, or metabolic function and regulation analysis and simulation. Among the different computational techniques that are currently being applied to discover relevant knowledge from biological data it is important to cite modelling, simulation, data abstraction and manipulation or pattern discovery.

In this chapter we summarise the main fields of research and resources within bioinformatics, paying special attention to those related with this PhD Thesis work, detailing first the biological theories and concepts necessary for the understanding of the subsequent chapters.

2.1 The Principles of Life

Life can be defined using several common characteristics of living beings. These characteristics may be summarized in the following three:

- All living beings are made up of *cells*, and every cell is made up of the same types of molecules. These molecules intervene in the formation of different cellular components, in metabolic reactions or in the conservation and transmission of genetic information.

- Multicellular organisms have different levels of organization, ranging from simplest (cells) to most complex (organisms). Individual cells may perform specific functions and also work together for the good of the entire organism, becoming thus dependent on one another.

- There are three basic or vital functions for every living being: *nutrition*, *relation* and *reproduction*. Nutrition consist of constantly taking nutrients to transform them into energy and organic compounds. Relations means to interact with other organisms and the environment by signal reception and interchanging and stimulus responses. Finally, by means of reproduction, living organisms produce offspring, either genetically identical (asexual reproduction) or by genetic interchange among different sexual individuals (sexual reproduction).

Although the term *cell* was first introduced in 1665 by Robert Hooke, who observed stark cells by means of a very rudimentary microscope, it was during the XIX century when the study of the cell was propitiated by microscopy technological advances. Cell theory refers to the idea as cells are the basic unit of structure in every living being, from the most simplest (micro-organisms) to most complex organisms like animals and plants. Cell theory may be summarised in the following central ideas:

- All living being are made up of one or more cells.

- The cell is the fundamental unit of structure and function in all living organisms.

- All cells arise from pre-existing cells by division. The new cell is identical to the one from which it is preceded.

- The cell is the more elemental independent unit of life.

Nevertheless, in a more specific level, every living thing is made up of biomolecules, which are also made up of biogenic elements, such as carbon, oxygen, hydrogen, nitrogen, phosphorous and sulphur. Biomolecules can be categorized into *inorganic biomolecules*, such as water and mineral salts, or *organic biomolecules*, which are exclusive for living beings. Organic biomolecules can be classified in four different groups: *carbohydrates*, *lipids*, *proteins* and *nucleic acids*.

Carbohydrates are made up of *carbon*, *oxygen* and *hydrogen atoms*, and they are used for energy extraction. They can also be divided into *monosaccharides*, which constitute a direct source of energy, and *polysaccharides*, composed of long chains of monosaccharide units bound together,and are mainly used as energy reserves. Lipids constitute a quite heterogeneous group, though they share several properties, such as water insolubility. Among their multiple functions it is important to cite energetic, structural or regulatory. Proteins are macromolecules built up using 20 different amino acids, the order in which they are placed determines the specific protein, according to the specie. Proteins are very specific, since different organisms have different proteins, determining thus organisms biological identity. Protein functions include structural, hormonal, transport, immunity, contractile or homoeostatic. Many of them also acts as enzymes, favouring chemical reactions. Finally, nucleic acids are mainly made up of *carbon*, *hydrogen*, *oxygen*, *nitrogen* and *phosphorous*. They are responsible for every basic function of living beings, since they have the information for carrying out every vital process. There exist two different kind of nucleic acids, *deoxyribonucleic* and *ribonucleic acids*, which we will discuss later.

2.1.1 Structural Cell Biology

As aforementioned, cells constitute the morphological, structural and functional units of every living being. Despite the great variety of existing cells, all of them share common structural and functional characteristics:

- They present a cell *membrane* or plasma membrane that separates the interior of all cells from the outside environment. This membrane is able to regulate what enters and exits the cell, thus facilitating the transport of materials needed for survival.
- Inside the cell there exists a gel-like substance called the *cytoplasm*, consisting of a colloidal dissolution of biomolecules.
- Within the cytoplasm and the nucleus the biochemical reactions needed for life are carried out.

- More evolved cells (eukaryotes) have *organelles* in their cytoplasm, in charge of the realization of specific functions.
- All cells hold nucleic acids molecules, containing the genetic material with the information for the regulation and coordination of all cell activities.

Although these characteristics are shared by all cells in all organisms, there exists many different types of cells having different functions, and therefore their structures are also different. For example, neurons are built and work differently than muscle cells. This way, cell diversity refers to the wide variety of cells and the differences between various cells, even in the same organism.

Attending to their structural complexity it is possible to differentiate two kind of cells, *eukaryotes* and *prokaryotes*, depending on the existence or absence of an inner nucleus in the cytoplasm. There exists also other several differences.

Prokaryotic Cells

Prokaryotic cells correspond to those organisms that do not have a membrane-bound nucleus or membrane-bound organelles. They are characteristic and exclusive of bacteria, and are usually small, the size of any eukaryotes organelles.

Figure 2.1 shows the structure of a typical prokaryote. Apart from plasma membrane and cytoplasm, in the figure it is possible to differentiate the following elements:

- **Cell wall,** which surrounds the plasma membrane. It protects the bacterial cell and gives it shape.
- **Circular DNA** or nucleoid region. This is the area of the cytoplasm that contains the DNA molecule.
- **Other elements.** Several groups of bacteria also contain:
  - Flagelum. Long protrusion that aids in cellular locomotion.
  - Pilus. Hair-like structures on the surface of the cell that attach to other bacterial cells.
  - Capsule. Additional outer covering that protects the cell when it is engulfed by other organisms. It also helps the cell to adhere to surfaces and nutrients.

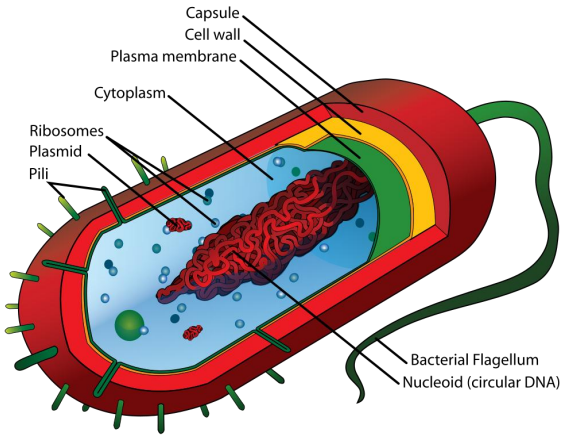


Figure 2.1: Structure of a prokaryotic cell

– Internal membrane systems, such as ribosomes, responsible for protein production.

Eukaryotic Cells

Eukaryotic cells correspond to those organisms whose cells contain a membrane-bound nucleus and other membrane-bound organelles. The inclusion of organulles in cells allow them to carry out more functions than prokaryotes can. Excluding bacteria, every living being present an eukaryotic organization of their cell or cells. For animal cells, the cell surface consists of the plasma membrane only, but plant cells have an additional layer called cell wall, which is made up of cellulose and other polymers.

Figure 2.2 shows the structure of a typical eukaryotic cell, where it is possible to differentiate the following characteristic elements:

- The **cytoskeleton** is a network of microtubules, corresponding to the skeleton within the cell. Its functions include maintaining cells shape, protecting the cell, enabling cell motion (using flagella) and intracellular transport.

- **Ribosomes** are cellular structures involved in making proteins under the instruction of DNA. They are found attached to the rough endoplasmic reticulum or floating free in the cytoplasm.

- **Mitochondria and chloroplasts** are responsible for the energy production by breathing and photosynthesis processes. Chloroplasts are

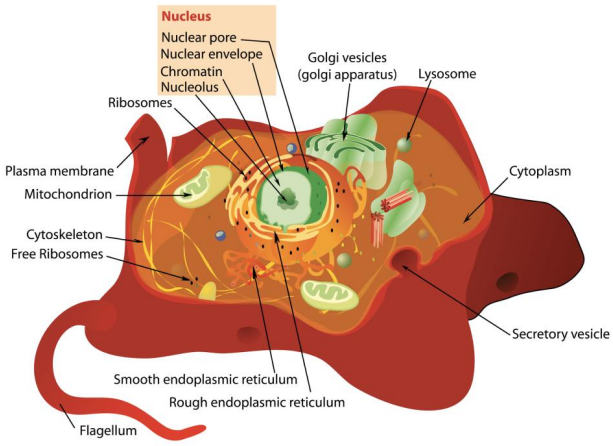


Figure 2.2: Structure of an eukaryotic cell

possessed only by plants, while mitochondria are typical of animal cells.

- **Endomembrane System,** consisting of a set of cellular organelles working together for the manufacturing and transportation of materials into, out of, and within the cell. It is made of the nuclear envelope, rough and smooth endoplasmatic reticulum, the Golgi apparatus and vesicles.

- Other membranous organelles within the cell are the **lysosomes**, containing enzymes for materials digestion, and **vacuoles**, helping with the transportation and storage of water and other materials.

- **Nucleus.** A double membrane-bound control center separating the genetic material, DNA (deoxyribonucleic acid), from the rest of the cell. Inside the nucleus is a fluid called **nucleoplasm**, a **nucleolus**, and linear chromosomes forming structures known as **nucleosomes**. The nucleolus is an area within the nucleus that is involved in the assembly of ribosomal subunits, while the nucleosomes are part of what is called chromatin, the DNA and proteins that make up the chromosomes.

2.1.2 Molecular Genetics

Molecular genetics is the field of biology and genetics that studies the structure and function of genes at a molecular level. In this section we summarise the main concepts needed to understand gene expression and its regulation.

Nucleic acids are the carriers of all biologic information, specifying both physiological and morphological characteristics of every living being. Deoxyribonucleic acid (DNA) passes this information from one generation to the next (although some viruses use ribonucleic acid (RNA) for this purpose). In order to distribute the information in the DNA, a process of replication is carried out, in which a series of copies are distributed to the offspring cells, sometimes including several variation or mutations.

DNA and RNA consist of a nucleotide sequence that are interpreted leading to specific characteristics at different levels of every organism. An individual nucleotide itself has 3 components: a *sugar*, a *phosphate* and a *base*. Nucleotides are hooked together to make a long sugar-phosphate backbone with a sequence of bases sticking off. The sugar may be one of two kinds, which distinguish DNA (the sugar is deoxyribose) from RNA (the sugar is ribose). There are 5 kinds of bases, disposes in different ways in the nucleotides sequences of both DNA and RNA. The bases in DNA are *adenine* (A), *cytosine* (C), *guanine* (G) and *thymine* (T), while RNA has in place of T another base called *uracil* (U). By studying the DNA, it was found to have the shape of a helix with two strands, where the number of A's was the same as the number of T's. Similarly, the number of C's was equal to the number of G's. Adenine was observed to fit together with thymine, while guanine paired with cytosine. In the case of RNA, uracil is complementary to adenine.

The principle of complementarity is the key to replication, as well as to the genes second main function: making proteins. A gene was defined as a fragment of nucleic acid containing the information for a specific characteristic. The place within de DNA in which each gene is located is referred to as *locus*. The most important development in the understanding of the relationship between genes and proteins was the discovery of the structure of DNA by Watson & Crick in 1953 and the sequence hypothesis (Crick 1958), which states that a simple congruence exists between the one-dimensional nucleotide sequence of a gene and the one-dimensional amino acid sequence of a protein. This hypothesis makes two predictions: 1) There will be a linear correspondence between the gene and the protein it determines (collinearity). 2) Each amino acid will be specified by a given set of nucleotides. After several experiments, collinearity was established to be a correspondence between three nucleotides for each amino acid, although some amino acids correspond to different triplets of nucleotides.

The mechanism by which a sequence of amino acids is originated from a sequence of nucleotides can be divided in two different steps: *transcription* and *translation*. Transcription occurs in the nucleus (eukaryotes) or nucleoid (prokaryotes) and is the process of creating a complementary mRNA copy of a

sequence of nucleotides of a gen (DNA). Translation occurs in the ribosomes, where messenger RNA (mRNA) produced by transcription is decoded to produce a specific amino acid chain, that will later fold into an active protein. This process constitutes the central dogma of molecular biology and is also called *gene expression*.

If cells were constantly synthesizing every kind of proteins for which they have information, metabolic chaos would occur. Consequently, there may exists a gene expression regulation system, which will control the synthesis of proteins depending on the special needs at a certain period of life time. These needs also depend on both intra and extra cellular variations. Furthermore, in multicellular organisms gene regulation drives the processes of cellular differentiation and morphogenesis, leading to the creation of different cell types that possess different gene expression profiles though they all possess the same genome sequence. Since the amount of synthesized proteins depends on the amount of mRNA in the cell, gene expression may be regulated by regulating the production of mRNA. Therefore, the majority of gene expression regulation process are based on the regulation of mRNA production.

2.2 Bioinformatics Research Fields

The amount and variety of biological data now available, together with techniques developed so far have enabled research in bioinformatics to move in different biological scenarios. The areas where computer science can be applied range from assembly of sequence fragments, analysis of DNA, RNA and protein sequences, prediction and analysis of protein sequence and function, and the analysis and simulation of general metabolic function and regulation. Most relevant and studied topics in bioinformatics are presented in the next subsections, although the specified categories are not exclusive, since some biological problems may be included in more than one subdivision. For example, the problem of assigning new functionalities to proteins corresponds to *Protein Function Analysis* category but is also related to *Sequence Analysis*.

2.2.1 Sequence Analysis and Comparison

Within the cell, DNA is transcribed into RNA and then translated into proteins. This means millions of molecular sequences that present both great opportunities and challenges. Analysing and comparing these kind of sequences allow the scientists to understand the biology of the organisms at many different levels. For example, sequence analysis and comparison is used

to find new genes, determine their functions, study their regulation or assess their evolution over time, among others.

The development of methods of high-throughput production of gene and protein sequences has exponentially increased the rate of addition of new sequences to the databases. Nevertheless, this information by itself does not provide any useful knowledge. Applying computer technologies to analyse this kind of data is a key factor to generate new insights towards their understanding. Comparison between two or more sequences is usually carried out using *alignment* procedures, where the sequences are lined up to achieve a maximal level of identity. This process is called pair-wise alignment for the comparison of two sequences or multiple alignment if more than two sequences are used.

Sequence analysis includes a wide range of relevant topics, including homology search and identification of variations:

Homology search.

Sequences are called *homologous* if they have significant similarity and evolved from a common ancestral sequence, but it is not always easy to determine. The degree of similarity obtained by two sequences alignment can be useful in determining the possibility of homology between to sequences. The applications of homology searches include function and conserved regions determination. Ping et al (2007) presented a review of many different algorithms for homology searches.

Identification of sequence differences and variations.

Sequence difference variations include point mutations and *Single Nucleotide Polymorphisms* (SNPs). The genome is full of single nucleotide differences, called polymorphisms, or SNPs. The majority of SNPs are meaningless, however, some of these genetic differences have proven to be very important in the study of human health, for which then are being extensively studied. They help in the prediction of individuals responses to certain drugs, susceptibility to environmental factors such as toxins, and risk of developing particular diseases, and can also be used to track the inheritance of disease genes within families. The *National Center for Biotechnology Information* (NCBI) established the dbSNP database (Sherry et al (2001)), now it has been incorporated into NCBI's Entrez system and can be queried using the same approach as the other Entrez databases such as PubMed and GenBank. Johnson (2009) presented a review on different SNPs resources, dbSNP included among them.

2.2.2 Comparative Genomics

A genome is the collection of DNA that comprises an organism. Each individual genome contains the genes and other DNA elements that define its identity. Genomes range in size from the smallest viruses, which encode fewer than 10 genes, to eukaryotes such as humans that have billions of base pairs of DNA encoding tens of thousands of genes (Pevsner (2009)).

A genome does not capture the genetic diversity or the genetic polymorphism of a species. To learn what variations in genetic information underlie particular traits or diseases requires comparisons across individuals. Comparative genomics is the analysis and comparison of genomes from different species. The purpose is to gain a better understanding of how species have evolved and to determine the function of genes and noncoding regions of the genomes. Genome researchers look at many different features when comparing genomes: sequence similarity, gene location, the length and number of coding regions (exons) within genes, the amount of noncoding DNA in each genome, and highly conserved regions maintained in organisms as simple as bacteria and as complex as humans. For these purposes, the availability of computer programs capable of lining up multiple genomes is essential. One of the most widely used tools is BLAST ([1]), which comprises a set of programs for performing similarity searches on sequence data.

Bacteriophage φX174 was the first genome to be sequenced in 1977, a viral genome with only 5,368 base pairs (bp) (Sanger et al (1977)). Nevertheless, the first genome of a free-living organism was not sequenced until 17 years ago, in 1995. In this case the genome corresponded to the bacterium Haemophilus influenzae Rd, with 1,830,137 base pairs (Fleischmann et al (1995)).

In 1990, the *Human Genome Project* (HGP) [2] was organized to map and to sequence the human genome. The project was coordinated by the U.S. Department of Energy and the National Institutes of Health, and was planned to last 15 years. However, rapid technological advances accelerated the completion date to 2003, although the first draft was published in 2001 (Venter et al (2001)). Human genome is made up of 3,200,000,000 base pairs. Researchers used DNA samples from a number of donors, male and female, because all humans share the same basic set of genes and other DNA regions, this reference sequence represents every person.

The amount of sequence data that are generated continues to accelerate rapidly. For many genomes, even unfinished genomic sequence data are immediately available to the scientific community. There exist a great variate

[1] http://blast.ncbi.nlm.nih.gov/Blast.cgi

[2] http://www.ornl.gov/sci/techresources/Human_Genome/home.shtml

of web resources for the access and study of genomes. Among them, it is interesting to cite the *European Bioinformatics Institute* (EBI) [3], the *National Center for Biotechnology Information* (NCBI) [4], the *Comprehensive Microbial Resource* (CMR) [5], focuses on prokaryotic projects, or the genome browser at the *University of California* [6], putting their emphasis on vertebrate genomes.

The aim of comparative genomics is the use of related genomes to better understand each individual genome in the set, being the most fruitful method of understanding the functional content of genomes studying them in the context of related genomic sequences. Both intra and interspecific sequence comparisons are based on a variety of computational methods, including alignment, phylogenetic reconstruction, and coalescent theory (Haubold & Wiehe (2004)).

Among the multiple applications of comparative genomics in bioinformatics (see Chain et al (2003) for a review) we would like to bring out *Phylogenetics* and *Population Analysis*. We would also pay a special attention to *Next Generation Sequencing* technology, responsible for the production of massive genomic data.

Phylogenetics.

Phylogenetics is the study and identification of evolutionary relatedness species, both living (extant) and dead (extinct). Similarity among individuals or species is attributable to common descent, or inheritance from a common ancestor. The discovery of this kind of relationships is made through the study of the genomes of the different species, and reveals the species evolutionary history. The most convenient way of visually presenting evolutionary relationships is by means of *phylogenetic trees*, as in Figure 2.3, where nodes represent the different species or ancestors and branches define the relationships.

With the rapid accumulation of genomes sequence data, more and more phylogenies are being constructed based upon sequence comparisons. The combination of these phylogenies with powerful new statistical approaches for the analysis of biological evolution is challenging widely held beliefs about the history and evolution of life on Earth. Novel works on phylogenetics include Suchard & Rambaut (2009) one, where the authors describe novel algorithms and methods for evaluating phylogenies under arbitrary molecular

[3] http://www.ebi.ac.uk/genomes/
[4] http://www.ncbi.nlm.nih.gov/sites/genome
[5] http://cmr.jcvi.org/tigr-scripts/CMR/CmrHomePage.cgi
[6] http://genome.ucsc.edu/

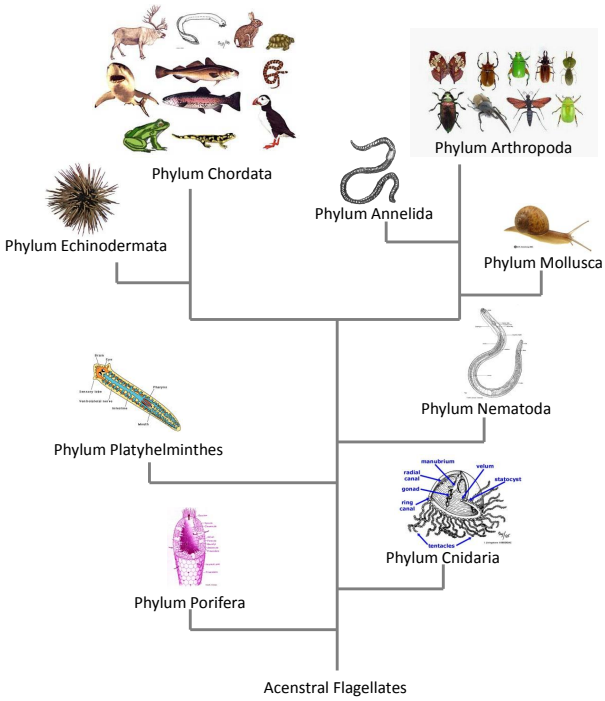


Figure 2.3: Phylogenetic tree

evolutionary models.

Population Analysis.

Population genomics analyzes DNA variations present in populations as a whole and at a whole genome level. The primary goal is to assess a population genetic constitution as well as its change over with time. Such studies are aimed at the detection of genomic variations and evolutionary processes that can influence the frequency of these variations, such as mutation, selection, genetic drift, gene flow, and population structure. A focus on variations and their frequency can shed light on the evolutionary history and structure of these populations. Among the multiple works on this field are included those of Wade et al (2009) on domestic horses or Winsor et al (2011) on pseudomonas.

Next Generation Sequencing (NGS).

Next Generation Sequencing techniques deliver fast, inexpensive and accurate genome information. Several companies, including IBM or LaserGen, among others, have made their investments in this kind of technologies. As a result, the introduction of instruments capable of producing millions of DNA sequence is rapidly changing the landscape of genetics, giving the opportunity to answer questions at a unimaginable speed. In the not too distant future, it is foreseeable that NGS technologies could be used to obtain high-quality sequence data from a genome isolated from a single cell, which would be a substantial breakthrough, particularly for cancer genomics (Metzker (2009)). The applications of NGS technologies are multiple, Mardis (2008) surveys next-generation sequencing technologies and consider how they can provide a more complete picture of how the genome shapes the organism. Reis-Filho (2009) discusses the potential impact of this technology on breast cancer research and the challenges that come with this technological breakthrough. Other reviews on NGS include those of Shendure & Ji (2008) and Metzker (2009). In their work, Magi et al (2010) guide readers in the choice of the available computational tools that can be used to face the several steps of the data analysis workflow.

2.2.3 Gene Expression Analysis

In genetics, gene expression is the most fundamental level at which genotype gives rise to the phenotype. Gene expression occurs when DNA is transcribed into RNA, corresponding to the process by which information from a gene is used in the synthesis of a functional gene product. These products are often proteins, but in non-protein coding genes the product is a functional RNA. The number of genes in a cell differs depending on the organisms, varying from 2,000 to 60,000 for the eukaryotic ones. Although the DNA sequence of all the cell in a certain organism is the same, at any given time each cell expresses only a subset ot those genes, depending on the intricacies of the regulation of gene expression. Gene expression is therefore regulated in several ways: by region, development stage, disease states, gene activity or by dynamic response to environmental signals (Pevsner (2009)).

Measuring gene expression means quantifying the level at which a particular gene is expressed within a cell, tissue or organism. The information obtained by this procedure allow the scientists to understand different process or stages within the cell, such as viral infections, susceptibility to cancer (oncogene expression) or the identification of activated genes during cell cycles or throughout development. In order to draw meaningful inferences from

gene expression data, it is important that each gene is surveyed under several different conditions. Gene expression analysis comprises a great variety of computational technologies that help inferring knowledge from gene expression datasets.

In order to carry out a computational analysis of gene expression data, diverse technologies have been applied for gene expression measurement. Northern blotting, also known as RT-PCR [7], has been used for studying the gene expression of only one transcript at a time. In contrast to this kind of approaches, high throughput techniques have emerged allowing a broad survey of gene expression, offering the possibility of performing comparison of gene expressions between different individuals, tissues or any kind of samples. Among these technologies we would like to pay a special attention to SAGE, microarrays and MPSS.

Serial Analysis of Gene Expression (SAGE).

SAGE was developed in 1995 (Velculescu et al (1995)). It allows the analysis of overall gene expression patterns with digital analysis. Two basic principles underlie the SAGE methodology: 1) a short sequence tag (10 base pairs) within the defined position contains sufficient information to uniquely identify a transcript; 2) the concatenation of tags in a serial fashion allows for an increased efficiency in a sequence-based analysis. SAGE seems to be a better choice for the identification of new genes and alternatively processed transcripts that are unique to a specific cell type and for the analysis of previously uncharacterized organisms (Ye et al (2002)).

Microarray Technologies.

DNA microarrays were invented by Schena et al (1995) in 1995 and constitute an empirically-based methodology to determine which gene(s) are responsible for creating the phenotype change. By putting some microscopic DNA spots on a solid surface, the microarray chips are able to measure at the same time the expression level of an important number of genes for a tissue. In their first years only 48 genes could be measured at one time, whereas now we can measure ten thousands of genes. The use of microarrays increased considerably from 2000, mainly due to the sequencing of the human genome and still are extensively used within bioinformatics (Bucca et al (2009)). The applications of microarray are plenty, mainly related to disease pathology, progression, diagnosis and resistance to treatment. In section 3.1 we present a more in-depth description of microarrays types and applications.

[7] http://www.rtpcr.co.uk/

NGS for Gene Expression Analysis.

NGS platforms can nowadays be used for the majority of the task carried out with microarrays, thus challenging their use. However, several studies have concluded that microarrays and NGS are actually complementary platforms, rather than competitive alternatives, that should be used together to gain the maximum results (Euskirchen et al (2007)). Furthermore, these new methodologies also have their limitations, for example, around 20 per cent of the reads in the human genome cannot be unambiguously mapped to a single location because they occur more than once in the genome (Pop & Salzberg (2008)).

2.2.4 Proteomics

Proteomics refers to the study of proteins, being the main components involved in the chains of molecular interactions in cells (*metabolic pathways*). The sequencing of the human genome has increased interest in proteomics because while DNA sequence information provides a static snapshot of the various ways in which the cell might use its proteins, the life of the cell is a dynamic process. Analysing protein data would reverberate through different proteomic applications in science, medicine, and also pharmaceuticals. In fact, the identification of potential new drugs for diseases treatments constitutes one of the most important applications of the study of proteins.

The word *proteome* comes from protein and genome, corresponding to the set of expressed proteins in a given type of cells or an organism at a given time under defined conditions. Therefore, proteome will vary with time and distinct requirements or stresses that a cell or organism undergoes. Bioinformatic research on proteins can be grouped in three different although related aspects of proteins: protein *structure*, *function* and *interactions*.

Structural Bioinformatics

Structural bioinformatics is the branch of bioinformatics related to the analysis and prediction of the three-dimensional structure of biological macromolecules. The physical properties of the structure of proteins in particular is crucial in understanding molecular moves and interactions, providing the details needed to understand complex molecular interactions in fields such as immunology and systems biology (Gu & Bourne (2009)).

Protein *folding* is the process by which a protein folds into its characteristic and functional three-dimensional structure from its amino acids sequence. Figure 2.4 shows an example of the different levels of structure of a protein.

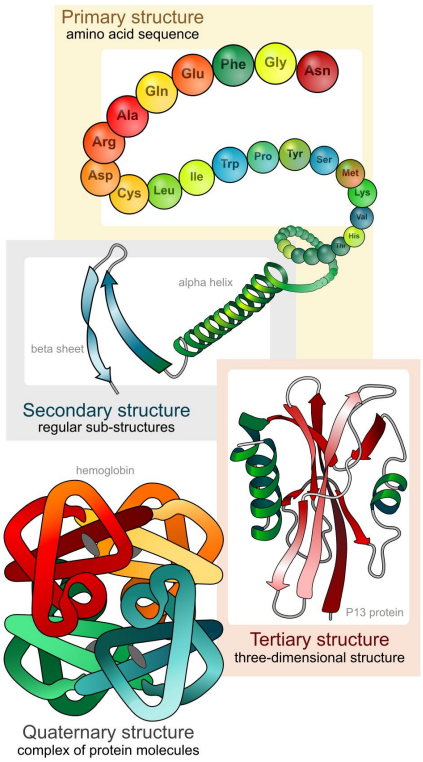


Figure 2.4: Structure levels of proteins molecules

The study of the physical properties and configurations of proteins entails several difficulties, for instance, for some time it had been thought that a gene was responsible for producing a single protein, while today it has been proved that a gene may generate several proteins. Furthermore, similarity at one level cannot be generalized to another, for example, similar 3D substructures may originate from different sequences of amino acids, or vice-versa. In addition, it is also known that simple changes of nucleotides in DNA may result in entirely different protein structures and function. All these adversities together makes the study of protein structures a very challenging and promising field of research. In this context, Kota et al (2011) have recently developed Gaia [8], a tool for evaluating the packing and covalent geometry of a given protein structure and provides quantitative comparison of the given structure to high-resolution crystal structures. Hassanzadeh (2009) presents a survey on classification of proteins based on their structure.

[8] http://chiron.dokhlab.org

Protein Function

Protein function is defined as the role of a protein in a cell (Jacq (2001)). Each protein is a gene product that interacts with the cellular environment in some way to promote the cell growth and function. The different protein functionalities can be divided in *antibodies*, *contractile*, *enzimatic*, *hormonal*, *structural*, *storage* or *transport*. The structure of a protein determines its function, for example, collagen has a super-coiled helical shape, which is long, stringy, strong, and resembles a rope. This structure is great for providing support. Hemoglobin, on the other hand, is a globular protein that is folded and compact; its spherical shape is useful for manoeuvring through blood vessels.

Protein function is predicted based on several criteria, such as sequence similarities, homology or structure. Nevertheless, it is important to beware of biological paradoxes, for example, a certain protein structure may correspond to many different functions, while one function might correspond to many protein structures (Orengo et al (2003)). Hawkins & Kihara (2007) have categorized several approaches beyond traditional sequence similarity that utilize large amounts of available data for computational function prediction, including structure, association, interaction, process, and proteomics-experiment-based methods.

Protein Interaction Networks

Protein–protein interactions form the basis for a vast majority of cellular events. It has been discovered that most of the proteins interact with multiple partners and thousands of different proteins from interaction networks or highly regulated pathways. Thanks to high throughput experimental data, researches have begun to uncover general rules obeyed by protein-protein interaction networks, principles of their evolution, and the means of their functioning (Panchenko & Przytycka (2008)). Protein-protein interaction maps provide a valuable framework for a better understanding of the functional organization of the proteome.

Stelzl et al (2005) built up a network in which they discovered up to 3186 novel interactions of human proteins. They also searched the network for interactions linking uncharacterised gene products and human disease proteins to regulatory cellular pathways. Raman (2010) discusses some of the important computational methods for the prediction of functional linkages between proteins, giving also an overview of some of the databases and tools that are useful for a study of protein-protein interactions.

Data from protein interaction networks have also been integrated with

microarray data for different purposes. For example, in their work, Devaux et al (2010) designed an approach for identifying new prognostic biomarkers in myocardial infarction patients based on this data integration.

In this context, a related work has been carried out during the leave of absence required to obtain the European doctoral dissertation. Two visits of a total period of three months have been made to Radboud University in Nijmegen (The Netherlands), where we have been working with professor Elena Marchiori and her research group at the Department of Information and Knowledge Systems (IRIS). At the end of the leave we proposed a novel methodology for quantifying the functionality of the orthology signal in a PPI network at a functional, complex level. The methodology performs a differential analysis between the functions of those complexes detected by clustering a PPI network using only proteins with orthologs in another given species, and the functions of complexes detected using the entire network or sub-networks generated by random sampling of proteins. We applied the proposed methodology to a Yeast PPI network using orthology information from a number of different organisms. The results indicated that the proposed method is capable to isolate functional categories that can be clearly attributed to the presence of an evolutionary (orthology) signal and quantify their distribution at a fine-grained protein level. Our work was finally published in the International Journal of Bioinformatics Research and Applications (Jancura et al (2011)).

2.2.5 Systems Biology

Systems biology investigates the behaviour and relationships of all of the elements in a particular biological system while it is functioning, instead of studying individual genes or proteins individually. By studying the relationships and interactions between various parts of a biological system (genes, mRNAs, proteins, etc) it is hoped that an understandable model of the whole system can be developed. The development of systems biology has been driven by a number of recent advances in the ability to perturb biological systems systematically, such as genetic manipulation. Ideker et al (2001) describes the emergence of systems biology, as are several examples of specific systems approaches.

In this context, SBGN has recently been presented as a *Graphical Notation for Systems Biology*, a visual language developed by a community of biochemists, modelers and computer scientists, consisting of process diagram, entity relationship diagram and activity flow diagram. This language allows the representation, storage and exchange of different kinds of biological information, including gene regulation, metabolism and cellular signalling

(Le Novère et al (2009)).

Systems biology has also been applied to ecology (Evans (2012)), in order to understand the nature of the world change and to make predictions about the way in which it might affect systems of interest. In order to address questions about the impact of environmental change, and to understand the actions that might be taken to ameliorate it, ecologists need to develop the ability to project models into novel, future conditions. This will require the development of models based on understanding the processes that result in a system behaving the way it does, rather than relying on a description of the system.

Evolutionary Systems Biology

Evolutionary Systems Biology (Koonin & Wolf (2006)) involves integrating systems biology modelling, microbial laboratory evolution experiments and large-scale mutational analyses in order to quantify the evolution of biological systems. This has become now possible by the recent availability of the necessary computational tools and experimental techniques. Many works are being currently carried out in this area, Papp et al (2011) reviews recent progresses in mapping evolutionary trajectories and discusses the degree to which these predictions are realistic. Loewe (2009) propose a multilayered mechanistic framework for evolutionary systems biology that centres on fitness, the adaptive landscape and the quantitative modelling of evolutionary processes, and combines knowledge about well-known biological systems from several disciplines.

Research on evolutionary systems biology will benefit both evolutionary theory and current systems biology, understanding robustness by analysing distributions of mutational effects and epistasis is pivotal for drug design, cancer research, responsible genetic engineering in synthetic biology and many other practical applications.

2.2.6 Other Related Research Fields

In this section we point out other research fields born out of the application of other disciplines to bioinformatics, such as text mining or image analysis.

Text Mining for Bioinformatics

Text mining is the technology which discovers patterns and trends semi-automatically from huge collections of unstructured text. It is based on several technologies, such as *natural language processing*, *information retrieval*,

information extraction and *data mining* (Uramoto et al (2004)). Lots of work is being currently done in this field, mainly due to the exponential increase of the biological knowledge in the literature, where the ultimate goal is to find the useful and needed information from a great variety of resources (full texts, patients records, annotations in data bases, etc). Furthermore, on-line biological information exists in a combination of different forms, including structured, semi-structured and unstructured forms, which makes the use of computational techniques essential for this task. A review on different text mining approaches for biological data has been written by Qi & Zhang (2009).

Some applications based on the integration of different approaches include those of Malik et al (2006), who show that by combining different text mining algorithms and their outcome, the results improve significantly. They also propose a system named CONAN which integrates different processes and biological data, such as tagging of gene/protein names, finding interaction and mutation data, tagging of biological concepts and linking to MeSH and Gene Ontology terms. More recently, Papanikolaou et al (2011) presented an application that offers an array of visualization tools for efficient navigation among biomedical records, and concept extraction. Their approach is named bioTextQuest [9] and combines automated discovery of significant terms in article clusters with structured knowledge annotation.

BioImage Informatics

The last two decades have witnessed great advances of many related techniques such as image signal digitization and storage or biological tissue labelling. As a consequence, the number of biological images acquired in digital forms is growing rapidly, and large bioimage databases such as *Allen Brain Atlas* [10] and the *Cell Centered Database* (CCDB) [11] are becoming available. These databases include diverse types of images, such as two-dimensional (2D), 3D spatial or even 4D spatio-temporal information, among others. Analyzing these images is critical for biologists to seek answers to many biological problems, such as differentiating cancer cell phenotypes, categorization of neurons, etc (Peng (2008)).

Bioimage informatics involves the generation, visualization, analysis and management of biological images, paying special attention to the generation of biological knowledge from image data. This field is becoming so important that some high-impact journals are opening up new categories for bioimage

[9] http://biotextquest.ucy.ac.cy/cgi-bin/textQuest/textQuest.cgi
[10] http://www.brain-map.org/
[11] http://ccdb.ucsd.edu/index.shtm

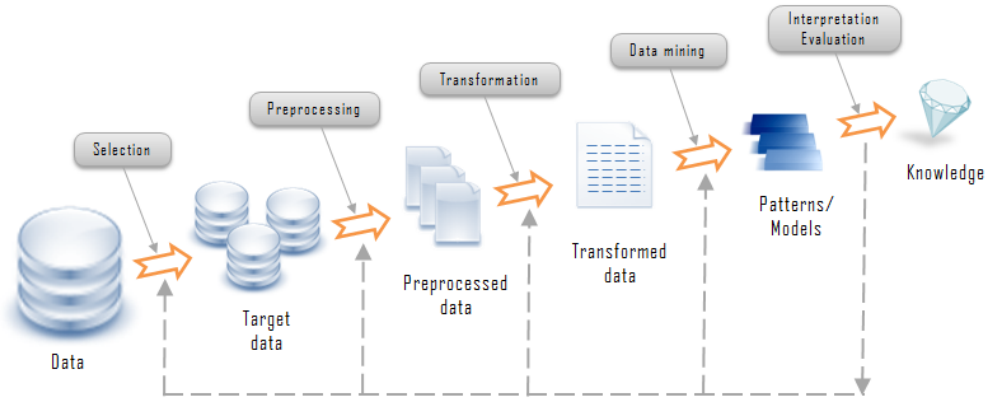


Figure 2.5: Knowledge discovery in databases

informatics. Peng (2008) and Swedlow et al (2009) review the advances of this field from several aspects, including descriptions of available bioimage resources, discussing also future developments in the area. In a very recent work, Kozhenkov & Baitaluk (2012) describe a resource of pathway diagrams retrieved from article and web-page images through optical character recognition, in conjunction with data mining and data integration methods, being the software and the pathway repository freely available [12].

2.3 Data Mining in Bioinformatics

The need for computational theories and techniques to assist in the extraction of useful knowledge from digital data gave rise to *Knowledge Discovery in Databases* field (KDD). With the substantial growth of biological data, KDD will play a significant role in analysing the data and in solving emerging problems. Figure 2.5 shows the general process of KDD for extracting interesting, non-trivial, implicit, previously unknown and potentially useful information from data.

Data mining is a step in the KDD process which corresponds to the application of specific algorithms for the analysis of a previously preprocessed data in order to obtain new and relevant patterns and facts. Extracting a pattern means fitting a model to data, finding structure from data, or, in general, making any high-level description of a set of data (Witten & Frank (2005)). In selection, processing and transformations steps in Figure 2.5 orig-

[12]http://www.biologicalnetworks.org

inal data is modified in several ways in order to reduce its dimensionality or noise, trying at the same time to minimize the loss of relevant information. Evaluation and interpretation of the results of the data mining process is usually carried out by an expert, though with the assistance of some computers tools. In the case of biological data, the question is how to bridge the two fields, KDD and bioinformatics for successfully discovering sequential patterns, gene functions, protein-protein interactions or any other useful knowledge depending on the input data (Wang et al (2005)).

Although every step of the KDD process are as important as data mining, we are going to focus our attention in the data mining component of KDD, which has received the most attention in the literature and also represent the central part of the proposals in this PhD Thesis. The data mining component of KDD relies heavily on known techniques from machine learning, pattern recognition, and statistics to find patterns from data (Bishop (2007)).

Two different goals can be achieved by using data mining techniques: verification and discovery. The first one parts from a user hypothesis needed to be checked out, while the second one aims at the finding of new patterns in the data. This finding can be subdivided into prediction, where the obtained patterns are used for behaviour prediction of new entities, and description, where the patterns are used for categorization and presentation of the entities under study. In the study and analysis of biological data, discovery-oriented data mining is mostly applied, although verification-oriented techniques are usually applied for validation procedures.

The common goal of the majority of data mining methods is developing models or determining patterns from observed data. Differences among distinct methodologies resides in the search method as well as in the goodness of the criterion used for the evaluation of the model or the fitting to a pattern Fayyad et al (1996). Although most methods can be viewed as extensions or hybrids of a few basic techniques and principles, we summarize in the following what we consider the most relevant techniques, for both predictive and descriptive purposes. Nevertheless, the boundaries between these two aspects are not sharp, since some of the predictive models can be descriptive and vice versa.

2.3.1 Supervised methods

Supervised methods attempt to discover the relationship between input attributes (independent variables) and a target attribute (dependent variable). The relationship discovered is often represented in a structure referred to as a model. In biology, these kind of methods have been broadly applied, among others applications, to the classification and identification of genes. Jensen

& Bateman (2011) have recently concluded in their experimental work that the application of this kind of methods continues to grow in the biomedical literature. For this purpose, they have created a list of supervised machine learning techniques used in bioinformatics, searching afterwards this list against PubMed [13] titles and abstracts.

The availability of a priori knowledge is crucial for applying any supervised methodology, since this previous information is used to train and calibrate the model on an input set. Furthermore, within the training process of the model, the most important attributes are computed, implicitly building up the subset of decisive attributes. For example, in the case of microarray data analysis, where attributes refer to genes, at the end of the process, the user is provided with the model for new genes predictions and also the set of decisive genes, thus gaining insights into the underlying molecular biology.

It is possible to distinguish between two main supervised models: *classification* models (classifiers) and *regression models*. Classifiers map the input space into pre-defined discrete classes, while regression models map the input space into a real-value continuous domain. In both cases, once the model has been constructed, it is crucial to use an independent test set or a cross-validation technique to estimate the classification/regression error.

Among the different applications of classification techniques in bioinformatics we would like to cite protein structure prediction, protein function prediction, genome annotation, biological sequences classification or classification of microarray data (Duval & Hao (2010)), among others. Regression has been mainly applied in bioinformatics to gene expression analysis, although it has also been used for different studies, including proteins (Giard et al (2009)).

Most common computer techniques for classification of biological data include:

- **Naive Bayes.** It is a probabilistic classifier that makes use of *Bayesian theory* and is the optimal supervised learning method if the predictors are independent given the class. Bayesian methods are used to obtain class predictions, while the naïvety arises from the independence assumptions. Although predictors independence is unlikely to be valid in real data, this technique appears to work well. The predictions will be accurate as long as the probability of being in the correct class is greater than that for any of the incorrect classes. Naive Bayes classifier has been broadly used in different bioinformatic fields, such as sequence (Murakami & Mizuguchi (2010)) or genome (Rosen et al (2011)) analyses.

[13] http://www.ncbi.nlm.nih.gov/pubmed

- **Artificial Neural Networks (ANN).** ANN are based on the use of parallel distributed processing in an attempt to simulate a real neural network, composed of a large number of interconnected, but independent neurons. *Artificial Neural Networks* are usually applied when data structures are not well understood, or when the main requirement is for a black box classifier rather than a classifier that provides insight into the nature of class differences. Linder et al (2004) applied ANN to the study of microarray data. It has also been applied to sequence and genome analyses (Viklund & Elofsson (2008) and Firpi et al (2010) respectively).

- **Support Vector Machines (SVMs).** *Support Vector Machines* are characterized by their accuracy and their ability to deal with a large number of predictors. SVMs extend the concept of hyperplane separation to data that cannot be separated linearly. The method name derives from the support vectors, which are lists of the predictor values obtained from cases that lie closest to the decision boundary separating the classes, having the greatest impact on the location of the boundary. SVMs are being extensively applied in may bioinformatic areas, including phylogenetics (Blouin et al (2009)), structural bioinformatics (Ng & Mishra (2007)).

- **Decision trees.** This kind of classifiers assign class labels to cases by following a path through a series of simple rules or questions, the answers to which determine the next direction through the pathway. Rules are created using the information within the predictors, where the two most important stages in the algorithm are identifying the best splits and determining when a node is terminal. *Decision trees* have no assumption of a linear model and they are particularly useful when the predictors may be associated in some non-linear fashion. Furthermore, they are non-parametric and predictors can be reused in different nodes with different threshold values. Decision trees applications within bioinformatics include structural proteomics (Huang et al (2007)) or system biology (Hautaniemi et al (2005)), among others.

- **k-Nearest Neighbours (k-NN).** k-NN is a type of *lazy learning* classifier, distinguished by delaying computation until classification time. Lazy learning algorithms access training examples to make a prediction when facing a new test example. In this case, the class of a new test case would be given by the majority vote of its nearest neighbours. Three fundamental issues need to be taken into account. Firstly, how is

distance measured to find the nearest neighbours. This distance function will have a large effect on the classifier performance and need to be adapted to the specific features of the application problem. Secondly, how many neighbours should be examined, where this number (k) may be obtained by cross-validation. Finally, neighbours do not need to make the same contribution to the final vote. For example, unequal class proportions create difficulties because the majority vote rule will tend to favour the larger class. Weighting neighbours according to their nearness seem to be the most adopted procedure. *K-Nearest Neighbours* classifiers have been mostly used in proteomics (Shen & Chou (2006), Masso & Vaisman (2008)) and also for microarray gene expression analysis (Mi et al (2010)).

Regression techniques uses the same principles of classification strategies, incorporating different variations in order to make them more suitable to the regression problem. In particular, decision trees, support vector machines and artificial neural networks have been extensively applied to regression problems in many contexts. Within bioinformatics, regression studies are recently receiving a special attention in different fields, such as system biology (Lu et al (2011)), population analysis (Lourenço et al (2011)) or gene expression analysis (Chen et al (2010)).

2.3.2 Unsupervised methods

Unsupervised strategies try to find hidden structure in unlabelled data, where no previous knowledge is available. The goal is therefore how to learn to represent particular input patterns in a way that reflects the statistical structure of the overall collection of input patterns, where there are no explicit target outputs or evaluations associated with each input (Barlow (1989)). In this sense, unsupervised methods corresponds to the branch of data description within data mining discovery techniques.

Cluster analysis is the most used approach for data analysis and description without previous knowledge. It consists of finding structures in data by identifying natural groupings of categories within the data (clusters). The categories can be mutually exclusive and exhaustive or consist of a richer representation, such as hierarchical or overlapping categories. A cluster is therefore a collection of categories or cases that are more similar to each other than they are to cases in other clusters. If any analysis produces obvious clusters it may be possible to name them and summarise its characteristics.

All clustering algorithms begin by measuring the similarity between the cases to be clustered, placing similar cases into the same cluster. It means

that there must exist some means of measuring distance among cases. In fact, there exist a great number of metrics by which distance can be measured, some of them restricted to particular data types and specific problems. Selecting an appropriate distance metric is very important, since different metrics will lead to different clusters and a consequential change in the results interpretation (Fielding (2007)).

Different clustering methods may be classified into four different categories, although they may be overlapped, since some methods might have features from several categories. In general, the major fundamental clustering methods applied in bioinformatics can be classified using the following taxonomy:

- **Partitioning methods.** A partitioning method creates k clusters of the input data, where each group must contain at least one object and k is provided by the user. Typically, this kind of methods performs exclusive cluster separation, where each object must belong to exactly one group, although this requirement might be relaxed. Achieving global optimality of the clustering is often computationally prohibitive, requiring thus the application of different heuristics. *k-means* and *k-medoids* are two popular partitioning clustering methods based on the use of greedy approaches. Harder et al (2012) have recently apply k-means clustering for clustering protein structures, while Hazelhurst & Lipták (2011) use a partition clustering method for the analysis of expression data.

- **Hierarchical methods.** This kind of methods creates a hierarchical decomposition of the given set of data objects. Two different strategies are used: the *bottom-up* approach, in which each object initially forms a cluster and the algorithm iteratively merges different groups; and the *top-down* approach, where all the objects are initially placed in the same cluster and are iteratively separated into smaller clusters. This kind of methods are useful when the data has to be partitioned into groups at different levels and the number of desired clusters is not known beforehand. *Self-Organizing Tree Algorithm* (SOTA) (Dopazo & Carazo (1997)) is an example of a top-down hierarchical method which was originally developed for clustering of biological sequences (both protein or nucleotide). More recently, Miele et al (2012) have applied a bottom-up approach for the clustering of homologous sequences, where the initial group of clusters is obtained by comparing all of them with each other and clustering them into pre-families, based on pairwise similarity criteria.

- **Density-based methods.** They are based on the concept of *density* instead of distances measures. The general idea is to continue growing a given cluster as long as the number of objects in the neighbourhood exceeds some threshold. This kind of methods are useful when the data contains noise or outliers or when the shape of the clusters are not spherical. One of the most popular algorithms in this category is DBSCAN (*Density-Based Clustering Based on Connected Regions with High Density*) (Ester et al (1996)). DBSCAN identifies core objects in the data, which are those having dense neighbourhoods. Furthermore, an user-specified parameter is used to specify the radius of the considered neighbourhood. DBSCAN has been applied to population analysis studies in order to identify low-quality SNPs (Pongpanich et al (2010)).

- **Grid-based methods.** These methods quantize the object space into a finite number of cells that form a grid structure, where all the clustering operations will be performed. The main advantage of these approaches is their fast processing time, independent of the number of data objects, yet only dependent on the number of cells in each dimension. Two typical examples of algorithms in this category are STING and CLIQUE. STING (*STastistical INformation Grid*) (Wang et al (1997)) partitions the dataset into rectangular cells, where a cell at a high level is partitioned to form a number of cells at the next lower level, and hierarchical levels of cells correspond to different levels of resolution. CLIQUE (*CLustering In QUEst*) (Agrawal et al (2005)) is useful for clustering high-dimensional data and is insensitive to object ordering. CLIQUE partitions space into non-overlapping rectangular units and identifies the dense units (a unit is dense if the fraction of total objects contained in it exceeds an user-specified value) and considers only hyper-rectangular clusters and projections parallel to the axes.

Hierarchical clustering and k-means partitioning are the most popular methods being used in bioinformatics. Numerous improvements of these two clustering methods have been introduced, as well as completely different approaches such as grid-based or density-based. Andreopoulos et al (2009) present in their work a set of desirable clustering features that are used as evaluation criteria for clustering algorithms in bioinformatics, reviewing and comparing many different algorithms.

Clustering can also be performed on two dimensions simultaneously whenever the data can be represented in the form of a matrix. This technique is

named *co-clustering* or *biclustering* and also covers other characteristics, such as overlapping (inclusion of each element into none or more than one group). The resulting groups of elements found by biclustering approaches are called biclusters. Biclustering has been mainly applied to microarray data. Chapter 4 review the most important biclustering approaches for gene expression microarray data analysis.

In other applications, such as protein structure classification, only a few labelled samples (protein sequences with known structure class) are available, while many other samples (sequences) with unknown class are available as well. In such cases, semi-supervised techniques can be applied to obtain a better classifier than could be obtained if only the labelled samples were used. This is possible, for instance, assuming that class labels can be reliably transferred from labelled to unlabelled objects that are nearby in feature space. We refer the reader to Zhu (2005) survey on semi-supervised learning for more information on this topic.

2.4 Summary

This chapter consists of three main sections referring to the three most important topics within bioinformatics: biology, computer science, and the intersection between them. The goal of this chapter is to provide the reader with the necessary background towards the correct understanding of the contents of this document. This way, the principles of life are first described, including structural cell biology concepts and molecular genetics. They are summarized at a very superficial level but deep enough for the comprehension of the rest of contents in this PhD Thesis.

The areas of biology in which computer science techniques are being currently applied have been summarized in section 2.2, organized according to their specific sub-categories, up to a total of fourteen research fields. All of them have been categorized into one of the six following groups: sequence analysis and comparison, comparative genomics, gene expression analysis, proteomics, system biology or other related research fields.

Finally, last section of this chapter introduces the most common computational techniques applied to the different bioinformatic fields. Our attention has been centred towards data mining approaches, corresponding to specific algorithms for the analysis of previously preprocessed data. The output obtained from the use of these kind of tools reveals previously unknown and potentially useful information from the examined data.

Part II

Foundations

Chapter 3

Microarray: Technology and Analysis

This chapter is dedicated to the presentation of microarray technology and the current tendencies in the analysis and interpretation of the experiments results. Although microarrays have already been introduced in section 2.2.3, we give here a more in-depth description of this widespread technology. This way, after exposing the different kinds of microarray and its experimental cycle process, we focus our study on gene expression microarray high-level analysis. High-level analysis correspond to the phase in which the raw data from the microarray experiment has already been filtered and pre-processed, giving thus way to more complex computational analysis aiming at gaining insights into the data. We finalise this chapter with the description of the available biological databases for the verification and interpretation of the high-level analysis output.

3.1 Microarray Technology

Molecular Biology research evolves through the development of the technologies used for carrying them out. Since it is not possible to research on a large number of genes using traditional methods, DNA microarray enables the researchers to analyse the expression of many genes in a single reaction quickly and in an efficient manner.

DNA microarrays were invented by Schena et al (1995) in 1995. The first microarray ever is shown in Figure 3.1. By putting some microscopic DNA spots on a solid surface, the microarray chips are able to measure at the same time the expression level of an important number of genes for a tissue. The measure can be done using fluorescence as was the case with the

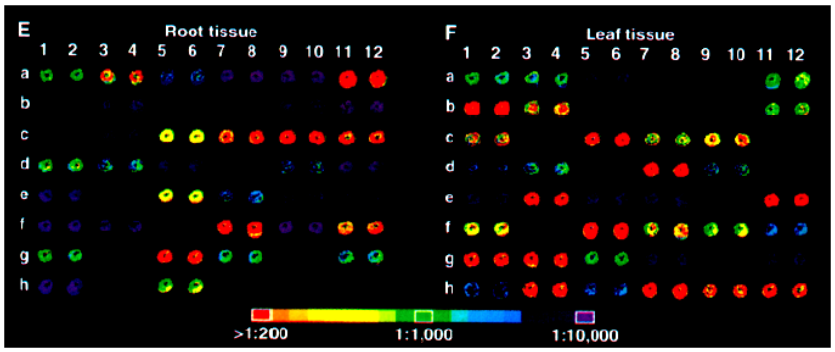


Figure 3.1: The first DNA microarray

first invented microarray. This technology could only measure 48 genes at one time whereas now we can measure tens thousands of genes. The use of microarrays increased considerably from 2000, mainly due to the sequencing of the human genome.

DNA microarrays exploit nucleic-acid sequences complementarity. A microarray is composed of a surface area to which single-stranded DNA molecules are attached at fixed locations, named spots. There may be tens of thousands of spots on an array, where each of them contains thousands of copies of a sequence that matches a segment of a gene coding sequence (probes). Different spots typically represent different genes, but some genes may be represented by multiple spots. When microarrays are used to detect mRNA transcripts, RNA from the samples to be analysed are extracted, reversed and transcribed into cDNA (cDNA is a strand of DNA that is complementary to part of an mRNA), as shown in Figure 3.2. After this process, they are labelled with a fluorescent dye and placed on the microarray, hybridizing to their complementary sequences in the spots. The presence and abundance of specific target sequences within the sample is reflected by the intensity of the hybridization signal at the corresponding probe locations. If the RNA within the sample is present in large copy numbers, the spot will be intense; otherwise the signal from the spot will be faint or even absent. The particular used dye label make it possible to quantify the amount of nucleic acid bound to the probe on a spot, where a laser excites the dye and a scanner records an image of the slide.

Depending on the specific application, there exists many different types of microarrays. We have focused this doctoral dissertation on gene expression microarrays, which provide a snapshot of all the transcriptional activity in a biological sample, and constitutes the most widespread use of microarrays. The true power of microarray analysis does not come from the analysis of

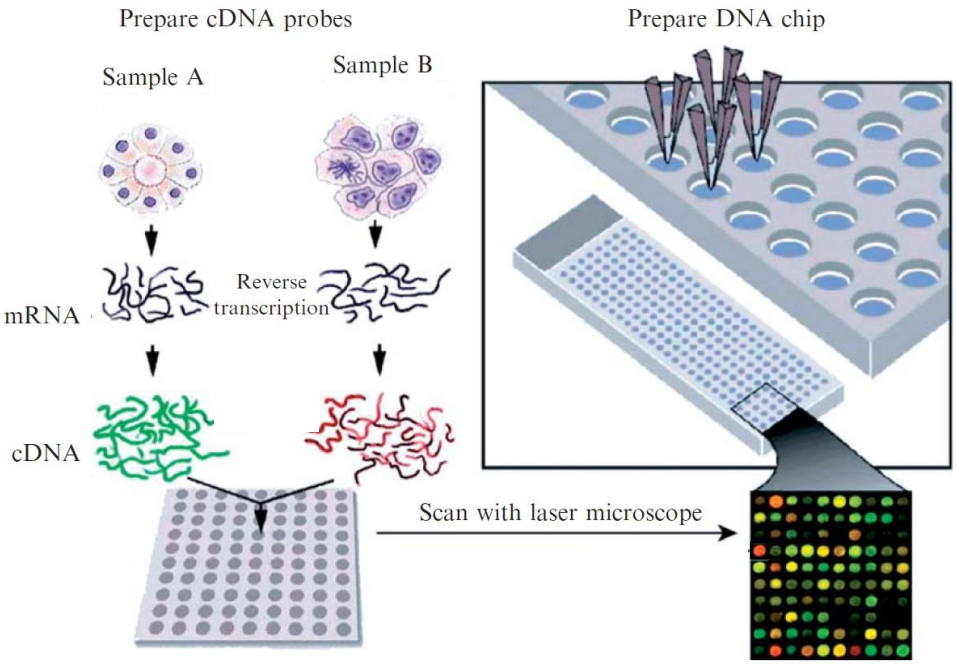


Figure 3.2: Microarray production process

single experiments, but from the analysis of many hybridizations to identify common patterns of gene expression. For this reason, after visually analysing several related microarrays chips we are provided with bi-dimensional matrices of data, called expression matrices (see Figure 3.3). Rows of an expression matrix represent the different genes under study, while columns represent experimental conditions, such as temporal conditions, tissues, patiences, etc. This way, every element of such expression matrix stands for the expression level of a given gene under a specific condition (Baldi & Hatfield (2002); Tilstone (2003)).

3.1.1 Applications

The applications of microarrays are diverse. For example, they can be used to increase our understanding of the relationships existing between genes or to find the genes related to a particular disease. It may also be possible to associate a gene profile with a specific therapy in order to provide a person-

alized treatment. Gene profiles can also be used to discover subclasses of diseases or to automatically give a prognosis for a patient.

According to Iida & Nishimura (2002), applications of gene expression microarrays can be classified in four different categories:

- Gene discovery: Gene discovery is one of the main applications of mRNA measurement in microarrays because of the rich information that can be derived about the functions of genes in cells and tissues. The list of tissues where a gene is expressed provides a key clue to the function of the gene. Also, if two genes have similar patterns of expression across tissues, this is a clue to functional relatedness. Kuninger et al (2004) demonstrate in their work the power of transcriptional profiling for gene discovery, providing also opportunities for investigating new proteins potentially involved in different aspects of growth factor action in muscle.

- Disease diagnosis: The most common expression profiling experiment design compares two biological conditions, such as disease state versus normal state. Genes up-regulated (or down-regulated) offer a detailed molecular phenotype of the disease. DNA Microarray technology helps researchers learn more about different diseases and especially the study of cancer. Until recently, different types of cancer have been classified on the basis of the organs in which the tumours develop. Now, with the evolution of microarray technology, it will be possible for the researchers to further classify the types of cancer on the basis of the patterns of gene activity in the tumour cells. Yoo et al (2009) focused their review on the applications of DNA microarrays for diagnosing diseases, although application of microarrays related to cancer studies is not included in this work, since cancer diagnosis has been paid a special attention in other works, such as the one of Perez-Diez et al (2005).

- Drug discovery: Molecular species are spatially sorted in a microarray. Therefore, their concentrations can be independently estimated, giving thus way to the study of any complex reaction product. Pharmacogenomics takes advantage of this characteristic to study the correlations between therapeutic responses to drugs and the genetic profiles of the patients, identifying the biochemical constitution of the proteins synthesised by diseased genes and designing specific drugs for reducing the effect of these proteins. In his work, Hardiman (2008) compares and contrasts microarray platforms currently in use in the context of pharmacogenetic testing.

- Toxicological research: Toxicogenomics field has emerged thanks to the use of microarrays for analysing the effect of chemicals on a large number of genes in a single experiment. The measurements of gene expression levels upon exposure to a chemical can be used both to provide information about the mechanism of action of the toxicant and to form a sort of genetic signature for the identification of toxic products. Lettieri (2006) reviews in his work recent applications of microarray to toxicology, also evaluating the potential of its application to ecotoxicology.

3.1.2 Experiment Cycle

The design of a microarray experiment is a procedure which comprises five main steps. A summary of the whole process can be seen in Figure 3.3. The process starts with a biological question that needed to be answered and ends up with the biological interpretation of the results.

1. **Experimental design.** This phase consist of the definition of the objectives, selecting the genes and experimental conditions under study, as well as choosing the platform, the marking methodology and the number of replicates. In Figure 3.3 this phase is depicted by experimental box.

2. **Data generation.** Raw data after the microarray has been created is obtained in this phase. This phase correspond to data generation in Figure 3.3

3. **Data Pre-Processing.** Raw data from the former step need to be pre-processed before being used. This step comprise background correction of the image, value extraction, data normalization and data summarization. After this step, if the quality of the data is not good enough the process continues with the first step.

4. **High-level analysis.** Once the expression matrix has been obtained, it is essential to perform one or more statistical and computational methodologies to the data in order to extract useful and relevant information. Although numerous data mining techniques are applied to microarray data nowadays, there is still many work to do within this field.

5. **Biological verification and interpretation.** This last phase corresponds to the biological interpretation of the obtained results after all

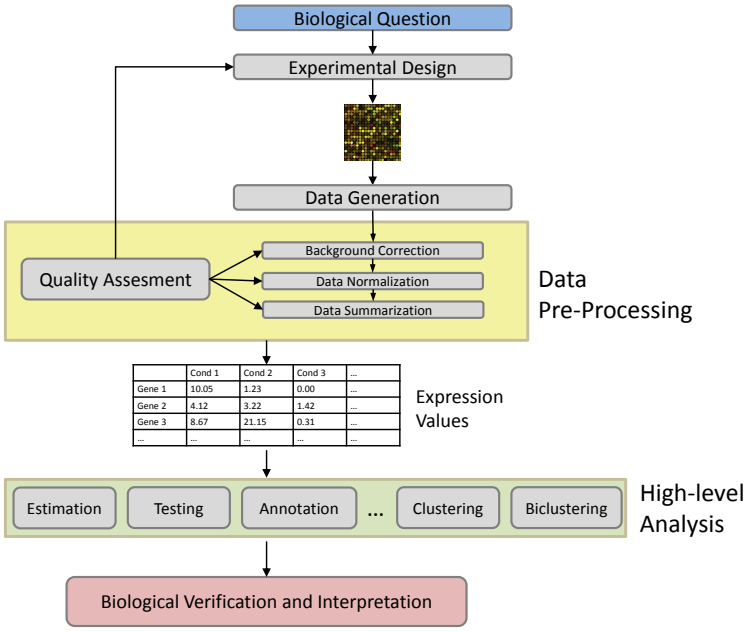


Figure 3.3: Microarray experiment cycle

the former phases. This interpretation give us the opportunity to have new hypothesis and new insights to the input biological question.

In this PhD Thesis, we focus our work in phases four and five.

3.2 Gene Expression Microarray Analysis

The task of analysing microarray data typically consumes considerably much more time than the laboratory protocols required to generate the data. Part of the challenge is assessing the quality of the data and ensuring that all samples are comparable for further analysis. Many methods for data pre-processing such as visualization, quality assessment, and data normalization have been developed (Quackenbush (2002)). When multiple probes represent a single transcript, they are usually reduced to a single expression measurement by means of summarization (Bolstad et al (2003)).

Once the final dataset has been generated after pre-processing, different

high-level analyses may be applied over the obtained data matrix. The methods that are used to analyse the data can have a profound influence on the interpretation of the results. Therefore, it is important to have a basic understanding of the available computational tools, in order to design an optimal experimental procedure and a meaningful data analysis.

Identification of differential gene expression is the first task of an in depth high-level microarray analysis. Since microarray data sets are typically very large, it is extremely useful to reduce the dataset to those genes best distinguished between the different cases or classes. Such analyses produce a list of genes whose expression is considered to change and known as differentially expressed genes. After this process, a new reduced dataset is obtained, which is analysed in order to identify patterns of gene expression which provide much greater insight into their biological function and relevance. In the following subsections we analyse existing data mining techniques to carry out these tasks.

3.2.1 Identification of Differentially Expressed Genes

The main issue in differential expression analysis is the experiment group size, which is always smaller than the number of genes or transcript to be investigated. The simplest method for identifying differentially expressed genes is to evaluate the log ratio between two conditions, considering all genes that differ by more than an arbitrary cut-off value to be differentially expressed (Draghici (2002)). This non-statistical test is called *fold-change*, and does not provide any associated value indicating the level of confidence in the designation of genes as differentially expressed. Furthermore, it is subjected to bias if the data has not been properly preprocessed. Although there is no firm theoretical basis for selecting a cut-off level as significant, the approach defined by Chen et al (1999) provides confidence intervals that can be used to identify differentially expressed genes.

Apart from fold-change, there are several statistical methods to determine either the expression or relative expression of a gene from normalized microarray data. A standard t test can be conducted for each gene (gene-specific t test) and it will not be affected by heterogeneity in variance across genes since it only uses information from one gene at a time. Nevertheless, it sample size may affect it powerfulness, and also the variances estimated from each gene are not stable. In a global t test, the variance is assumed to be homogeneous between different genes, although it may suffer from the same biases than a fold-change method if the error variance is not constant for all genes. As the error variance is hard to estimate, several modified versions of the t test have been developed in order to overcome gene-specific and global

t tests.

SAM (*Significance Analysis of Microarrays*) is a modified gene-specific t test where a constant called *fudge factor* has been added in order to remove the stability problem (Tusher et al (2001)). *Regularized* t test (Baldi & Long (2001)) combines gene-specific and global approaches, and has been proven to outperform SAM by Choe et al (2005). Lönnstedt & Speed (2001) *statistic* B avoids the problems of using averages or t-statistics, working at a gene-specific level but also combining information across many genes. Smyth & Smyth (2004) developed the hierarchical model of Lönnstedt & Speed (2001) into a practical approach for general microarray experiments with arbitrary numbers of treatments and RNA samples. Finally, RP (Breitling et al (2004)) is based on calculating *Rank Products* (RP) from different experiment. Its performance is particularly strong when the data are contaminated by non-normal random noise or when the samples are very non-homogeneous. However, it assumes equal measurement variance for all genes. Other rank-based variant provides an alternative when it is not the case (Breitling & Herzyk (2005)).

Two types of errors may be committed when conducting a single hypothesis test: *false positive* error when a gene has been declared to be differentially expressed but it is not, and *false negative* error when a differentially expressed gene has not been detected as such. In a microarray experiment, thousand statistical tests are usually performed (one for each gene), where an important number of false positive may be accumulated. In order to overcome this situation, *multiple testing corrections* need to be carried out. FWER (*Family Wise Error Rate*) strategies such as *Bonferroni correction* or FDR (*False Discovery Rate*) procedures are usually applied.

Differentially Expressed Genes in Multi-series Microarray Experiments

Microarray experiments usually involve more than two conditions (multi-series experiments). In occasions, these experimental conditions are also time related (multi-series time-course experiments). In these types of experiments, the researcher is interested in studying gene expression changes over different experimental conditions and in evaluation trend differences among various experimental groups. If this is the case, ratios cannot be simply computed, and a more general concept of relative expression is needed. ANOVA analysis of variance is often applied to obtain a derived data set, where the relative expression of each gene in each sample is estimated (Kerr et al (2000)).

This new data set is ready to be analysed for testing differences among conditions. Different generalizations of the methodologies described for the

two-sample microarray experiments might be applied. For example, classical ANOVA F test is a generalization of the t test for the comparison of more than two samples, existing also several variations on this test.

When working with time-series expression data, specific variations may be found. In particular, Conesa has published two different variations of the F test: maSigPro (Conesa et al (2006)) and ANOVA-SCA (Nueda et al (2007)).

3.2.2 Classification and Clustering Analyses

Classification and Clustering techniques have already been mentioned in section 2.3. However, in gene expression analysis it is not possible to identify a universal classification or clustering approach because the optimal algorithm to be used depends on the nature of the dataset and what constitutes meaningful groupings in the problem under analysis. Therefore, although classification is by no means a new subject in the statistical literature, the large and complex multivariate datasets generated by microarray experiments raise new methodological and computational challenges.

Classification

Classification is a prediction or learning problem in which the variable to be predicted assumes one of K unordered values, corresponding to K predefined classes, e.g., tumour class or bacteria type. The task is to classify new objects into one of the K classes on the basis of a set of labelled objects, named the training or learning set.

For microarray data, prediction generally refers to the classification of patients samples by characteristics such as disease subtype or response to treatment. The goal might be diagnostic (Buness et al (2009)) or an effort to predict clinical outcome (Pomeroy et al (2002)).

Gene expression data presents several challenges for the application of typical machine learning algorithms. These kind of algorithms are generally designed for large number of samples and a few variables, while data from a microarray experiment correspond to the contrary situation, involving thousands of genes (variables) and only a few samples (tens or hundreds at most). Another important characteristic of microarray data is the high level of noise, where many data points may be missing.

Choosing a classification method requires selecting from a vast range of techniques (see section 2.3), and constitutes an important task, since the outcome may differ from applying one or another prediction technique. Furthermore, several other issues may be taken into account for the selection, like

the number of classes to distinguish, which imposes a modest constraint on the choice, since some algorithms only distinguish between two classes. The number of predictive genes, or the characteristics of the classification model itself (Slonim (2002)) are also useful in the selection process. Predictive error rate should also be estimated using methods such as cross-validation. Wood et al (2007) review in their work existing methods for estimating and reporting classification prediction accuracy. Another important issue is whether the classification algorithm should consider the data for all available genes, or whether prior gene selection (through differential expression detection) should be applied to reduce the data dimensionality. For example SVM classifiers have been proven to improve their prediction when using the whole set of genes (Ramaswamy et al (2001)).

Asyali et al (2006) review in their work different class-prediction and discovery methods that are applied to gene expression data, also discussing other related issues such as pre-processing and feature selection. Buness et al (2009) present a systematic evaluation of strategies for the integration of independent microarray studies in a classification task, assessing the potential benefit of data integration on the classification accuracy in breast cancer studies, and making use of several classification strategies.

Clustering Analysis

In clustering, the analytical goal is to find clusters of samples or clusters of genes such that observations within a cluster are more similar to each other than they are to observations in different clusters. Cluster analysis can also be viewed as a data reduction method in that the observations in a cluster can be represented by an 'average' of the observations in that cluster.

Eisen et al (1998) demonstrated in an early paper the potential of clustering techniques to reveal biologically meaningful patterns in microarray data. The many applications of clustering in expression data analysis depend on the dimension in which the clustering is applied. Clusters of genes similarly expressed allow to identify groups of co-regulated genes and spatial or temporal expression patterns, offering thus clues to unknown gene function. Clustering samples aids in the identification of new disease subclasses.

In general, all the issues that must be addressed for classification must also be addressed for clustering. In addition, with clustering there is no learning set of labelled observations, and the number of groups is usually unknown previously. In particular, choosing the right number of clusters is crucial for many clustering algorithms, being particularly problematic for microarray data, which may not have any solution isolating compact clusters. Ben-Hur et al (2001) addressed this problem by using repeated sampling to

determine the number of clusters that provides the most stable solution.

Choosing a clustering method (see section 2.3) is still very much dependent on the data and goals than in the classification case. In particular, the selection of the distance measure is crucial since the output grouping of the data is highly dependent upon the distance measure used. Furthermore, many clustering algorithms are computationally too complex to have exact solutions and approximate solutions are used instead, where reproducibility is not guaranteed.

Tibshirani et al (1999) review the application of different clustering approaches to microarray data, including hierarchical, K-means, block clustering, and tree-structured vector quantization. Shannon et al (2003) also study the application of several clustering techniques to microarray data analysis, highlighting the need for the inclusion of biological knowledge into the process. Finally, Genesis (Sturn et al (2002)) was presented as a platform that integrates various tools for microarray data analysis, also including common clustering algorithms such as hierarchical clustering, self-organizing maps, k-means, principal component analysis, and support vector machines.

3.2.3 Other Analyses

Identification of differentially expressed genes and classification and clustering approaches are considered to be basic microarray data analysis task, in which the analysis is performed on gene expression profiles alone. However, gene expression profiles can be linked to other biological external resources, providing thus new biological discoveries and knowledge. In general, models that incorporate existing constraints from other data sources seem to produce hypotheses that agree better with existing biological knowledge than do models learned from the expression data alone (Hartemink et al (2002)). Some of the common applications in this sense are transcription factor biding site analysis, pathway analysis, and protein-protein interaction network analysis.

Transcription factors play a prominent role in transcription regulation, acting as critical molecular switches in the gene expression profiling. Therefore, identifying and characterizing their binding sites is essential for the annotation of genomic regulatory regions and understanding gene regulatory networks. In their work, Pritsker et al (2004) use network-level conservation information in order to map transcription factor binding sites on a genomic scale.

Protein-protein interaction network information might be combined together with expression data in order to perform predictions related to gene functions and evolutionary relationships. Bhardwaj & Lu (2005) investigate

the global relationship of protein-protein interactions with gene expression, also integrating ortholog information, and demonstrating that protein interactions are reflected in gene expression and that the correlation between the two is strengthened by evolution information.

Pathway analysis involves looking for consistent but subtle changes in gene expression by incorporating either pathway or functional annotations, thus allowing the identification of the mechanisms that underlie diseases, adaptive physiological compensatory responses and new avenues for investigation. Curtis et al (2005) review in their paper several methods of pathway analysis and compare the performance of three of then on two microarray datasets.

3.2.4 Biological Databases for Verification and Interpretation

Thanks to journals, databases and repositories, the outcomes of multiple research on biological data become available almost at real time. As a result, the knowledge of biology grows everyday at a high rate, and every biological experiment, such as microarray experiments, are designed based on the available knowledge. This way, microarray data analysis makes use of this information to validate its results, and also to tighten and simplify the analysis.

In this subsection we briefly survey the most common information sources for microarray data analysis validation and interpretation, considering both dimensions of a microarray experiment, genes and experimental conditions. We finally discuss some consideration on the use of different sources of information and their interoperability.

Gene Information Resources

Biological knowledge related to genes can be classified in three categories. As information may come from very different sources, a great effort has to be made by the scientific community in order to integrate all this knowledge into public repositories and databases. In the following we enumerate the different categories, together with the most common used repositories.

- Basic information. It correspond to the general information on genes, including a brief description, gene name, synonyms, organism, location, sequence and other general characteristic of genes. Although it is the most stable information, some changes may occur if, for example, unknown functions or locations are discovered. Entrez Gene (Maglott

et al (2011)) from the *National Center for Biotechnology Information* (NCBI) is the most important database of genes. It maintains records from completely sequenced genomes and the content represents the integration of curation and automated processing from the *Reference Sequence project* (RefSeq), collaborating model organism databases, consortia such as Gene Ontology and other databases within NCBI. Entrez Gene is accessible via interactive browsing, via programming utilities and for bulk transfer by FTP.

- Annotations. Annotations relate genes with different biological concepts. *Gene Ontology* (GO) is the most used database of annotations, which links genes to biological terms from a vocabulary of GO terms. GO (Ashburner et al (2000)) is divided into three ontologies: *Biological Process* (BP), which contains terms related to a biological objective to which the gene or gene product contributes, *Molecular Function* (MF), defining biochemical activities of a gene product and *Cellular Component* (CC), which refers to the place in the cell where a gene product is active. It also coordinates the annotation of several organisms, including human, yeast and rat, among others. There exist also a great variety of tools [1] in order to access the database and perform different kind of analysis using its information.

- External relationships. This category correspond to those resources in which information on gene relationships to other biological concepts, such as gene products (proteins), biological pathways or transcription regulatory networks is stored. UniProt [2] (*Universal Protein Resource*) is the main database for proteins, and provides the scientific community with a comprehensive, high-quality and freely accessible resource of protein sequence and functional information. It also allows searches by gene, so it is possible to link a gene to its gene products. The *Kyoto Encyclopedia of Genes and Genomes* [3] (KEGG) is probably the best resource for gene mapping to biological pathways, although there exist some other biological pathway databases such as BioCarta [4] and Reactome [5].

[1] http://www.geneontology.org/GO.tools.shtml
[2] http://www.uniprot.org/
[3] http://www.genome.jp/kegg/
[4] http://www.biocarta.com/
[5] http://www.reactome.org/

Condition Information Format

Unfortunately, the information on microarray experimental conditions is not as well structured in computational resources as gene related information is. Usually, natural language is used to describe the conditions for a microarray experiment, and they are only available through publications. The *Microarray Gene Expression Data Society* [6] (MGED) was founded in order to provide a collaborative atmosphere where the basic and central issues of data gathering, handling and analysis of high-throughput gene expression technologies can be discussed and solved. Primary interests of MGED included the elaboration of a guide including experiment description and data representation standards, a microarray data XML exchange format, ontologies for sample description, and other issues related to microarray experiments development and analysis.

Within MGED, MIAME was developed by Brazma et al (2001) in order to describe the *Minimum Information About a Microarray Experiment* that is needed to enable the interpretation of the results of the experiment unambiguously and potentially to reproduce the experiment. Regarding the information on conditions, MIAME says that it must be included the experimental factors and their values, where an *experimental factor* (EF) refers to a variable of the experiment (for example the studied organism, age, sex or disease state), and an *experimental factor value* (EFV) specifies a concrete value of an EF for a given experiment. MIAME does not specify a particular format, however, FGED recommends the use of MAGE-TAB format, which is based on spreadsheets, or MAGE-ML.

Major microarray public repositories are MIAME-compliant. ArrayExpress (Brazma et al (2003)) is a public database of microarray gene expression data at the *European Bioinformatic Institute* (EBI). It uses the annotation standard MIAME and the associated XML data exchange format *Microarray Gene Expression Markup Language* (MAGE-ML). *Gene Expression Omnibus* (GEO) (Barrett & Edgar (2006)) a public functional genomics data repository supporting MIAME-compliant data submissions. It archives and freely distributes microarray, next-generation sequencing, and other forms of high-throughput functional genomic data submitted by the scientific community.

Data Integration

Biological data are often most valuable when it can be integrated with other types of related biological data from different sources of information. These kind of analyses require individual researches to assemble and integrate enor-

[6] http://www.mged.org/

mous amounts of data gathered from different remove providers. Typically, there exist many integration problems, such as the fact that each of the database resources contains a different subset of biological knowledge, according to different domains. Furthermore, other differences among databases might be taken into account, including data format compatibilities, together with differences in the nomenclatures and the ways of accessing the data. Nevertheless, the design of a single biological database would be a poor solution, since it would reflect a series of compromises that would impoverish the currently available information resources. A better solution would consist of a different design, consisting of a search engine able to cross-reference the original databases, thus maintaining the diversity of expertise and interests of every primary resource. In this context, Bellazzi et al (2012) describe in their work some of the most important design features of an infrastructure of this kind, arguing that the widespread availability of clinical data warehouses and the noteworthy increase in data storage and computational power justify research and efforts in this domain.

Three different integration approaches have been reviewed and discussed in the literature (Stein et al (2003)): link integration, data warehousing and database federations. In the following we give a brief description of each one of them:

- Link integration or navigational approach: This approach correspond to the simplest one, where the different databases are connected via hypertext links, and navigating around the hypertext links is like flipping from one entry in an encyclopaedia to another. Although this solution is simple to maintain and easy to use, it requires a significant amount of manual work for data integration. Furthermore, there exist some other disadvantages related to the use of different nomenclatures and ambiguities, and also to the used of cross-references, since the links need to be updated so that they are always valid.

- Data warehousing: A data warehouse is a single database that is constructed by physically consolidating a collection of data sources into an integrated one. In the creation process, the data from each original source must be converted to a compatible form in the warehouse, which constitutes a complex problem, also dependant on the number of databases to merge. The biggest problems in this scheme are scalability and keeping the data up to date. This way, maintainer labours need to be frequently carried out, sometimes involving more complex modifications, such as adding new data types, relationships or even including adaptations in the data model, according to the adjustments in the different sources.

- Database federations: The main idea behind this scheme is that interoperability does not require the global integration. This way, data is left in their native locations and representations, and a mediator software is used for translating user queries, optimizing them, connecting the original data resources and returning the required information. Although the development of such a software might seem very costly, the investment is justified when all responsibility for data management and updating is left to the provider. The most important drawback of this approach is its slow performance, since all data conversions and merging procedures need to be done at runtime. Furthermore, a single query may require different database accesses, at their different locations, adding a considerable vulnerability due to possible sudden loss of service at the provider site.

3.3 Summary

In this chapter we have centred our attention on microarray technology, and more specifically on gene expression microarrays, where the goal is the identification of differentially expressed genes. After describing this technology and its experimental cycle process, we focussed our study on high-level analysis. This experimental phase corresponds to the development and use of analytical methods to retrieve useful information, once raw data from the microarray experiment has been filtered out and preprocessed. Identification of differentially expressed genes and classification and clustering approaches are considered to be the basic high-level microarray data analysis tasks, where the analysis is performed on gene expression profiles alone. Finally, the last section of this chapter is dedicated to reviewing the most used public biological databases for the verification and interpretation of the high-level analysis output.

Chapter 4

Biclustering Algorithms

In section 3.2, different kinds of analysis of microarray data were detailed. Clustering analysis, which was also described as an unsupervised data mining technique in section 2.3, has been extensively applied to microarray data. The goal is to extract information on how gene expression levels vary among the different samples, finding groups of co-expressed genes. If two different genes show similar expression tendencies across the samples, this suggests a common pattern of regulation, possibly reflecting some kind of interaction or relationship between their functions Baldi & Hatfield (2002).

Nevertheless, there exists two main restrictions in the use of clustering algorithms: (1) genes are grouped together according to their expression patters across the whole set of samples, and (2) each gene must be clustered into exactly one group. This last limitation is two-fold: firstly, it means that a certain gene cannot be present in different groups, thus forbidding overlapping among clusters; secondly, it confines each gene to be a member of any cluster, even if it is not co-regulated with any of the other genes in the cluster.

However, genes might be relevant only for a subset of samples. This is essential for numerous biological problems, such as the analysis of genes contributing to certain diseases, assigning biological functionalities to genes or when the conditions of a microarray are diverse (Wang et al (2002)). Thus, clustering should be performed on the two dimensions (genes and conditions) simultaneously. Also, many genes may be grouped into diverse clusters (or none of them) depending on their participation in different biological processes within the cell (Gasch & Eisen (2002)). These characteristics are covered by biclustering techniques, which have also been largely applied to microarray data (Madeira & Oliveira (2004); Tanay et al (2005); Busygin et al (2008); Eren et al (2012)). The groups of genes and samples found by biclustering approaches are called biclusters.

Biclustering was introduced in the 1970s by Hartigan (1972), although Cheng & Church (2000) were the first to apply it to gene expression data analysis. Other names such as co-clustering, bi-dimensional clustering, two-way clustering or subspace clustering often refer to the same problem formulation.

Finding significant biclusters in a microarray is a much more complex problem than clustering (Divina & Aguilar-Ruiz (2006)). In fact, it has been proven to be a NP-hard problem (Tanay et al (2002)). Consequently, the majority of the proposed techniques are based on optimization procedures as the search heuristic. The development of both a suitable heuristic and a good cost function for guiding the search is essential for discovering interesting biclusters in an expression matrix. Nevertheless, not all existing biclustering approaches base their search on evaluation measures for biclusters. There exists a diverse set of biclustering tools that follow different strategies and algorithmic concepts which guide the search towards meaningful results.

We focus our work on the importance of having a suitable evaluation measure for biclusters. It can be used for guiding the search in different heuristics, allowing thus contrasting distinct search strategies. Furthermore, it can also be used for comparing the results of different biclustering approaches, based or not on evaluation metrics, which is jet an unresolved task nowadays.

In this chapter, we have collected all information related to the biclustering problem, including the categorization of different types of patterns in biclusters and currently available quality measures for evaluating biclusters. We have also reviewed the main biclustering approaches existing in the literature, emphasising those based on the use of evaluation measures.

4.1 Bicluster Unified Notation

Biclusters are represented in the literature in different ways, where genes can be found either in rows or columns, and different names refer the same expression sub-matrix. In this section we define a common notation for biclusters, which will be used not only in this chapter but in the whole document. This way, we facilitate the reader understanding different authors approaches, in spite of adapting the original definitions.

Let, from now on, $\mathcal{B}$ be a bicluster consisting of a set I of $|I|$ genes and a set J of $|J|$ conditions, in which b_{ij} refers to the expression level of gene i

under sample j. Then $\mathcal{B}$ can be represented as follows:

$$\mathcal{B} = \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1|J|} \\ b_{21} & b_{22} & \dots & b_{2|J|} \\ \vdots & \vdots & \ddots & \vdots \\ b_{|I|1} & b_{|I|2} & \dots & b_{|I||J|} \end{pmatrix}$$

where the gene g_i is the i^{th} row, e.g., $g_i = \{b_{i1}, b_{i2}, \dots, b_{i|J|}\}$, and condition c_j is the j^{th} column, e.g., $c_j = \{b_{1j}, b_{2j}, \dots, b_{|I|j}\}$.

In the evaluation measures presented in section 4.3 genes and samples means in biclusters are frequently used. We represent these values as b_{iJ} and b_{Ij} referring to the i row (gene) and j column (sample) means, respectively. Furthermore, the mean of all the expression values in $\mathcal{B}$ is referred to as b_{IJ}.

Note that these definitions above may alter the original authors notations in the contributions reviewed in this chapter.

4.2 Gene Expression Patterns

Several types of biclusters have been described and categorised in the literature, depending on the pattern exhibited by the genes across the experimental conditions (Mukhopadhyay et al (2010)). For some of them it is possible to represent the values in the bicluster using a formal equation.

- **Constant values.** A bicluster with constant values reveals subsets of genes with similar expression values within a subset of conditions. This situation may be expressed by: $b_{ij} = \pi$.
- **Constant values on rows or columns.** A bicluster with constant values in the rows/columns identifies a subset of genes/conditions with similar expression levels across a subset of conditions/genes. Expression levels might therefore vary from genes to genes or from condition to condition. It can also be expressed either in an additive or multiplicative way:
  - Additive: $b_{ij} = \pi + \beta_i$, $b_{ij} = \pi + \beta_j$
  - Multiplicative: $b_{ij} = \pi \times \alpha_i$, $b_{ij} = \pi \times \alpha_j$
- **Coherent values on both rows and columns.** This kind of biclusters identifies more complex relations between genes and conditions, either in an additive or multiplicative way:
  - Additive: $b_{ij} = \pi + \beta_i + \beta_j$

– Multiplicative: $b_{ij} = \pi \times \alpha_i \times \alpha_j$

where π represents any constant value for $\mathcal{B}$, $\beta_i (1 \leq i \leq |I|)$ and $\beta_j (1 \leq j \leq |J|)$ refer to constant values used in additive models for each gene i or condition j; and $\alpha_i, (1 \leq i \leq |I|)$ and $\alpha_j, (1 \leq j \leq |J|)$ correspond to constant values used in multiplicative models for each experimental gene i or condition j.

Other kind of biclusters correspond to those in which their values exhibit *coherent evolutions*, thus showing an evidence that the subset of genes is up-regulated or down-regulated across the subset of conditions without taking into account their actual expression values. In this situation, data in the bicluster cannot be represented by any mathematical model.

4.2.1 Shifting and Scaling Expression Patterns

Aguilar-Ruiz (2005) presented an in-depth discussion on the possible patterns in gene expression data, according to the former definitions. He formally described two kind of patterns: shifting and scaling patterns. They have been defined using numerical relations among the values in a bicluster. Several works based their principle in the pattern concept in order to mine the data.

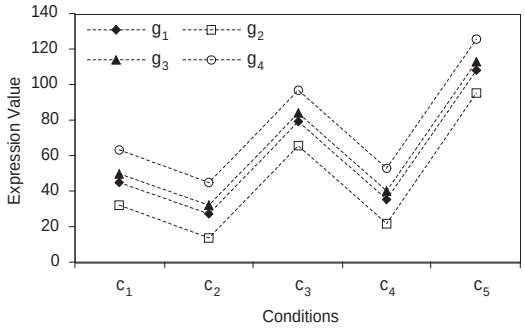
A bicluster $\mathcal{B}$ follows a *perfect shifting pattern* if its values can be obtained by adding a constant-condition number β_j to a typical value for each gene (π_i). β_j is said to be the *shifting coefficient* for condition j. In this case, the expression values in the bicluster fulfil the following equation:

$$b_{ij} = \pi_i + \beta_j \tag{4.1}$$

Graphically, a perfect shifting pattern gives a parallel behaviour of the genes. Figure 4.1 illustrates an example of bicluster presenting a perfect shifting pattern. This bicluster contains four genes g_i (with $1 \leq i \leq 4$) and five conditions c_j (with $1 \leq j \leq 5$). Below the graphic, the expression values (matrix on the left) are provided next to the typical pattern values (π_i) and shifting coefficients (β_j). As we can observe, π_i is constant for each gene (row), while the shifting coefficient β_j is constant for each condition (column).

Similarly, a bicluster follows a *perfect scaling pattern* changing the additive value in the former equation by a multiplicative one. This new term α_j is called the *scaling coefficient*, and represents a constant value for each condition. The following equation defines whether a bicluster follows a perfect scaling pattern or not:

$$b_{ij} = \pi_i \times \alpha_j \tag{4.2}$$



$$\mathcal{B} = \begin{pmatrix} 45 & 27 & 79 & 35 & 108 \\ 32 & 14 & 66 & 22 & 95 \\ 50 & 32 & 84 & 40 & 113 \\ 63 & 45 & 97 & 53 & 126 \end{pmatrix} = \begin{pmatrix} 25+20 & 25+2 & 25+54 & 25+10 & 25+83 \\ 12+20 & 12+2 & 12+54 & 12+10 & 12+83 \\ 30+20 & 30+2 & 30+54 & 30+10 & 30+83 \\ 43+20 & 43+2 & 43+54 & 43+10 & 43+83 \end{pmatrix}$$

$$\begin{array}{rccccc} & \pi_1 & \pi_2 & \pi_3 & \pi_4 & \\ \{\pi_i\} = & \{25 & 12 & 30 & 43\} & \end{array}$$

$$\begin{array}{rccccc} & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 \\ \{\beta_j\} = & \{20 & 2 & 54 & 10 & 83\} \end{array}$$

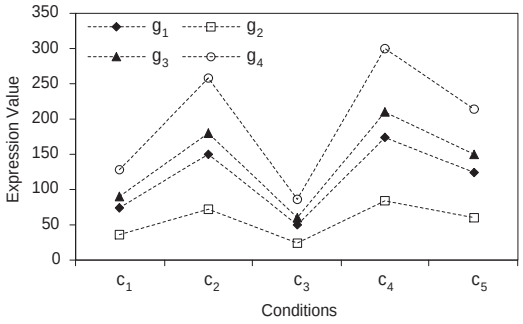
Figure 4.1: Graphical representation of a bicluster containing a perfect shifting pattern.

In Figure 4.2, an example of perfect scaling pattern is displayed. The bicluster contains four genes and five conditions. The expression values are also provided next to the pattern typical values and scaling coefficients, π_i and α_j, respectively. In this case, the genes do not follow a parallel tendency. Although the genes present the same behaviour with regard to the regulation, changes are more abrupt for some genes than for others.

A bicluster may include some of the aforementioned patterns or even both of them, shifting and scaling, at the same time. In fact, it is the most probable case when real data are used. This kind of pattern corresponds to the most general situation that can be described using a mathematical formula, when a bicluster has coherent values on both rows an columns, for the additive and multiplicative model at the same time. When it is the case, it is said that the bicluster follows a *perfect shifting and scaling pattern*, and its values can be represented by this equation:

$$b_{ij} = \pi_i \times \alpha_j + \beta_j \tag{4.3}$$

Observe the example given in Figure 4.3. This bicluster includes simultaneously the perfect patterns shown in Figures 4.1 and 4.2, keeping the same scaling and shifting coefficients. Nevertheless, to visually identify that this bicluster follows a combined pattern is more difficult that to find a single



$$\mathcal{B} = \begin{pmatrix} 75 & 150 & 50 & 175 & 125 \\ 36 & 72 & 24 & 84 & 60 \\ 90 & 180 & 60 & 210 & 150 \\ 129 & 258 & 86 & 301 & 215 \end{pmatrix} = \begin{pmatrix} 25\times 3 & 25\times 6 & 25\times 2 & 25\times 7 & 25\times 5 \\ 12\times 3 & 12\times 6 & 12\times 2 & 12\times 7 & 12\times 5 \\ 30\times 3 & 30\times 6 & 30\times 2 & 30\times 7 & 30\times 5 \\ 43\times 3 & 43\times 6 & 43\times 2 & 43\times 7 & 43\times 5 \end{pmatrix}$$

$$\begin{array}{lcccc} & \pi_1 & \pi_2 & \pi_3 & \pi_4 \\ \{\pi_i\} = & \{25 & 12 & 30 & 43\} \end{array}$$

$$\begin{array}{lccccc} & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 \\ \{\alpha_j\} = & \{3 & 6 & 2 & 7 & 5\} \end{array}$$

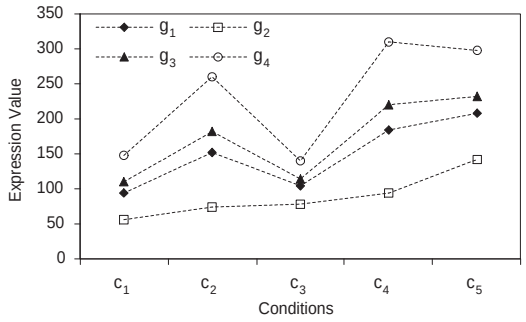
Figure 4.2: Graphical representation of a bicluster containing a perfect scaling pattern.

shifting or scaling pattern, since the effects of one has influence on the other.

At first sight, genes g_1, g_3 and g_4 have similar behaviour, although g_4 differs for the last condition. However, gene g_2 seems independent of the other genes, since it has ascending tendency across every conditions, while the other genes presents a fluctuating behaviour. This fact happens when the shifting coefficients β_i are of the same magnitude that $\pi_j \times \alpha_i$. Note this aspect by observing the second column (gene g_2) of the matrix. It is also interesting that the shifting causes the genes g_1, g_2 and g_3 to significantly change for the last condition with regard to Figure 4.2. Observe that the shifting coefficient for this condition (fifth row of the matrix), that is 83, has the same magnitude that $\pi_j \times \alpha_i$ for such genes. Therefore, when a bicluster includes shifting and scaling patterns simultaneously, identifying it as a good bicluster is a difficult task (Xu et al (2006)).

4.3 Evaluation Measures

Several quality measures for biclusters have been proposed together with different heuristics. Nevertheless, none of the proposed quality measures is able to recognize a perfect shifting and scaling pattern in a bicluster, which is the most general situation and also the most probable when working with



$$\mathcal{B} = \begin{pmatrix} 95 & 152 & 104 & 185 & 208 \\ 56 & 74 & 78 & 94 & 143 \\ 110 & 182 & 114 & 220 & 233 \\ 149 & 260 & 140 & 311 & 298 \end{pmatrix}$$

$$= \begin{pmatrix} 25\times 3+20 & 25\times 6+2 & 25\times 2+54 & 25\times 7+10 & 25\times 5+83 \\ 12\times 3+20 & 12\times 6+2 & 12\times 2+54 & 12\times 7+10 & 12\times 5+83 \\ 30\times 3+20 & 30\times 6+2 & 30\times 2+54 & 30\times 7+10 & 30\times 5+83 \\ 43\times 3+20 & 43\times 6+2 & 43\times 2+54 & 43\times 7+10 & 43\times 5+83 \end{pmatrix}$$

$$\begin{array}{lccccc} & \pi_1 & \pi_2 & \pi_3 & \pi_4 & \\ \{\pi_i\} = & \{25 & 12 & 30 & 43\} & \\ & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 \\ \{\alpha_j\} = & \{3 & 6 & 2 & 7 & 5\} \\ & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 \\ \{\beta_j\} = & \{20 & 2 & 54 & 10 & 83\} \end{array}$$

Figure 4.3: Graphical representation of a bicluster containing perfect shifting and scaling patterns.

gene expression data (Gan et al (2008)). This section presents some of most known different existing evaluation measures for biclusters.

4.3.1 Variance

Hartigan (1972) used bicluster variance in equation 4.4 as a coherence measure, where the goal of his algorithm was to minimize the sum of the bicluster variances.

$$VAR(B) = \sum_{i=1}^{|I|}\sum_{j=1}^{|J|}(b_{ij} - b_{IJ})^2 \tag{4.4}$$

It can be noted that variance only detect constant biclusters. Therefore, biclustering approaches based on variance minimization methods also include other homogeneity criteria to detect other types of biclusters.

4.3.2 Mean Squared Residue

Cheng & Church (2000) were the first in applying biclustering to gene expression data. They introduced one of the most popular biclustering algorithms that combines a greedy search heuristic for finding biclusters with a measure for assessing the quality of such biclusters.

In order to assess the quality of biclusters the algorithm uses the *Mean Squared Residue* (MSR). This measure aims at evaluating the coherence of the genes and conditions of a bicluster B consisting of I rows and J columns. MSR is defined as:

$$MSR(B) = \frac{1}{|I| \cdot |J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} (b_{ij} - b_{iJ} - b_{Ij} + b_{IJ})^2 \tag{4.5}$$

The lower the mean squared residue, the stronger the coherence exhibited by the bicluster, and the better its quality. If a bicluster has a mean squared residue lower than a given value δ, then it is called a δ-bicluster. If a bicluster has a MSR equal to zero, it means that its genes fluctuate in exactly the same way under the subset of experimental conditions, and thus it can be considered a perfect bicluster.

Nevertheless, MSR has been proven to be inefficient for finding certain types of biclusters in microarray data, especially when they present strong scaling tendencies (Aguilar-Ruiz (2005)). In fact, MSR is only able to capture shifting tendencies within the data (Bozdağ et al (2010)).

4.3.3 Scaling Mean Squared Residue

Mukhopadhyay et al (2009b) have recently developed an evaluation measure for biclusters which is able to recognize scaling patterns. In their work, they analyse the reasons why MSR is able to recognise shifting patterns in biclusters but no scaling patterns. Using the mathematical formula for scaling patterns, they define a metric which is then proved to identify scaling patterns. This new measure is named SMSR, from *Scaling MSR*, and it is shown in equation 4.6. Nevertheless, SMSR is not capable of identifying shifting patterns.

$$SMSR(B) = \frac{1}{|I| \cdot |J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} \frac{(b_{iJ} \times b_{Ij} - b_{ij} \times b_{IJ})^2}{b_{iJ}^2 \times b_{Ij}^2} \tag{4.6}$$

4.3.4 Relevance Index

Yip et al (2004) proposed an evaluation metric slightly different from MSR, in which the quality of a bicluster is measured as the sum of the relevance indices of the columns. *Relevance index* R_{Ij} for column $j \in J$ is defined as:

$$R_{Ij} = 1 - \frac{\sigma_{Ij}^2}{\sigma j^2} \tag{4.7}$$

where σ_{Ij}^2 (local variance) and σj^2 (global variance) are the variance of the values in column j for the bicluster and the whole data set, respectively. Note that the relevance index for a column is maximized if its local variance is zero, provided that the global variance is not. Based on this relevance index, the quality of a cluster is measured as the sum of the index values of all the selected conditions.

Due to the nature of the evaluation, the only biclusters patterns that maximize the quality are constant biclusters (either on rows or on columns).

4.3.5 Correlation-based Measures

Correlation coefficients have extensively been used in different kind of microarrays analyses, such as clustering. These statistics measure the overall similarity of the genes without placing any emphasis on the concrete magnitudes, also taking into account negative correlations as well as positive. In this subsection we review the correlation-based approaches which have been proposed for bicluster evaluation.

Pearson's Correlation Coefficient (PCC)

PCC between two variables is defined as the covariance of the two variables divided by the product of their standard deviations. It is a measure of the linear dependence between two variables, and gives a value between $+1$ and -1, both inclusive. A value of 1 implies that a linear equation describes the relationship between the two variables perfectly (positive correlation). A value of -1 implies that all data points lie on a line for which one variable decreases as the other increases (negative correlation). A value of 0 implies that there is no linear correlation between the variables.

PCC is a very effective metric in order to quantify co-regulation between pairs of genes (Bozdağ et al (2009)), and it allows capturing both shifting and scaling patterns that would be separately identified by additive and multiplicative models, respectively. Nevertheless, PCC is not effective for

recognizing constant biclusters or constant row patterns, since these kind of patterns would make the denominator to be zero.

PCC between two rows (genes) $i_1, i_2 \in I$ with respect to the columns (conditions) $j \in J$ is defined as:

$$PCC = \frac{\sum_{j=1}^{|J|}(b_{i_1 j} - b_{i_1 J})(b_{i_2 j} - b_{i_2 J})}{\sqrt{\sum_{j=1}^{|J|}(b_{i_1 j} - b_{i_1 J})^2 \sum_{j=1}^{|J|}(b_{i_2 j} - b_{i_2 J})^2}} \tag{4.8}$$

where $b_{i_1 j}$ and $b_{i_2 j}$ denote the elements in rows i_1, i_2 and column j, and $b_{i_1 J}$, $b_{i_2 J}$ represent the means of rows i_1 and i_2, respectively.

PCC quantifies coherences between pairs of genes. Therefore, in order to measure biclusters coherences, one has to compute all pairwise PCC values between the rows in the same bicluster. Another challenge is that PCC is only meaningful to measure coherence between rows but is too restrictive if it is used to measure coherence between columns simultaneously. Although the PCC has been used as such in some works (Bozdağ et al (2009), Mitra et al (2009), Nepomuceno et al (2011)), in order to overcome this issue, both Yang et al (2011) and Teng & Chan (2008) have defined a PCC-based evaluation measure for the biclusters evaluation in terms of the better correlation, either on rows or columns.

Average Correlation Value (ACV)

ACV was proposed by Teng & Chan (2008) to evaluate the homogeneity of a bicluster or a data matrix in the following way:

$$ACV(\mathcal{B}) = \max \left\{ \frac{\sum_{i_1=1}^{|I|}\sum_{i_2=1}^{|I|} |r\_row_{i_1 i_2}| - |I|}{|I|^2 - |I|}, \frac{\sum_{j_1=1}^{|J|}\sum_{j_2=1}^{|J|} |r\_col_{j_1 j_2}| - |J|}{|J|^2 - |J|} \right\} \tag{4.9}$$

where $r\_row_{i_1 i_2}, r\_col_{j_1 j_2}$ refer to the correlation between any pair of rows i_1, i_2 or columns j_1, j_2, according to the *Pearson coefficient*.

$ACV(\mathcal{B})$ has been proven to be in the interval $[0, 1]$, where a value equal to 1 means that the rows or columns of the bicluster are highly co-expressed, while a low ACV means that neither the conditions nor the genes are similar. Therefore, higher values of ACV are preferred.

The authors proved in their work that ACV always gives the desirable value for both the additive and the multiplicative model, contrary to MSR. They also performed a study on both ACV and MSR behaviours in seven different types of biclusters.

Although ACV is presented as the criterion to evaluate biclusters, it has not been used in order to guide the search in their algorithm. Instead, the algorithm is based on the use of a weighted correlation coefficient, a variation of the one proposed by Bland & Altman (1995), in which weight factors are assigned to the different features (genes or samples) according to their importance. This way, those features with more importance will have more impacts than the others.

Sub-Matrix Correlation Score

Yang et al (2011) used the *Pearson correlation score* as the base to define their measure, assuming that a perfect correlated pattern satisfies perfect linear correlation on row and on column vectors. *Scores correlations* for rows and columns are first defined as in equations 4.10 and 4.11:

$$S_{row} = \min_{i_1 \epsilon I}(S_{i_1 J}),\ S_{i_1 J} = 1 - \frac{\sum_{i_2 \neq i_1, i_2 \epsilon I} |cor(x_{i_1 J}, x_{i_2 J})|}{|I| - 1} \tag{4.10}$$

$$S_{col} = \min_{j_1 \epsilon J}(S_{I j_1}),\ S_{I j_1} = 1 - \frac{\sum_{j_2 \neq j_1, j_2 \epsilon J} |cor(x_{I j_1}, x_{I j_2})|}{|J| - 1} \tag{4.11}$$

where $cor(x_{i_1 J}, x_{i_2 J})$ and $cor(x_{I j_1}, x_{I j_2})$ represent the PCC of any pair of genes or conditions in the bicluster, respectively. S_{row} score reflects the correlation degree on the rows of the bicluster, while S_{col} reflects the correlation degree on columns, being both definitions asymmetric. The *sub-matrix correlation score* is defined as the minimum value of these two scores:

$$S(\mathcal{B}) = \min(S_{row}(I, J), S_{col}(I, J)) \tag{4.12}$$

Since absolute values are used in S_{row} and S_{col} definitions, a perfect correlated bicluster will satisfy $S(I, J) = 0$, meaning that the rows or columns of the bicluster are perfectly linearly correlated. Yang et al (2011) also defined a $\delta - corbicluster$ as a bicluster that satisfies $S(I, J) < \delta$, for some $\delta > 0$.

Average Spearman's Rho (ASR)

ASR was first proposed by Ayadi et al (2009) and is based on the use of *Spearman's rank correlation*, which measures the statistical dependence between two variables, assessing how well their relationship can be described using a monotonic function. Its value varies from -1 to +1, occurring one of these two values when the two variables being compared are monotonically related, even if their relationship is not linear.

Most important difference between Spearman and Pearson correlations is that Pearson assesses linear relationship, while Spearman assesses monotonic relationship. This way, the Spearman correlation is less sensitive to strong outliers that are in the tails of both samples. Nevertheless, when the data are roughly elliptically distributed and there are no prominent outliers, the Spearman and Pearson correlations give similar values.

ASR is computed as in equation 4.13 and outputs a value in the interval $[-1, 1]$, where a high/low value close to 1/-1 indicates that the genes or conditions of the bicluster are strongly correlated, either positively or negatively, respectively.

$$ASR(B) = 2 \times \max \left\{ \frac{\sum_{i \epsilon I} \sum_{j \geq i+1, j \epsilon I} \rho_{ij}}{|I|(|I|-1)}, \frac{\sum_{k \epsilon J} \sum_{l \geq k+1, l \epsilon J} \rho_{kl}}{|J|(|J|-1)} \right\} \tag{4.13}$$

where ρ_{ij}, ρ_{kl} refer to the Spearman correlation between two genes or conditions, respectively.

4.3.6 Similarity Score

Liu & Wang (2007) proposed the use of a *similarity score* between two genes and also a similarity score for a sub-matrix. The first one is used when the reference gene is known in advance. When it is not known, a number of genes might also be randomly selected as the reference genes.

Similarity score between genes

The similarity score between two genes (gene i and a reference gene i^*) under condition j is computed as in equation 4.14:

$$s_{ij} = \begin{cases} 0 & \text{if } d_{ij} > \alpha \cdot d_{avg} \\ 1 - \frac{d_{ij}}{\alpha \cdot d_{avg}} + \beta & \text{otherwise} \end{cases} \tag{4.14}$$

where d_{avg} is defined as the average distance value of all the elements in the expression matrix:

$$d_{avg} = \frac{\sum_{i \epsilon I, j \epsilon J} d_{ij}}{|I||J|} \tag{4.15}$$

and d_{ij} is the absolute value of the expression difference between the gene i and the reference gene i^* for condition j in the expression matrix a:

$$d_{ij} = |a_{ij} - a_{i^* j}| \tag{4.16}$$

$\alpha \cdot d_{avg}$ is used as a threshold to ignore elements with big d_{ij}, in order to find constant biclusters, and β is the bonus for small d_{ij}. This way, β enlarges the similarity score for small d_{ij} and ignores those d_{ij} greater than the threshold.

According to equation 4.14, the value of s_{ij} will always be greater than or equal to 0, being 0 its worst value of similarity.

Similarity score for a bicluster

For each row $i \epsilon I$, the similarity score of row i in $\mathcal{B}$ is:

$$s(i, J) = \sum_{j \epsilon J} s_{ij} \tag{4.17}$$

Similarly, for each column $j \epsilon J$, the similarity score in $\mathcal{B}$ corresponds to:

$$s(I, j) = \sum_{i \epsilon I} s_{ij} \tag{4.18}$$

Using these equations, the similarity score for a bicluster is computed as the minimum value of the similarity scores of both genes and conditions in the bicluster:

$$s(\mathcal{B}) = s(I, J) = \min\{\min_{i \epsilon I}\ s(i, J), \min_{j \epsilon J}\ s(I, j)\} \tag{4.19}$$

The goal when looking for biclusters is to find those sub-matrices with higher values of the similarity score. In order to improve the quality of the output, Liu & Wang (2007) also introduced the *average similarity score* as a second criterion. It consists of the average of all the similarity values of equation 4.14 for all the elements in the bicluster.

Although the type of biclusters found using the similarity score depends on the values for the different thresholds (α, β, and γ for the average), only constant and additive biclusters are recognized.

4.4 Biclustering Approaches Based on Evaluation Measures

As it has already been mentioned, the biclustering problem is NP-hard. This implies that an exhaustive search of the space of solutions may be infeasible. When a measure of quality of the possible solutions is available, the application of a meta-heuristic to solve the problem seems the most appropriate.

Meta-heuristics make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions by iteratively trying to improve a candidate solution with regard to a given quality measure. However, meta-heuristics do not guarantee that optimal solutions are ever found.

Many different meta-heuristics have been used in the biclustering context, and also new variants are continually being proposed, in which meta-heuristics are employed together with other kinds of search techniques. In this section we summarize the most important contributions to the biclustering problem when an evaluation measure is available. We have grouped them according to the type of meta-heuristic in which they are based on. Since MSR was the first quality metric for biclusters, it has been included in many approaches from different authors, although it has been proven not to be the most effective.

4.4.1 Iterative Greedy Search

Direct Clustering

Direct clustering of a data matrix by Hartigan (1972) was one of the first works ever published on biclustering, although it was not applied to genetic data. The algorithm is based on the use of a divide and conquer strategy, in which the input matrix is iteratively partitioned into a set of sub-matrices, until k matrices are obtained, where k is an input parameter for the number of desired biclusters. Although being very fast, the main drawback of this heuristic is that partitions cannot be reconsidered once they have been split. This way, some quality biclusters might be missed due to premature divisions of the data matrix.

Within the partitioning process, variance is used as the evaluation measure (see subsection 4.3.1). At each iteration, those rows or columns that improve the overall partition variance are chosen. This will lead the algorithm towards constant biclusters. Due to the characteristics of the search, overlapping among biclusters is not allowed.

Biclustering of Expression Data by Cheng and Church (CC)

Cheng & Church (2000) were the first in applying biclustering to gene expression data. Their algorithm adopts a sequential covering strategy in order to return a list of n biclusters from an expression data matrix. In order to assess the quality of a bicluster the algorithm makes use of MSR (see subsection 4.3.2).

After preprocessing the missing values of the input data matrix by replacing them with random numbers, the bicluster discovering process is repeated as many times as many biclusters are desired. In each iteration, the bicluster B is initialized to the whole matrix. Next, three different phases for *multiple node deletion*, *single node deletion* and *node addition* are applied. These phases iteratively perform the removal and addition of rows and columns, ensuring that the result is a δ-bicluster. Finally, a *substitution phase* replaces the elements of the input matrix that are contained in the recently found bicluster with random values. This substitution is applied in order to prevent overlapping among biclusters, since it is very unlikely that elements covered by existing biclusters would contribute to any future bicluster. Although this strategy succeeds in avoiding the overlapping, CC presents several drawbacks due to this elements masking and also due to the use of a threshold for rejecting solutions, which is dependent on each database and has to be computed before applying the algorithm.

CC(SMSR)

A similar strategy to that of Cheng and Church has been adopted by Mukhopadhyay et al (2009b) in order to incorporate their evaluation measure SMSR (see subsection 4.3.3) into a search heuristic. This methodology, therefore, shares the same disadvantages with CC, and it is also necessary to stablish a limit value for SMSR for each database. Since SMSR is only able to recognize multiplicative models, the authors propose and adapted algorithm in which CC is applied twice, the fist time using MSR as the evaluation measure and the second time using SMSR. This allows to find biclusters with shifting patterns and also biclusters with scaling patterns, but it does not find biclusters with both kind of patterns simultaneously.

HARP Algorithm

Yip et al (2004) presented a biclustering approach named HARP (*Hierarchical approach with Automatic Relevant dimension selection for Projected clustering*) based on projected clustering. They also introduced an evaluation metric slightly different from MSR named *relevance index* (see subsection 4.3.4).

At the beginning of the algorithm there are as many biclusters as genes. The process consist in iteratively merging biclusters until a certain criterion is met, choosing those experimental conditions that satisfy a specific threshold requirements, taking into account the relevance indexes. Optionally, a reassignation procedure is applied, where biclusters with very few elements are

removed and elements are assigned to the closest bicluster, according to a distance measure.

This algorithm presents several drawbacks, being the most important one the kind of biclusters it is capable to find. Due to the nature of their evaluation measures, the only bicluster patterns that maximize the quality are constant biclusters (either on rows or on columns). Furthermore, the way in which the algorithm works does not allow overlapped elements among biclusters, which is one of the most important differences between clustering and biclustering methodologies.

Maximum Similarity Bicluster Algorithm (MSB)

MSB was proposed by Liu & Wang (2007) together with the similarity score for biclusters reviewed in section 4.3.6. The authors highlighted three different characteristics of their approach: 1) no discretization procedure is required, 2) MSB performs well for overlapping bi-clusters and (3) it works well for additive bi-clusters. This third property is a direct consequence on the use of the similarity score in equation 4.19.

The algorithm starts with the whole matrix as the bicluster. Then a process of iteratively remove the row or column in the bicluster with the worst similarity score is perform, until there is one element left in the bicluster. During this process, $n + m - 1$ sub-matrices have been computed, where n and m refer to the number of rows and columns in the input matrix, respectively. MSB only outputs one bicluster, corresponding to the one in the $n + m - 1$ sub-matrices with the maximum similarity score.

MSB works for the special case of approximately squares biclusters. In order to overcome this issue and also to speed up the process, an extension algorithm named RMSBE (*Randomized MSB Extension algorithm*) is also presented. RMSBE makes use of the average of the similarity scores between some pairs of genes in the bicluster (see equation 4.14), as well as of randomly selection to chose the reference genes.

Biclustering by Iteratively Sorting with Weighted Coefficients

In their approach, Teng & Chan (2008) alternately sort and transpose the gene expression data, using weighted correlations at the same time to measure gene and condition similarities. The weighted correlation index is a variation of *Pearson correlation coefficient*, which was originally adapted by Bland & Altman (1995) in order to add weights when working with multiple observations. In this work, Teng & Chan (2008) have redefined this index so that weights are assigned to the different features (genes or samples) accord-

ing to their importance. This way, those features with more importance will have more impacts than the others.

The algorithm is based on the dominant set approach work of Pavan & Pelillo (2003). In order to find a bicluster, genes and conditions are iteratively sorted using weight vectors, which are also iteratively refined using the sorting vector of the previous iteration. In each iteration the matrix is transposed and the process is repeated over the other dimension, thus alternating from genes to conditions. At the end of the process, the highly correlated bicluster is located at one corner of the rearranged gene expression data matrix, and is defined using a threshold for the correlation coefficients between adjacent rows and columns.

To find more than one bicluster, the authors use the weight vectors. This way, any time a bicluster is found, the weights of those features that have not been included in it are enhanced, at the same time as reducing the weights of those features included in it. Using this approach, overlapping among biclusters is permitted but controlled and penalized any time a gene or condition is included in a solution.

4.4.2 Stochastic Iterative Greedy Search

Some authors have preferred using a stochastic strategy in order to add a random component to the search, rending thus the algorithm to be non-deterministic. Most important stochastic iterative greedy approaches are reviewed in this section.

FLexible Overlapped biClustering (FLOC)

Yang et al (2005) proposed a different heuristic for coping with the random masking of the values in the data matrix. To address this issue and to further accelerate the biclustering process, the authors presented a new model of bicluster to incorporate null values. They also proposed an algorithm named FLOC (*FLexible Overlapped biClustering*) able to discover a set of k possibly overlapping biclusters simultaneously based on probabilistic moves.

The algorithm begins with the creation of k initial biclusters with rows and columns added to them according to a given probability. After that, these biclusters are iteratively improved by the addition or removal of one row or column at a time, determining the action that better improves the average of the MSR values of the k biclusters. Bicluster volumes are also taken into account within the possible actions, where bigger biclusters are preferred, and the variance is used to reject constant biclusters. The whole process ends when no action that improves the overall quality can be found.

Random Walk Biclustering (RWB)

Angiulli et al (2008) presented a biclustering algorithm based on a greedy technique enriched with a local search strategy to escape poor local minima. Their algorithm makes use of a gain function that combines three different objectives: MSR, gene variance and the size of the bicluster. These objectives are compensated by user-provided weights.

RWB produces one bicluster at a time. Starting with an initial random solution, it searches for a locally optimal solution by successive transformations that improve the gain function. A transformation is done only if there is reduction of MSR or an increase either in the gene variance or the volume. In order to avoid getting trapped into poor local minima, the algorithm executes random moves according to a probability given by the user. To obtain k biclusters RWB is executed k times by controlling the degree of overlapping among the solutions. This degree is controlled for genes and conditions independently by using two different frequency thresholds. This way, during any of the k executions of the algorithm, whenever a gene or condition exceeds the corresponding frequency threshold, it is removed from the matrix and therefore it will not be taken into account in the subsequent executions.

Reactive GRASP Biclustering

GRASP is a multi-start meta-heuristic for combinatorial problems, consisting of iterations made up of two phases: construction of a greedy randomized solution and local search in which its neighbourhood is investigated until a local minimum is found. The best overall solution is kept as the result. *Reactive GRASP* is variant of GRASP in which a the threshold parameter α associated to the candidate list of solutions is self-tuned, and its value is periodically modified according with the quality of the solutions recently obtained.

Dharan & Nair (2009) proposed a reactive GRASP biclustering method in which high quality bicluster seeds are generated using one-dimensional k-means clustering. Afterwards, these seeds are further enlarged by adding more rows and columns to them, employing the reactive GRASP method, and also making use of randomized heuristics for escaping from local optima. The algorithm makes use of MSR score as the cost function to evaluate the quality of the obtained biclusters. In order to obtain k biclusters, k seeds need to be generated in the first step, where their overlapping amount is controlled by the setting of the α values.

Pattern-Driven Neighbourhood Search (PDNS)

Neighbourhood search consist of iteratively improving an initial candidate solution by replacing it with a higher quality neighbour. It is usually obtained by performing little modifications on the former one.

PDNS has been recently proposed by Ayadi et al (2012) as a *Pattern-Driven Neighbourhood Search* approach for the biclustering problem. Prior to the search, the method first applies a preprocessing step to transform the input data matrix into a behaviour matrix M'. Each row of this behaviour matrix represents the trajectory pattern of a gene across all the combined conditions, while each column represents the trajectory pattern of all the genes under a pair of particular conditions in the data matrix. M' defines the problem search space as well as the neighbourhoods used within the search process.

The initial bicluster is obtained by using two fast well-known greedy algorithms: CC (see 4.4.1) and OPSM (see 4.5.5) and is encoded into its behaviour matrix before being improved by PDNS. The algorithm alternates between two basic components: a descent-based improvement procedure and a perturbation operator. The descendent strategy is used to explore the neighbourhood, moving to an improving neighbouring solution at each iteration. This process is employed to discover locally optimal solutions. In order to displace the search to a new starting points, the perturbation operator is applied. It is carried out after the descent improvement stops according to one of two stopping criteria: the solution reaches a fixes quality threshold or a fixed number of iterations has been reached. A perturbed bicluster is then computed by a randomly replacement of 10 per cent of genes and conditions of the recorded best bicluster so far. This perturbed bicluster is used as a new starting point for the next round of the descent search, and the whole PDNS algorithm stops when the best bicluster is not updated for a fixed number of perturbations. For assessing the quality of two biclusters any time a replacement is taking place the ASR evaluation function is used (see 4.3.5).

The algorithm outputs one bicluster at a time. Therefore, in order to obtain several biclusters it must be run several times with different initial solutions. In this work, the authors use the output of two fast well-known algorithm as initial biclusters. Nevertheless, no overlapping control is carried out among the reported solutions.

4.4.3 Nature-inspired Meta-heuristics

Nature-inspired meta-heuristics are characterized by reproducing efficient behaviours observed in the nature. Examples of this kind of heuristics include evolutionary computation, artificial immune systems, ants colony optimization or swarm optimization, among others. All of them make use of algorithmic operators simulating useful aspects of various natural phenomena and have been proven to be very effective for complex optimization problems. In this context, many biclustering approaches have been proposed based on the use of any of this kind of meta-heuristics, being evolutionary computation the most used. For more information on this technology, chapter 5 of this PhD Thesis has been entirely dedicated to evolutionary computation. In the following, the most important biclustering approaches based on any nature-inspired meta-heuristic have been review.

Simulated Annealing Biclustering

Simulated Annealing (SA) stochastic technique originally developed to model the natural process of crystallisation and which has been adopted to solve optimization problems (Kirkpatrick et al (1983)). SA algorithm iteratively replaces the current solution by a neighbour one if accepted, according to a probability that depends on both the fitness difference and a global parameter called the temperature. This temperature is gradually decreased during the process, decreasing the probability of randomly choosing the new solution when it gets lower.

The specific behaviour of any simulated annealing approach is mainly given by the fitness function an the depth of the search at each temperature. In Bryan et al (2006) approach, the fitness of each solution is given by its MSR value, and ten times the number of genes successes needed to be achieved before cooling. This number determines the depth of the search, being a success an improvement on the fitness function. The initial temperature of the system, as well as the rate at which it is lowered are also important, since both of them determine the number of total iterations and also affect the convergence. The authors set them experimentally.

The algorithm must be run k times in order to obtain k biclusters. Solutions are masked in the original matrix in order to avoid overlap among them. In their work, Bryan et al (2006) used the same method of Cheng & Church (2000), replacing the original values for random ones, in an attempt to prevent them to be part of any further bicluster.

Crowding distance based Multi-objective Particle Swarm Optimization Biclustering (CMOPSOB)

Particle Swarm Optimization (PSO) technique simulates the social behaviour of a flock of birds or school of fishes which aim to find food. The system is initialized with a population of random solutions and searches for optima by updating generations. Potential solutions are called particles and fly through the problem space by following the current optimum. Particles have memory for retaining part of their previous state, and decide the next movement influenced by to randomly weighted factors affecting the best particle previous position and the best neighbourhood previous position. The global optimal solution is the best location obtained so far by any particle in the population and the process end when each particle reaches its local, neighbourhood and global best positions.

Liu et al (2009) based their biclustering approach on the use of a PSO together with crowding distance as the nearest neighbour search strategy, which speed up the convergence to the Pareto front and also guarantee diversity of solutions. Three different objectives are used in CMOPSOB: the bicluster size, gene variance and MSR, which have been incorporated into a multi-objective environment based on the individuals dominance. For more information on multi-objective optimization see section 5.4.1. Being a populational approach, several potential solutions are taken into account in each generation. Those non-dominated solutions in the last generation will be reported as the output.

Multi-objective Multi-population Artificial Immune Network (MOM-aiNet)

Inspired by biological immune systems, *Artificial Immune Systems* (AIS) have emerged as computational paradigms that apply immunological principles to problem solving in a wide range of areas. Coelho et al (2009) presented an immune-inspired algorithm for biclustering based on the concepts of clonal selection and immune network theories adopted in the original aiNet algorithm by de Castro & Von Zuben (2001). It is basically constituted by sequences of cloning, mutation, selection and suppression steps.

MOM-aiNet explores a multi-population aspect, by evolving several subpopulations that will be stimulated to explore distinct regions of the search space. After an initialization procedure consisting on random individuals made up of just one row and one column, populations are evolved by cloning and mutating their individuals. The authors apply three different kind of mutations: insert one row, insert one column or remove one element, either row

or column. They only consider two different objectives: MSR and the volume, using the dominance among individuals in order to replace individuals with higher quality ones. A suppression step is periodically performed for the removal of individuals with high affinity (overlap), causing thus fluctuations in the populations sizes. Nevertheless, this step is followed by an insertion one, in which new individuals are created, giving preference to those rows or columns that do not belong to any bicluster. After a predefined number of iterations, MOM-aiNet returns all the non-dominated individuals within each sub-population.

Evolutionary Algorithms for Biclustering

Evolutionary Algorithms are based on the theory of evolution and natural selection. Being the oldest of the nature-inspired meta-heuristics, they have been broadly applied to solve problems in many fields of engineering and science. Many biclustering approaches have been proposed based on evolutionary algorithms. Being a populational approach, a larger subset of the whole space of solutions is explored, at the same time that it helps them to avoid becoming trapped at a local optimum. These reasons make evolutionary algorithms very suited to the biclustering problem.

Starting by an initial population, evolutionary algorithms select some individuals and recombine them to generate a new population of individuals. This process is repeated for a number of generations until the algorithm converges or certain criterion criteria is met. For more information on evolutionary algorithms we refer the reader to chapter 5. Here, we summarise the most relevant biclustering approaches based on evolutionary algorithms, both single or multi-objective. Although the majority of them aim at optimise the popular metric residue (MSR), some of them are also based on correlation coefficients.

Bleuler et al (2004) were the first in developing an evolutionary biclustering algorithm. They proposed the use of binary strings for the individuals representation, and an initialization of random solutions but uniformly distributed according to their sizes. Bit mutation and uniform crossover are used as reproduction operators, and a fitness function that prioritises MSR. For those solutions with MSR below the threshold δ, bigger biclusters sizes are preferred. Tournament has been used as selection mechanism, where populations are completely replaced by new offspring. A diversity maintenance strategy is carried out which decreases the amount of overlapping among bicluster, and CC algorithm is also applied as a local search mainly to increase the size of the individuals. At the end, the whole population of individuals is returned as the set of quality biclusters.

SEBI was presented by Divina & Aguilar-Ruiz (2006) as a *Sequential Evolutionary BIclustering* approach. The term sequential refers the way in which bicluster are discovered, being only one bicluster obtained per each run of the evolutionary algorithm. In order to obtain several biclusters, a sequential strategy is adopted, invoking several times the evolutionary process. Furthermore, a *matrix of weights* is used for the control of overlapped elements among the different solutions. This weight matrix is initialized with zero values and is updated every time a bicluster is returned. Individuals consists of bit strings, and are initialized randomly but containing just one element. Together with tournament selection, three different crossover and mutation operators are used with equal probability in reproduction: one-point, two-points and uniform crossovers, and mutations that respectively add a row or a column to the bicluster or the standard mutation. Elitism is also applied in order to preserve the best individual through generations. Individual evaluations are carried out by a single fitness function in which four different objectives have been put together: MSR, row variance, bicluster size and an overlapping penalty based on the weight matrix.

BiHEA (*Biclustering via a Hybrid Evolutionary Algorithm*) was proposed by Gallo et al (2009) and is very similar to the evolutionary biclustering algorithm of Bleuler et al (2004). Both of them perform a local search based on CC algorithm, and return the set of individuals in the last population as the output. However, they differ in the crossover operator (BiHEA uses two-point crossover) and BiHEA also incorporates gene variance in the fitness function. Furthermore, two additional mechanisms are also added in order to improve the quality of the solutions. The first one is elitism, in which a predefined number of best biclusters are directly passed to next generation, with the sole condition that they do not get over a certain amount of overlap. The second one makes use of an external archive to keep the best generated biclusters through the entire evolutionary process, trying to avoid the misplacement of good solutions through generations.

More recently, Huang et al (2012) have proposed a new biclustering algorithm based on the use of an EA together with hierarchical clustering. The authors argue that with such a huge search space, the EA itself should not be able to find optimal or approximately optimal solutions within a reasonable time. Therefore, they propose to separate the conditions into a number of conditions subsets, also called *subspaces*. The evolutionary algorithm is then applied to each subspace in parallel, and a expanding and merging phase is finally employed to combine the subspaces results into the output biclusters. As it is related only to the condition dimension, the EA is called CBEB, from *Condition-Based Evolutionary Biclustering*, where the normalized geometric selection method is used as the selection function and the simple crossover

and binary mutation methods are employed for reproducing the offspring. In both CBEB and the expanding and merging phase MSR score has been used for the evaluation of the potential solutions, always using the predefined threshold δ as the upper limit.

Multi-objective Evolutionary Algorithms for Biclustering

Apart from the specific homogeneity measure for biclusters, many authors have also incorporated other objectives to the search, such as the bicluster volume or gene variance. These requirements are often conflicting. For example, a the bigger a bicluster is more probable to have a higher MSR value. Nevertheless, bigger biclusters with low MSR values are preferred. The four EA approaches review above make use of a single aggregate objective function which combines the different objectives. In this section we review those EA approaches that optimize the different objectives according to other multi-objectives schemes (see section 5.4.1 for more information on this multi-objective EAs).

In the work of Mitra & Hayashi (2006), the first approach that implements a *Multi-Objective EA* (MOEA) based on Pareto dominance is presented. The authors based their work on the NSGA-II (Deb et al (2000)), and look for biclusters with maximum size and MSR value, as long as it is smaller than the upper bound δ. Also, a local search strategy based on the *node insertion* and *node deletion* phases of CC algorithm is applied to all of the individuals at the beginning of every generational loop. Populations are ranked according to the dominance criterion (see section 5.4.1), crowding tournament selection is performed, the selected individuals are crossed and mutated, and the best individuals among the new and old populations are selected to remain in the next generation.

In MOGAB (*Multi-Objective GA-based Biclustering algorithm*) (Mukhopadhyay et al (2009a)), the authors propose the use of a new individual representation, encoded as strings made up of two parts: in the first one indexes of genes acting as clusters centres of sets of genes are represented, while the second one keeps the indexes of the conditions acting as cluster centres of sets of conditions. This way, each individual does not represent a bicluster, but a set of biclusters, obtained with the different possible combinations of clusters of genes and conditions. The initial population contains randomly generated individuals, where each gene or condition is equally probable to become the centre for a gene or a condition cluster, respectively. MOGAB is also based on the NSGA-II strategy, and performs crowded binary tournament selection, single-point crossover and standard mutation, although these two last operators are carried out on gene and condition centres strings in-

dependently, and invalid individuals are marked when appear in order not to let them reproduce in the next generation. Elitism has also been incorporated in MOGAB to track the non-dominated Pareto-optimal solutions. Within the evaluation, MSR and row variances are computed for all the $\delta-$biclusters denoted by each individual. A fitness vector is afterwards created with the mean of their fitness. From the non-dominated individuals in the final population, all the $\delta-$biclusters are extracted and output as the final biclusters.

4.4.4 SVD and Clustering-based Approaches

Possibilistic Spectral Biclustering (PSB)

A biclustering approach based on the use of *Singular Value Decomposition* (SVD) together with one-dimensional clustering named PSB was proposed by Cano et al (2007). The use of SVD techniques enhances the clustering process by performing dimensionality reduction.

This algorithm consists of firstly apply SVD method on an eigenproblem formulated on the input matrix and get $\min(n, m)$ solution eigenvectors, where n and m refers to the number of genes and conditions in the input matrix. Using these eigenvectors several partition matrices are created, to which two independent clustering algorithms are executed: for those rows representing genes and for those representing conditions in the original matrix, respectively. Each combination of a cluster of genes and a cluster of conditions is a possibilistic bicluster, which will be post-processed in order to improve its quality when possible or be rejected if it is not considered a quality solution. Finally, the whole process is repeated with a linear inversion of the input expression matrix, in order to also obtain under-expressed genes.

The clustering algorithm used in PSB is a variation of the *Improved Possibilistic Clustering* (IPC) by Zhang & Leung (2004), which mixes possibilistic and probabilistic approaches. MSR is used at both the clustering and the crisping of the possibilistic biclusters, though in this last step the volume is also taken into account. Possibilistic clustering allows a considerable amount of overlapping, so the authors have also added to the process an overlapping control, in which a bicluster is checked for its overlapping amount before being added to the result set.

Biclustering with SVD and Hierarchical Clustering

Yang et al (2011) have recently proposed a strategy similar to the one of Cano et al (2007), by using *Singular Value Decomposition* (SVD) together with

clustering and a final stage to merge and filter the clusters. In this approach, the authors make use of the *sub-matrix correlation score* (see 4.3.5) also presented in their work. A upper bound δ is used to defined a $\delta-$corbicluster as a bicluster with a sub-matrix correlation score lower than δ.

In a first step, two different matrices named $R^{(l)}$ (a group of basis genes) and $C^{(l)}$ (a group of basis conditions) are obtained by using SVD. Secondly, after centralizing the rows of these two matrices, clustering is applied to the both of them by the *Mixed Clustering algorithm*, based on agglomerative hierarchical clustering and on the use of the sub-matrix correlation score as dissimilarity measure. The number of clusters produced by this technique is not known beforehand. This way, a set of m and n groups of clusters are obtained from both matrices. Every pair of these groups constructs a bicluster, obtaining $m \times n$ biclusters in this way. Nevertheless, not every pair of groups may be a $\delta-$corbicluster, and a final step is executed in order to obtain inclusion-maximal biclusters. This last step is carried out by the *Lift algorithm*, inspired in the *node-deletion* and *node-addition* phases proposed by Cheng and Church, but according to the sub-matrix correlation score. Since the clustering algorithm generates not mutually exclusive clusters, the biclusters obtained by this method are possible overlapped.

4.5 Non Metric-based Biclustering Algorithms

In this section we review the most important biclustering approaches that exclude the use of any evaluation measure for guiding the search. We have classified them according to their most relevant characteristic: specific algorithm, data structure representation or the main important basis. Note that different categories are not exclusive. Although we have grouped the algorithms attending to what we have considered to be their most distinctive property, some of them may as well be assigned to more than one group.

4.5.1 Graph-based Approaches

SAMBA

Tanay et al (2002) based their approach on graph theoretic coupled with statistical modelling of the data, where SAMBA stands for *Statistical-Algorithmic Method for Bicluster Analysis*. In their work, they model the input expression data as a bipartite graph whose two parts correspond to conditions and genes, respectively, and edges refer to significant expression changes. The vertex pairs in the graph are assigned weights according to a probabilistic

model, so that heavy sub-graphs correspond to biclusters with high likelihood. Furthermore, they present two statistical models of the resulting graph, the second one being a refined version of the first in order to include the direction of expression change, allowing thus to detect either up or down regulation. Weights are assigned to the vertex pairs according to each model so that heavy sub-graphs correspond to significant biclusters. This way, discovering the most significant biclusters means finding the heaviest sub-graphs in the model bipartite graph, where the weight of a sub-graph is the sum of the weights of the gene-condition pairs in it. In order to cope with noisy data, Tanay et al (2002) searched for sub-graphs that are not necessarily complete, assigning negative weights to non-edges.

In order to reduce the complexity of the problem, high-degree genes are filtered out, depending on a pre-defined threshold. According to the authors, the number of genes is reduced by around 20 per cent, considering it to be a modest reduction. The proposed algorithm is an iterative polynomial approach based on the procedure for solving the maximum bounded bi-clique problem, where a hash-table is used for the identification the heaviest bi-clique. A generalization of this method is applied in order to give the k heaviest non-redundant sub-graphs, where k is an input parameter. Before applying the algorithm, the graph structure may be created, using the signed or unsigned model depending on the input data.

Bimax

Bimax was presented by Prelić et al (2006) as a fast divide-and-conquer algorithm capable of finding all maximal bi-cliques in a corresponding graph-based matrix representation, where the graph representation of the data is similar to the one used by Tanay et al (2002). In this case, Bimax uses an underlying binary data model which assumes two possible expression levels per gene. Therefore, as a preprocessing phase, it is compulsory to discretize the expression values to binary values at a specific threshold and with a specific scheme. All values above the threshold will be set to one, all those below to zero. The discretization scheme defines if only down or up-regulated genes (or both) will be considered.

The binarized matrix is regarded as an adjacency matrix of a graph. By exploiting the fact that the graph induced by the matrix is bipartite, an incremental algorithm can be tailored to this application. Prelić et al (2006) proposed a fast divide and conquer approach, the *Binary Inclusion-MAXimal biclustering algorithm* (Bimax), which first partitions the data matrix into three sub-matrices, one of which contains only 0-cells and will be disregarded. After that, the algorithm is recursively applied to the remaining tow sub-

matrices, ending when the current matrix only contains 1s, representing thus a bicluster. At each step, the two partitioned matrices may have elements in common or not, allowing thus the possibility of finding overlapped biclusters.

MicroCluster

MicroCluster was developed by Zhao & Zaki (2005) as a biclustering method for mining maximal biclusters satisfying certain homogeneity criteria, with possible overlapped regions. Biclusters with shifting patterns are detected by using exponential transformations. Furthermore, by means of these kind of transformations, scaling patterns may also be detected.

MicroCluster uses an enumeration method consisting of three steps. Firstly, a weighted, directed range multi-graph is created for representing the possible valid ratio ranges among experimental conditions and the genes that meet those ranges. A valid ratio range is an interval of ratio values satisfying several constraints on expression values. In this graph, vertices correspond to samples and each edge has an associated gene set corresponding to the range on that edge. The construction of this range multi-graph filters out most unrelated data. Once the multi-graph is created, a second step is applied for mining the maximal clusters from it, based on a recursive depth-frist search. Although the output of this step is the final set of biclusters, a final step is optionally executed in order to delete or merge those biclusters according to several overlap conditions. This last step is also applied to deal with noise in the data, controlling the noise tolerance.

QUBIC

QUBIC has been more recently presented as a *QUalitative BIClustering algorithm* (Li et al (2009)), in which the input data matrix is first represented as a matrix of integer values, either in a qualitative or semi-qualitative manner. In this representation, two genes are considered to be correlated under a subset of conditions if the corresponding integer values along the two corresponding rows of the matrix are identical. The qualitative (or semi-qualitative) representation is such that allows the algorithm to detect different kind of biclusters, also including scaling patterns. It also suitable for finding both positively and negatively correlated expression patterns, where negative correlations will be represented by opposite signs across the entire row. The biclustering problem consist now in finding all the optimal correlated submatrices.

The first step of the algorithm correspond to the construction of a weighted graph from the (semi-)qualitative matrix, with genes represented as ver-

tices, and edges connecting every pair of genes. Edge weights are computed in the base of the similarity level between the two corresponding rows. After the graph has been created, biclusters are identified one by one, starting for each bicluster with the heaviest unused edge as a seed. This seed is used to build an initial bicluster and the algorithm iteratively adds additional genes into the current solution. The consistency level marks the end of the search for a bicluster, since it determines the minimum ratio between the number of identical non-zero integers in a column and the total number of rows in the sub-matrix.

4.5.2 One-way Clustering-based Approaches

Coupled Two-way Clustering

Coupled Two-Way clustering (CTWC), introduced by Getz et al (2000) defines a generic scheme for transforming a one-dimensional clustering algorithm into a biclustering algorithm. They define a bicluster as a pair of subsets of genes and conditions, or a pair of gene and conditions clusters. They also define a stable cluster as a cluster that is statistically significant according to some criterion (such as stability, critical size, or the criterion used by Hartigan (1975)). Getz et al (2000) also applied a normalization step based on euclidean distance as a previous step to the application of the algorithm.

The heuristic provided by CTWC consist in an iterative process restricting the possible candidates for the subsets of genes and samples, only considering and testing those gene and sample clusters previously identified as stable clusters. The iterative process is initialized with the full matrix. Both sets of genes and samples are used to perform two-way clustering, storing the resulting stable clusters in one of two registers of stable clusters (for genes or samples). Pointers that identify parent clusters are also stored, consisting thus in a hierarchical approach. These steps are iterated further, using pairs of all previously found clusters, and making sure that every pair is treated only once. The process is terminated when no new clusters that satisfy some criteria are found.

The success of this strategy depends on the performance of the one-dimensional clustering algorithm. According to the authors, CTWC can be performed with any clustering algorithm. Nevertheless, many popular clustering algorithms (e.g. K-means, Hierarchical, SOM) cannot be used in this approach, since they do not readily distinguish significant clusters from non-significant clusters or make a-priori assumption on the number of clusters (Tanay et al (2002)). Getz et al (2000) recommend the use of the

SPC (superparamagnetic clustering algorithm), which is especially suitable for gene microarray data analysis due to its robustness against noise and its ability to identify stable clusters.

Interrelated Two-way Clustering

Interrelated Two-Way Clustering (ITWC) developed by Tang & Zhang (2005) is an algorithm similar to CTWC, combining the results of one-way clustering on both dimensions separately. A pre-processing step based on row normalization is also applied, where rows with little variation are removed from the input matrix. Although correlation coefficient are used as similarity measure to measure the strength of the linear relationship between two rows or two columns in the process, Tang & Zhang (2005) do not propose any quality metric for the evaluation of a sub-matrix as a whole. For this reason we have categorized this approach as a non-metric based.

The idea in ITWC is to discover the relationships between gene and sample clusters while iteratively clustering through both dimensions to extract important genes and classify samples simultaneously. Within each iteration there are five main steps. First step performs clustering on rows, while in the second step clustering is performed in the column dimension, for each group of genes from step one. In this second step, only two clusters are obtained for each gene group. Third step combines the former steps results by computing diverse sets intersections for the sample groups, resulting in 2^k sample groups, where k is the number of gene clusters obtained in step one. Fourth step aims at finding heterogeneous groups of conditions (do not share rows used for clustering), being the result of this step is a set of highly disjoint biclusters. Last step of ITWC sorts the rows in descending order of the cosine distance between each row and a row representative of each bicluster. After that, only the first one third of rows is kept, reducing thus the row set for each heterogeneous group. A likelihood based on the correlation coefficient is calculated for each heterogeneous group, being the reduced genes of the group with the higher likelihood value the selected for the next iteration. These genes and the entire samples then form a new gene expression matrix from which a new iteration starts. Iterations will be terminated when a stable and significant pattern of samples has emerged. For this purpose, the authors use a criterion based on a coefficient of variation to measure how internally-similar and well-separated the partition is.

Biclusters identified by ITWC have no elements in common, due to the search strategy. This way, overlapping among biclusters is not allowed in this approach.

4.5.3 Probabilistic Models

Plaid Models

Lazzeroni & Owen (2002) proposed *plaid models*, a tool for exploratory analysis of multivariate data. In this approach, the genes-condition matrix is represented as a superposition of layers, corresponding to biclusters. Several versions of the model are described in their work, being the most general the one in equation 4.20, which allows a gene to be in more than one bicluster or in none at all.

$$Y_{ij} \doteq \sum_{k=0}^{K} \theta_{ijk}\rho_{ik}\kappa_{jk} \tag{4.20}$$

where Y_{ij} refers to the expression level of gene i under sample j in the input matrix, K is the number of biclusters, θ_{ij0} describes the background layer and θ_{ijk} represents four different types of models, depending on the types of biclusters (overlapped, exclusive ...). Each $\rho_{ik}\epsilon\{0,1\}$ is 1 if gene i is in the k'th bicluster, zero otherwise. Similarly, each $\kappa_{jk}\epsilon\{0,1\}$ is 1 if sample j is in the k'th bicluster, zero otherwise. Using this equation, a bicluster is assumed to be the sum of a bicluster background level plus row-specific and column-specific constants.

In order to find k biclusters in the data, Lazzeroni & Owen (2002) proposed a greedy algorithm that adds one layer at a time. The process seeks for a plaid model minimizing the sum of squared errors when approximating the data matrix to the model. For this purpose, an iterative approach is adopted with each cycle updating θ values, ρ values and κ values in turns. Assuming that residual data becomes more and more unstructured noise as each layer is removed from the data, authors propose a simple rule for stopping the process, in which only a small number of extra layers can be extracted once the data have been reduced to noise.

Rich Probabilistic Models

Rich probabilistic models was proposed by Segal et al (2001) for studying relations between expression, regulatory motifs and gene annotations. The main advantage of this approach is the use of additional types of information in the analysis, such us functional roles or cellular location for genes. In the case of samples, the information would depend on the type of arrays used. It might be the treatment applied to the sample or growth conditions among others.

Starting with the input array and the available additional information, a predictive model is learnt from the data. The approach is based on the language of *Probabilistic Relational Models* (PRMs) that extend Bayesian networks to a relational setting, also allowing the inclusion of multiple types of information. Furthermore, the *Expectation Maximization* (EM) algorithm is used for parameter estimation with incomplete data.

The outcome of the algorithm can be interpreted as a collection of disjoint biclusters generated in a supervised manner, where overlapped is not allowed.

Gibbs Sampling

A biclustering algorithm based on the use of *Gibbs sampling* was proposed by Sheng et al (2003), where a simple frequency model is adopted for representing the expression pattern of a bicluster. According to the authors, using Gibbs sampling as the method for parameter estimation avoids the problem of local minima that often characterizes expectation maximization. In order to achieve a better performance of Gibbs sampling, microrray data is first discretized, resembling thus the problem of finding sub-sequences sharing similar alphabetic expressions.

Background model has been introduced as a single multinomial distribution. This is suitable for data sets where the genes share the same distribution under every condition. If this is not the case, several multinomial distributions might be used, each of which describing the background under an individual condition.

The probabilistic model adopted considers only the presence of a single bicluster in the data set. In order to discover more than one bicluster an iterative process is carried out, masking the genes selected for the recently found bicluster and running the algorithm on the rest of the data. This masking strategy prevent genes from appearing in several biclustering, avoiding thus the possibility of overlap.

Bayesian Biclustering Model (BBC)

Gu & Liu (2008) developed a *Bayesian Biclustering Model* (BBC), also based on the use of a Gibbs sampling procedure to make the Bayesian inference of biclusters. For a single bicluster, the same model as in the plaid model (Lazzeroni & Owen (2002)) is assumed. Nevertheless, for multiple biclusters, the amount of overlap among the biclusters in the original plaid model was too high. In order to overcome this situation, in the BBC model overlap is only allowed in one direction, either gene or condition. This means that if overlapped among genes is permitted, then any experimental condition would

only appear in at most one bicluster, and vice-versa. Also, biclustering is not exhaustive in this approach, since any element (gene or condition) can belong to either one or none of the found biclusters.

BBC works on normalized microarray data, although this step is not included in the algorithm. The authors conducted a study on how different normalization methods affect the performance of their algorithm, showing that the model is stable for two normalization methods developed by themselves, the *Interquartile Range Normalization* (IQRN) and the *Smallest Quartile Range Normalization* (SQRN), being both of them inspired in column standardization.

According to the authors, missing data can be easily handled by just treating them as additional unknown variables. Furthermore, other types of information might be incorporated into the model in order to improve the search.

Conserved Gene Expression Motifs (xMOTIFs)

Murali & Kasif (2003) proposed the use of xMOTIFs (*conserved gene expression Motifs*) for the representation of gene expression data, where a xMOTIF is a subset of genes that is simultaneously conserved across a subset of samples, and a gene expression level is conserved across a set of samples if it is in the same state in each of the samples in the subset. Therefore, for each gene, a list of intervals representing the states in which the gene is expressed in the samples is required. In order to prevent the algorithm from finding too small or too large xMOTIFs, some constraints on their size, conservation and maximality have been added to its formal definition.

A probabilistic algorithm that exploits the mathematical structure of xMOTIFs to compute the largest xMOTIF was also developed by Murali & Kasif (2003). In order to identify several xMOTIFs in the data, an iterative strategy has been adopted, where samples satisfying each xMOTIF are removed from the data, and the new largest xMOTIF is searched. This process continues until all samples satisfy some xMOTIF. This search strategy allows gene overlap and also sample overlap, whenever any sample does not take part in more than one xMOFIT with the same gene.

cMonkey

cMonkey (David J. Reiss & Bonneau (2006)) is a biclustering algorithm specifically designed for genetic data, which integrates sequence data, gene expression data and gene network association data. cMonkey is model-based, where distribution variables are parametrized using simple statistical distri-

butions, being more robust to varying data size and quality. Each of the individual data types are modelled, using logistic regression to integrate them into a joint model.

Biclusters are modelled using a Markov chain process, in which an initial bicluster seed is iteratively optimized by updating its state based upon conditional probability distributions, also applying a simulated annealing procedure. Biclusters are optimized sequentially, starting with a new bicluster seed a predefined number of times. Seeds are initialized only with genes that have not been previously placed into any former bicluster, although in the subsequent iterations those genes can still be added to new biclusters. This way overlapping among biclusters is allowed but controlled, since a filtered set of biclusters is finally computed in order to reduce redundancy.

4.5.4 Linear Algebra

Spectral Biclustering

Spectral biclustering was especially designed by Kluger et al (2003) for analysing microarray cancer datasets. In this context, it is assumed that the expression matrix has a hidden checkerboard-like structure with blocks of high-expression levels and low-expression levels. Kluger et al (2003) approach consist in finding these distinctive checkerboard patterns, by using eigenvectors and commonly used linear algebra approaches, such as the *Singular Value Decomposition* (SVD). Furthermore, normalization steps have also been integrated into the search, in order to put the genes on the same scale so that they have the same average level of expression across conditions, and likewise for the conditions.

Biclusters found by spectral biclustering have no elements in common, forbidding thus any kind of overlap among them. Moreover, this is an exhaustive approach, meaning that every gene and every condition will be included in one bicluster.

Iterative Signature Algorithm

Iterative Signature Algorithm (ISA) was proposed by Bergmann et al (2003) and provides a definition of biclusters as *transcription modules* to be retrieved from the expression data. A transcription module consist of a set of coregulated genes and the set of experimental conditions under which this coregulation is the most stringent. Its size depends on the associated set of two thresholds that determine the similarity between the genes and conditions of the module, respectively.

In order to find transcription modules in the data, the signature algorithm consisting in a generalization of *Singular Value Decomposition* (SVD) is applied. The algorithm starts with a set of randomly selected genes or conditions, iteratively refining the genes and conditions until they match the definition of a transcription module. The method includes data normalization and the use of thresholds that determine the resolutions of the different transcription modules. In this search one bicluster is produced at each iteration. Initial seeds are randomly chosen without any overlap restriction, therefore, different biclusters may contain overlapped genes and/or conditions.

Non-smooth Non-negative Matrix Factorization

A method based on the application of the *non-smooth non-Negative Matrix Factorization* (nsNMF) technique for discovering local structures from gene expression datasets was developed by Carmona-Saez et al (2006). This approach consists in a variant of the classical NMF model, which adds non-smoothness constraints in order to produce more compact and localized feature representation of the data.

As well as NMF, nsNMF approximates the original matrix as a product of two sub-matrices, where the columns of the first one are basis experiments (also called factors) and the rows of the second constitute basis genes (or encoding vectors). Each factor determines a local gene expression feature or gene module, while each encoding vectors determine the set of experimental conditions highly associated to these modules. This way, biclusters are determined by the corresponding pairs of columns and rows of both matrices. Therefore, overlap among biclusters is not allowed in this approach.

4.5.5 Optimal Reordering of Rows and Columns

OPSM

Ben-Dor et al (2003) define biclusters to be order-preserving sub-matrices (OPSMs), in which the expression levels of all genes induce the same linear ordering of the experiments. This way, a sub-matrix is said to be order-preserving if there is a permutation of its columns under which the sequence of values in every row is strictly increasing. This strict condition might be relaxed for real expression data, where rows having a significant tendency to be similarly ordered are searched for instead. This relaxation introduces a new probability which will influence the generation of the probabilistic model used.

Guided by the probabilistic model, an efficient algorithm is also proposed

for finding the hidden OPSM in the data. It consists of an iterative greedy heuristic algorithm based on the concepts of partial and complete models. Since finding the best model would be infeasible, their approach consists of growing *partial models* iteratively, and trying to converge to the best complete model. At the beginning, all partial models of first order are generated, picking afterwards the best ones and computing for each of them the possible extensions to partial models of the next order. This process is successively repeated until the models of the top order are reached, returning the best of them as the complete model. The algorithm can also be used to discover more than one OPSM in the same dataset, even when they are overlapped.

OREO

OREO was proposed by DiMaggio et al (2008) as an approach based on the *Optimal RE-Ordering* of the rows and columns of a data matrix so as to globally minimize a dissimilarity metric. Contrary to OPSM, this approach allow for monotonicity violations in the reordering, but penalize their contributions according to a selected objective function.

Two rows or columns are defined to be adjacent if the second one is directly below the first in the final arrangement. Using this definition, the final ordering may be represented by a matrix of binary 0-1 variables for each dimension. The optimal rearrangement is obtained using a metric of similarity together with one of the two proposed problem formulations. Although the objective function might be defined by the user, the authors propose three different choices: the relative difference, the squared difference or a metric similar to the root-mean squared deviation. All of them only applied for the values of adjacent elements.

The selected objective function would guide either a network flow model or a *Travelling Salesman Problem* (TSP) approach in order to obtain the optimal rearrangement of rows and columns. In both models, two different elements (rows or columns) are connected if they are adjacent, although in the case of TSP they are weighted edges, where the weight is computed using the objective function. Furthermore, both approaches require the use of additional constraints to avoid cyclic arrangements.

The algorithm begins by optimally re-ordering a single dimension of the data matrix. After that, the median is computed for each pair of adjacent elements (either rows or columns), where the top 10 percent of largest median values define the cluster boundaries between the re-ordered elements. These cluster boundaries are then used to partition the original matrix into several sub-matrices. Finally, the other dimension of each sub-matrix is re-ordered and clusters in this dimension are again defined using the median value of

the objective function between neighbouring elements in the final ordering.

4.6 Biological Validation for Biclusters

Section 3.2.4 introduced the most common information sources for microarray data analysis validation and interpretation, such as KEGG (*Kyoto Encyclopedia of Genes and Genomes*) or GO (*Gene Ontology*). Although KEGG database has been used when external relationships such as biological pathways are involved in the study, the biological knowledge used in bicluster validations are mostly gene annotations from GO.

The GO project (Ashburner et al (2000)) is a initiative to unify the representation of gene and gene product attributes across all species. It is a directed acyclic graph whose nodes represent terms dealing with molecular functions, cell components or biological processes, and edges connecting nodes depict dependency relationships. Gene Ontology has been widely used in genome research applications, and also for the validation of results obtained after a microarray analysis process, such as clustering or biclustering.

Typical validation of a bicluster $\mathcal{B}$ consists in getting all GO terms annotated to any of the genes in $\mathcal{B}$ and then apply a statistical significance test to determine if each term appearance is relevant. *Term-for-Term* (TFT) analysis represents the standard method of performing statistical analysis for over-representation in GO, although other approaches for term enrichment include the *parent-child method* of Grossmann et al (2007), topology based methods as described in Alexa et al (2006) and Falcon & Gentleman (2007). Also, a new model-based approach is described in Bauer et al (2010).

In TFT analysis, starting from a subset of genes (study group) from a larger population (whole set of genes in the microarray), the interest resides in knowing if the frequency of an annotation to a Gene Ontology term is relevant for the study group compared to the overall population. Fisher's exact test is the most commonly used test for this purpose, together with the Bonferroni multiple test correction. This correction is advisable to be performed since Fisher's test is applied to many terms per study group, although it can also be performed using other methodologies, such as Westfall-Young or Benjamini-Hochberg, among others. After that, a Bonferroni adjusted p-value is obtained for each GO term for which genes in the study group are involved, where study groups correspond to the sets of genes in each bicluster. Depending on the desired confidence level, which determines the adjusted p-value, a bicluster is said to be significantly enriched if there exists at least one GO term for which genes in the bicluster are significantly annotated.

Results of bicluster biological validation using GO vary depending on the biclusters sizes. In fact, GO terms are organized in levels of the graph according, among other issues, to their specificity (Alterovitz et al (2007)). Terms in higher levels (nearer to the root of the graph) are considered to be more generic and have a greater number of genes annotated, while terms in lower levels of the graph are more specific and might have only a few genes annotated. For this reasons, when working with big sets of genes, it would be more probable that they will be enriched for more generic GO terms (higher in the graph structure).

One common issue in hierarchical ontologies is deciding the level of specificity to use in the analysis (Soldatova & King (2005)). On the one hand, GO terms that are too general may overlook significantly represented biological markers because many genes in the background genome are also annotated by the general GO terms. On the other hand, GO terms that are too specific can result in the same problem, since too few genes in the microarray are annotated by these GO terms.

Although this kind of validation is very useful for the objective of knowledge discovery, the mayor disadvantage of biological significance as a validation method is that biological knowledge is not complete. This way, when a bicluster does not group known GO annotations, it may be because it is a bad bicluster, or because GO annotations are not complete.

4.7 Summary

This chapter has been dedicated to the study of the different existing approaches for biclustering gene expression data. Biclustering consist of a variant of the popular clustering technique, which allows the user to obtain more and better information from a two-dimensional data structure, such as the obtained from a microarray experiment. The goal is to discover functionally related gene sets under different subsets of experimental conditions. Biclustering has been proven to be a much more complex problem than clustering, the reason for which the majority of existing solutions are based on the use of stochastic techniques. Furthermore, some approaches are also based on the use of evaluation measures for quantifying the quality of the obtained solutions.

First two sections of this chapter present an unified notation for bicluster representation, and a description of the different kind of expression patterns which biclustering algorithms aim at finding in their solutions. Among all of them, the combined shifting and scaling pattern has been proven to correspond to the more general situation, where any of the other pattern may

be represented by the combined one. Third section reviews the different existing evaluation measures for biclusters in the literature, together with their capabilities of detecting the distinct types of expression patterns. The availability of a suitable quality metric for bicluster is essential not only for guiding the search, but also for establishing comparison criteria among the results obtained by different biclustering techniques. Section four and five survey most important existing biclustering algorithms, based or not on the use of evaluation measures within the search, respectively. In both sections they have been classified according to the type of meta-heuristic in which they have been based on. Finally, last section describes the most common biological validation process used in the community, consisting in an statistical analysis based on the biological information within the *Gene Ontology* (GO) database.

Chapter 5

Evolutionary Computation

Evolutionary Computation (EC) is a programming technique that mimics biological evolution as a problem-solving strategy. It is mostly applied for optimization problems in which the search space is very large. Within the biclustering context, EC strategies have been widely applied, as it can be seen in section 4.4.3, where various bicluster metrics have been taking into account for the potential solutions evaluation in different approaches. EC has been applied in the context of this PhD Thesis in order to test the efficiency of our bicluster metrics and also to develop a customizable biclustering algorithm.

In this chapter we first present the common characteristics of the different EC strategies, emphasising individuals encoding and initialisation, evaluation, selection and reproduction. Afterwards, we study the most important EC paradigms, which share a set of common features but differ in the evaluation and representation strategies. Three different evolution models which can be adopted in any EC paradigm are also presented in section 5.3. Finally, last section of this chapter is dedicated to multi-objective and memetic evolutionary techniques, being both of them the most important modern trend arisen from the canonical EC.

5.1 Common Features of EC

EC is a search technique which uses computational models of processes of evolution and selection. Concepts and mechanisms of Darwinian evolution and natural selection (Darwin (1888)) are encoded in *Evolutionary Algorithms* (EAs) and used to solve problems in many fields of engineering and science. They are mostly applied for optimization problems, where a fitness measure defined by the environment is used in order to evaluate the different individual solutions. EAs are specially useful for problems where the search

space is very large or for multi-modal problems in which traditional heuristics get trapped in local optima.

Terminology used in EC borrows a lot from genetics, evolutionary theory and cellular biology. Thus, a candidate solution to a problem is called an *individual*, and a set of solutions is called a *population*. *Genome*, *genotype* or *chromosome* terms are used to refer to the encoding of an individual, while the *phenotype* describes the observable characteristics of the individual, according to the problem under study. Each genotype consist of a sequence of *genes*, corresponding to the attributes that describe an individual, and the specific value of a gene is called an *allele*. Individuals produce new candidate solutions by *breeding offspring* or *children*. The evaluation of candidate solutions consists of the assignation of a grade named *fitness*, which indicates the quality of the solution in the context of a given problem. New *generations* replace one population after another, until some stopping criterion is met. At the end of the process, the best individual of the last population is considered to be the optimal found solution to the problem. This entire process of search is called *evolution*.

Algorithm 1: A general scheme of an EA

```
1 Initialize the population;
2 Evaluate all members of the population;
3 while stopping criterion is not satisfied do
4 |   Select individual(s) in the population to be parent(s);
5 |   Create new individual(s) by applying the genetic operators to the
  |   copies of the parent(s);
6 |   Evaluate new individuals;
7 |   Replace some/all individuals in the current population with the new
  |   ones;
8 Extract solution form population;
```

A general scheme of an EA is presented in Algorithm 1. It starts by generating an initial set of candidate solutions for a given problem. These individuals are evaluated using problem-dependent metrics which provide a fitness for each candidate solution. Subsequently, offspring is produced by altering the existing solutions, where fittest solutions often have a higher probability of being selected for reproduction. Depending on the evolution strategy, offspring might be added to the existing population or replace it entirely. In both tactics, some of the worst solutions are deleted before iterating this whole process (Steps 4-7 in Algorithm 1), starting from selection for reproduction. When a given stop criteria is met, the iterations end and

the best individual(s) are returned. Stopping criteria is usually related to a significant improvement on the solutions through generations combined with a maximum number of iterations.

In order to simulate this Darwinian evolution process based on natural selection, four fundamental factors are needed to be taken into account: 1) individual encoding and initialization, 2) evaluation, 3) selection and 4) reproduction processes. All these processes work together in a framework like the one in Algorithm 1. Following subsections detail different strategies for the encoding and initialization, evaluation, selection and reproduction of individuals.

5.1.1 Individual Encoding and Initialization

Encoding refers to the individuals inner representation in the algorithm. It is usually referred to as genome, genotype or chromosome. Holland (1975) emphasized the importance of a universal genetic representation in order to focus on a single problem independent set of reproduction operators. Holland also suggested a binary string representation, or *binary encoding*, where alleles are either zero or one and the mapping to the corresponding phenotype is left unspecified and problem specific.

Although initially binary encoding was the most extended codification strategy, it is often inappropriate for many problems and has been therefore extended to non-binary representations for the application of EAs to a wider range of problems. This way, other kind of codifications allow genotypes to be more problem dependent. Successful EAs have used integer or real string individuals (Goldberg (1989)), or even more general representations such as tree and matrix structures (Michalewicz (1998)). Although the encoding does not alter evolution strategy, specialized genetic operators have been devised to handle the different kind of encodings.

Initial population strategy is essential in every EA, since depending on the adopted strategy, the algorithm may converge to different solutions. Also, a suitable initial population strategy can even speed up the convergence (Toğan & Daloğlu (2008)). Traditionally, the initial population is created following a totally random strategy, but several other initialization techniques have also been used. For example, random initialization has also been combined with the use of specific information on the problem in order to create a better distributed initial population in the search space. Other approaches start from a set of previously known or arbitrarily assumed solutions.

5.1.2 Evaluation

The evaluation of the individuals in each population is carried out in order to provide a fitness value for each candidate solution. This fitness quantifies how close an individual is to achieve the set of aims defined by a specific problem. This way, potential solutions are evaluated using problem dependent metrics. Individuals fitness is also taken into account in the selection of the parents for the next population, as it can be seen in next section. This way, the fitness function is in charge of guiding the algorithm towards optimal or near-optimal solutions.

Designing an adequate fitness function is also an essential and not trivial task since it must not only make the algorithm converge to an appropriate solution, but also must be computed quickly. Nevertheless, the fact that individuals in the population represent entire problem solutions frequently simplifies the design of the fitness evaluation process (De Jong & Sarma (1993)). A typical EA must be iterated many times until reach a reasonable result for any complex problem. Therefore, the speed of execution of all the processes within each iteration is crucial, including individual evaluations.

5.1.3 Selection

Using the fitness score, the selection mechanism chooses a subset of the current population as parents to create new individuals. As it can be seen in Algorithm 1, there exist two different stages in which selection is applied:

- In line 4 of the algorithm, in order to choose the set of parents to produce offspring.
- In line 7 of the algorithm, in order to determine which of the new individuals are going to be incorporated in the new population, as well as to decide the individuals that remains in it.

Both steps involve selecting a subset of individuals from a given set. There are several commonly used selection strategies in EC applications, being the majority of them based on the use of the fitness. Stochastic selection mechanisms are also preferred, as a way to add noise to the search, decreasing the likelihood of converging to sub-optimal solutions and improving the robustness of the algorithm.

Most popular selection mechanisms include fitness-proportional preferences. In this sense, *roulette selection* (Słowik & Białko (2004)) normalizes the fitness values of all individuals in the population and assigns these normalized values as probabilities that their respective individuals will be selected. *Rank selection* (Kuosmanen et al (1994)) works by first ranking all

individuals in the population by their fitness, and use these ranks, rather than actual fitness values, to determine selection probabilities of the individuals. Another popular selection strategy is *tournament selection* (Miller & Goldberg (1995)). In this strategy, a pool of individuals is picked at random from the population, where each of the individuals in the pool is selected independently. Tournament selection returns the fittest individual from the pool, where the pool size is a parameter that controls the magnitude of the selection pressure. Finally, *truncation selection* (Crow & Kimura (1979)) is a deterministic mechanism which chooses only a certain proportion of the best individuals in the population. For more information on these alternatives, Blickle & Thiele (1996) perform in their work a comparative analysis of the different selection mechanisms used in evolutionary algorithms.

Selection is just one the components that interact in the EA, therefore, there exists no best combination of selection mechanisms to achieve optimal EA performance. For problems that exhibit highly multi-modal fitness landscapes or landscapes that change over time, too much exploitation generally results in premature convergence to suboptimal peaks in the space. Conversely, performance on relatively smooth, time-invariant landscapes with a small number of peaks can be enhanced by increasing exploitation. Since selection mechanisms mostly affect exploitation, it is possible to adjust it by switching from one kind of selector to others.

5.1.4 Reproduction: Genetic Operators

Once the set of parents has been selected, the new individuals are created by copying them and applying genetic operators. The newly created individuals are evaluated and assigned fitness values.

The two most popular genetic operators are *mutation* and *recombination* or *crossover*. Mutation acts on a single individual and works by applying some variation to one or more genes in the individual chromosome. The amount of variation is controlled by specifying how many genes are to be modified and the manner in which genes are to be modified, although these two aspects depend on the individuals representation. For example, bit strings representations use the bit-flip mutation, while real-valued vectors use the Gaussian mutation.

Crossover, on the other hand, operates on multiple individuals (usually two) and combines parts of these individuals to create new ones. Again, recombination operator is representation dependent. Fixed-length linear genome representations traditionally take the form of crossover operators, in which the crossover points (usually one or two) mark the linear subsegments on the parents genomes. For these kinds of reproductive operators,

the amount of variation introduced when producing children is dependent on two factors: how many crossover points there are and how similar the parents are to each other. With real-valued vector it is also possible to use a recombination operator that averages parents alleles.

Reproductive operators introduce most of the exploration in a EA. Therefore, significant changes in performance can be obtained as well by switching to different reproductive operators. Also, EAs using strong selection pressure generally counterbalance that with more explorative reproductive operators, while EAs that use weaker forms of selection pressure use much less explorative reproductive operators.

5.2 EC Paradigms

There is a great variety of EAs that have been proposed and studied. They all share a common set of underlying assumptions but differ in the evolution strategy to be used and representation on which EAs operate. Thus, different choices on selection, reproduction, evaluation and representation issues can significantly alter EAs behaviour and its performance. Much of the EA research in the 1970s allowed to gain insight into these issues, giving rise to three distinct EAs paradigms: *Evolution Strategies* (ES), *Evolutionary Programming* (EP) and *Genetic Algorithms* (GAs). These algorithms have been mostly used to evolve solutions to parametrized problem domains. On the other hand, *Genetic Programming* (GP) has been used to evolve actual computer programs to solve a number of computational tasks. There are also many hybrid models incorporating various features of the above paradigms, including the CHC algorithm of Eshelman (1991), the *structured GA* of Dasgupta & McGregor (1992), the *breeder GA* of Mühlenbein & Schlierkamp-Voosen (1993) or the *messy GA* of Goldberg et al (1989).

Although there exist several documented differences among these paradigms, the distinction between them is not always so straightforward, and many more methods developed for a particular paradigm are also being adopted by the other ones. Next subsections explain the main variations among the four first mentioned archetypes.

5.2.1 Evolution Strategies

The focus of the *Evolution Strategy* (ES) paradigm was on real-valued function optimization, being individual representation based on vectors of real numbers. Furthermore, reproduction was initially purely asexual (single parent with a mutation operator), where 1 parent produced λ offspring and

the fittest of the $1 + \lambda$ individuals was selected to be the single parent for the next generation of offspring. In this context asexual reproduction took the form of mutating one or more of the parent gene values (real numbers) via a normally distributed perturbation. Nevertheless, after several empirical studies mutation operators co-evolved in response to the characteristics of the function being optimized. Also, nowadays ES also use crossover for individual reproduction.

In their work, Beyer & Schwefel (2002) survey the history of evolution strategies which dates back to the 1960s, discussing also the possible future branches of ES research.

5.2.2 Evolutionary Programming

Evolutionary Programming (EP) was first introduced in 1966 by Fogel et al (1966) for developing finite state automata for solving specific problems. The framework originally presented continues to be refined and expanded today and has been applied to a wide variety of problems beyond the evolution of finite state machines. Also, nowadays EP is often used to evolve individuals consisting of real-valued vectors.

Initial work had proposed both asexual reproduction and sexual reproduction (two parents combining to form offspring via a recombination operator). Since the individuals being evolved were finite state machines, mutation took the form of adding/deleting states/arcs, and recombination required combining pieces of two finite state machines into a new one. Nevertheless, experimental studied proved the difficulty of defining an effective recombination operator in this context, but reported impressive results in evolving useful finite state machines with only asexual reproduction and mutation.

5.2.3 Genetic Algorithms

Genetic Algorithms (GAs) were introduced by Holland (1975) and focus was on developing more application-independent algorithms. Initial approach was based on the use of fixed-length binary string representation for individuals. Nowadays, other kind of representations are used, such as real-valued strings. Mutation took the form of a *bit flip* with a fixed probability, and recombination took the form of a 1-point crossover operator, in which a randomly selected initial sequence of genes from one parent is combined with the remaining sequence of genes from a second parent to produce offspring with features of both parents. Nowadays, other kind of mutation and crossover operations are also adopted. In this scenario, GAs typically rely on crossover,

while mutation is considered a minor operator, applied with very low probability.

Most of the early studies involved generational GAs, in which a fixed-size population of N parents produce a new population of N offspring, which unconditionally replaces the parent population regardless of fitness. However, nowadays other types of evolution strategies are also used. Furthermore, parents are stochastically selected to produce offspring in proportion to their fitness. Thus, fittest individuals will have more probably of passing genetic information on to the next generation.

5.2.4 Genetic Programming

Genetic Programming (GP) (Koza (1992)) has been used to evolve actual computer programs to solve a number of computational tasks. An early example of this can be seen in Friedberg (1958), which describes the design and implementation of a *Learning Machine* that evolved sets of machine language instructions over time.

GP can be seen as a specialization of genetic algorithms where each individual represent a computer program, consisting of both data structures and functions applied to those data structures.

5.3 Evolution Models

EAs may adopt different evolution models. Within the generational change process, if the entire population is replaced by the new individuals then the algorithm is called *generational EA*. On the other hand, if only one individual is replaced then the algorithm is called a *steady-state EA*. *Generational gap EA* represents a halfway model in which a subset of the population is replaced from one generation to the next, where the gap represents the number of replaced individuals and is usually a percentage of the population size.

5.3.1 Generational EAs

Generational EAs represent an extreme case of replacement methods, although it continues to be the most extended within the community. Generational EAs were introduced by Holland (1975), including two variants of reproductive plans. In both circumstances, the entire population is replaced every generation.

The first plan, named R_1, maintains a fixed size population and at each time step a single individual is selected probabilistically to produce a single

offspring. To make room for this new offspring, one individual from the current population is selected for deletion via a uniform random distribution. In the second plan, R_d, at each time step all individuals are deterministically selected to produce their expected number of offspring in a temporary storage location and, when that process is completed, the offspring produced replaces the entire current population. Although both plans are equivalent from a theoretical point of view, R_d approach is being favoured due to practical problems of genetic drift in small populations and also at recalculating selection probabilities.

Generational processes are clearly non-overlapping population systems, where the lifetime of each individual in the population is only one generation. Furthermore, there is no guarantee of preservation of the best solution. In order to overcome this last situation, elitism is usually incorporated in the generational change, allowing thus the preservation of the best individual of each population. Elitism has been proven to be essential for the search of the global optimum of any optimization problem (De Jong & Sarma (1993)). Regarding the first issue, overlapping may be implicit in the generational change depending on the crossover and mutation probabilities. Thus, as an experimental study carried out by De Jong shows, using a crossover and mutation probability of 0.6 and 0.4 respectively, around 40% of the offspring are clones of their ancestors.

5.3.2 Steady State EAs

Steady state EAs were born in the context of neural network training domain. It consists of a one-at-time reproduction scheme, where only one or two new individuals are incorporated into the population in each stage. The offspring of the individuals selected from each generation go back into the pre-existing gene pool, replacing some of the less fit members of the previous generation. This way, some individuals are retained between generations, and parents and offspring co-exist. Also, elitism is usually implicit in this strategy. In general, larger populations are required in steady state systems in order to guarantee genetic diversity.

Selection and replacement criteria are essential in these kind of algorithms, since they can significantly vary the system behaviour, in terms of convergence and diversity. For example, replacement of the less fit individuals induces early convergence, while a random replacement leads to slow convergence.

GENITOR (Whitley et al (1989)) is a successful implementation of a steady state scenario, where selection is performed using rankings and the less fit individual(s) re-emplacement is carried out.

5.3.3 Generational Gap EAs

Generation EAs and steady state EAs represent two extreme cases of replacement methods. As a halfway strategy, generational gap EA introduces a parameter that controls the fraction of the population that is replaced at each generation.

De Jong & Sarma (1993) studied the pros/cons of overlapping generations with generational gap EAs, concluding that increasing the gap reduces the individuals variance, suggesting also avoiding small gap values together with small populations. Also, De Jong remarks that the important behavioural changes in any EA are due to changes in the exploration/exploitation balance, depending on the different selection and deletion strategies.

Cantú-Paz (2000) presented in his work a study on the selection intensity for common selection and replacement methods in GAs with generation gaps. The selection intensity is the normalized increase of the average fitness of the population after selection, and it can be used to predict the number of steps until the convergence to a unique solution. Main conclusion states that algorithms with small generation gaps experience a faster convergence.

5.4 Advanced Evolutionary Algorithms

Various modern trends in EC have arisen in order to relax some of the assumptions of the canonical EA. Some of these trends are *multi-objective* EAs, where several independent fitness criteria are taken into account, or *memetic* EAs, where a separate local search process is applied to refine individuals. In this section, both multi-objective and memetic EAs are detailed. Other EAs variations include *parallel* or *distributed* EAs, in which evolution occurs in parallel sub-populations independently. Also, *hybrid* methods combine EAs with other methodologies to solve constrained problems.

5.4.1 Multi-Objective EAs (MOEAs)

Multi-objective problems are characterized by having two or more, usually conflicting, objectives. The main difference from single-objective optimization is that multi-objective problems do not have one single optimal solution, but they instead have a set of optimal solutions, where each one represents a trade-off between the different objectives. A reasonable solution to a multi-objective problem is to investigate a set of solutions, each of which satisfies the objectives at an acceptable level without being dominated by any other solution. This set of solutions is called the Pareto front, and is made up of non-dominated individuals. An individual x is said to dominate another

individual y if x is not worst than y on all objectives, and x is better than y on at least one objective.

The use of EAs to solve problems of this nature has been motivated mainly because of the population-based nature of EAs, which allows the generation of several elements of the Pareto optimal set in a single run. Additionally, they are significantly more robust, compared to the classical methods, particularly when issues like the shape or continuity of the Pareto front are a matter of concern.

There are two major goals of multi-objective optimization. Firstly, the output consists of a large number of Pareto-optimal solutions to a given problem. Secondly, the solutions to the problem should be widely differentiated. Most significant progress in multi-objective EAs came with the non-dominated sorting procedure by Goldberg (1989). Since that time, many researchers have developed various versions of multi-objective optimization algorithms. Most important methods are summarized below:

- *Single Aggregate Objective Function* (AOF) consist in the combination of the individual objective functions into a single composite function. For this purpose, methods such as utility theory or weighted sum method are used. The major drawbacks of this method include difficulties in determining the appropriate weights and the fact that improper Pareto solutions may be generated (Coello et al (2006)). Nevertheless, using an AOF it is possible to specify the relative relevance of each objective in the individuals evaluation.

- VEGA was the first multi-objective GA, called *Vector Evaluated GA*, proposed by Schaffer (1985). VEGA was mainly aimed for solving problems in machine learning, and its simple unconstrained two-objective functions became the usual test suite to validate most of the evolutionary multi-objective optimization techniques developed during several of the following years.

  VEGA approach is an example of a criterion or objective selection technique where a fraction of each succeeding populations is selected based on separate objective performance, where a vector composed by the k objective functions is used. The specific objectives for each fraction are randomly selected at each generation. The main advantage of the alternating objectives approach is easy to implement and computationally as efficient as a single-objective GA. The major drawback of objective switching is that the population tends to converge to solutions which are superior for one objective, but poor at others.

- *Target Vector Approaches* try to minimize the difference between the current solution generated and a defined vector of desirable goals. Most popular target vector proposals are hybrids with: *Goal Programming* (Charnes & Cooper (1957)), *Goal Attainment* (Chen & Liu (1994)), and the *Min-max Algorithm*. This last method, the weighted min–max, has been used by Hajela & Lin (1992) to optimize a 10-bar plane truss in which weight and displacement were to be minimized, and by Carlos et al (1998) to optimize I-beams and truss designs (Coello & Christiansen (2000)).

  Target vector approaches require the definition of goals to be achieved, where the computation of these goals normally requires some extra computational effort and can lead to additional problems, such as misleading selection pressure. Also, the goals must be defined in the feasible domain in order to yield a non-dominated solution. This fact limits their applicability. In spite of these drawbacks, there are certain problems in which target vector approaches can provide very good approximations of the Pareto optimal set.

- *Multi-Objective Genetic Algorithm* (MOGA) was proposed by Fonseca et al (1993), in which the rank of a certain individual corresponds to the number of chromosomes in the current population by which it is dominated. All non-dominated MOGA individuals are assigned rank 1, while dominated ones are penalized according to the population density of the corresponding region of the trade-off surface. MOGA has been used in many engineering design applications, including for example a gas turbine controller (Chipperfield & Fleming (1995)) and supersonic wings (Obayashi (2002)).

  The type of blocked fitness assignment used in MOGA is likely to produce a large selection pressure that might produce premature convergence (Ghosh & Tsutsui (2003)). To avoid that, Fonseca and Fleming used a niche-formation method to distribute the population over the Pareto-optimal region. Main disadvantages of the use of MOGA are its slow convergence and problems related to the niche size parameter. This parameter is used for a niche-formation method that distributes the population over the Pareto-optimal region.

- *Non-dominated Sorting Genetic Algorithm* (NSGA) was defined by Srinivas & Deb (1994) and was based on Goldberg notion of non-dominated sorting with a niche and speciation method. NSGA also classifies the population into non-dominated fronts. Then, a dummy fitness value is assigned to each front using a fitness sharing function

such that the worst fitness value assigned to every front is better than the best fitness value assigned to the next one. Since individuals in the first front have the maximum fitness value, they always get more copies than the rest of the population. This results in a fast convergence of the population.

The first, very popular elitist genetic algorithm for multi-objective optimization was NSGA-II, an improved version of NSGA created by Deb et al (2000). It uses the crowding distance in its selection operator to keep a diverse front by making sure each member stays a crowding distance apart. This keeps the population diverse and helps the algorithm to explore the fitness landscape.

- Horn et al (1994) proposed a tournament selection scheme based on Pareto dominance defined as the *Niched-Pareto Genetic Algorithm*(NPGA). Selection works in the following way: two individuals randomly chosen are compared against a subset (randomly chosen) from the population. If one of them is dominated by the individuals in the subset and the other is not, then the non-dominated individual wins. When both competitors are either dominated or non-dominated, the result of the tournament is decided through fitness sharing.

- *Strength Pareto Evolutionary Algorithm* (SPEA) was proposed by Zitzler et al (1998) and integrates ideas from various existing evolutionary multi-objective optimization, adding also some new elements. SPEA uses an external archive containing non-dominated solutions previously found. At each generation, non-dominated individuals are copied to the external set, computing an strength value for each individual in the set. SPEA also uses clustering to truncate external population as a mechanism to maintain diversity.

5.4.2 Memetic EAs

Memetic Algorithms (MAs) (Moscato (1989)) are EAs that apply a separate local search process to refine individuals. These methods are inspired by models of adaptation in natural systems that combine evolutionary adaptation of populations with individuals learning within a lifetime. MAs are also known as an hybridization of an EA with local search. This local search is generally applied as in the Algorithm 2:

EAs are typically very efficient at the exploration of the search space but not as efficient in exploitation. The main idea of MAs is to incorporate a local search process to perform exploitation, as well as to reduce the likelihood

Algorithm 2: Local search scheme

```
1 Start with an initial solution x;
2 Generate a set of neighbour solutions of x;
3 while best solution in the neighbourhood set is better than x do
4 |  replace x with the best solution;
5 Return x;
```

of the premature convergence. Furthermore, MAs are intrinsically concerned with exploiting all available knowledge about the problem under study. The exploitation of this problem-knowledge can be accomplished by incorporating heuristics, approximation algorithms, local search techniques, specialized recombination operators, truncated exact methods, etc.

The local improver process in Algorithm 2 can be used in different parts of the generation process. For example, it can be performed over the initial population individuals, at the end of each generation or after the utilization of any other recombination or mutation operator. It can even be applied to only certain generations or to a representative subset of individuals.

Multi-objective MAs

Local search operations in multi-objective EAs can also provide good performance by exploring limit regions in objective space, moving towards specific regions on the Pareto front. The combination of these two kind of algorithms is generally defined as *Multi-Objective MAs* or *memetic MOEAs*.

In hybridization of MOEAs with local search algorithms, important issues are selecting the solution(s) to apply the local search and identifying a solution in the neighbourhood as the new best solution when multiple non-dominated local solutions exist. Several approaches have been proposed to address these two issues as follows, including optimizing only one objective, applying local search to only final solutions or using the weighted sum of the objective functions in the local search.

Various memetic MOEAs can be developed by incorporating a specific local search technique within a known MOEA. Examples include MOGLS (*Multi-Objective Genetic Local Search*) of Ishibuchi & Murata (1996) or Bosman & de Jong (2006) approaches, among others variations of general MOEA global/local search algorithms. For a historical review, Knowles & Corne (2005) summarize much of the relevant literature, highlighting the most important considerations for the design of Multi-Objective MAs.

5.5 Summary

The aim of this chapter is to provide the reader with the basic notions of evolutionary computation. These notions are essential for the understanding of some the proposals in this PhD Thesis, in which evolutionary strategies have been used for conducting the search of biclusters in microarray data.

Evolutionary Computation is a programming technique that is mostly applied for optimization problems in which the search space is very large. It works by repeatedly applying different operators to a set of candidate solutions. The use of a population instead of a single potential solution allows the exploration of a larger portion of the search space, giving also the possibility of escaping from local optima.

Common features of evolutionary computation are first presented in this chapter, including the individuals encoding and initialisation, evaluation, selection and reproduction processes. In the second section we described the four paradigms in which evolutionary computation is usually divided: *evolution strategies*, *evolutionary programming*, *genetic algorithms* and *genetic programming*. Depending on the size of the population and the way in which its individuals are evolve, there exists three different types of evolution models: generational, steady state or generational gap. All of them are described in section four. Finally, some advances evolutionary algorithms, such as multi-objective or memetic are explained in the last section. A multi-objective algorithm is used whenever two or more, usually conflicting, objectives are used in the search, tough there exists multiple variants to cope with this situation. Memetic algorithms are characterized by applying a separate local search process to refine individuals within the search, with independence of being a multi-objective approach or not.

Part III

Proposals

Chapter 6

Standardization-based Evaluation Measures

The importance of having an adequate bicluster coherence measure has already been discussed in chapter 4, where the different types of expression patterns in biclusters were also introduced. A suitable evaluation measure for biclusters can not only be used for guiding the search in different heuristics but also for comparing the results of different biclustering approaches.

We have centred our research upon the development of evaluation measures for biclusters based on the concept of expression patterns. More formally, we have been working towards the definition of a quality metric able to recognize the most general type of pattern: the shifting and scaling combined pattern (see section 4.2 for more information on expression patterns). In order to capture this kind of tendencies we have make use in our different approaches of a standardization procedure of the bicluster, either by rows or columns.

Following sections give formal definitions of our proposed evaluation measures for biclusters. After defining the standardization procedure applied, we describe three different metrics: *Maximal Standard Area* (MSA), *Virtual Error* (VE) and *Transposed Virtual Error* (VE^t). The latest one, VE^t, has been proven to be effective for capturing the combined patterns. Further research involving VE^t is presented in next chapter, where it has been incorporated into a customizable evolutionary biclustering algorithm.

6.1 Standardization Procedure

An important observation that can be extracted from an analysis of previously found biclusters is that the range of expression values assumed by

genes can vary substantially depending on the specific microarray taken as input. Therefore, in order to make an appropriate comparison between each gene and the pattern, it would be desirable to define a mechanism for scaling the expression levels to a common range. This mechanism would also be responsible for soften every gene behaviour, since the most important aspect is to characterize its tendency rather than its numerical values.

As usual in this PhD Thesis, we denote a bicluster $\mathcal{B}$ as a sub-matrix made up of a set I of $|I|$ genes and a set J of $|J|$ experimental conditions, where b_{ij} denotes the expression level of gene i under experimental condition j, for $1 \leq i \leq |I|$ and $1 \leq j \leq |J|$.

Definition 1 (Gene Standardization) *We define the standardization of $\mathcal{B}$ as the bicluster $\hat{\mathcal{B}}$, whose element $\hat{b_{ij}}$ are obtained as follows:*

$$\hat{b_{ij}} = \frac{b_{ij} - \mu_{g_i}}{\sigma_{g_i}}, 1 \leq i \leq |I|, 1 \leq j \leq |J|$$

where σ_{g_i} is the standard deviation of all the expression values of gene i and μ_{g_i} is the mean of row i in $\mathcal{B}$.

By means of the standardization, two distinct tasks are carried out. The first one is to shift all the genes to a similar range of values (near 0 in this case). The second one is to homogenize the expression values for each gene, modifying in this way their values under all the conditions, and smoothing their graphical representation, due to the correction of the global scaling factor in the denominator.

6.2 Maximal Standard Area (MSA)

As already stated, MSR by Cheng & Church (2000) cannot recognize some kind of biclusters as quality biclusters. In order to overcome this drawback, we proposed a novel measure for assessing the quality of biclusters on microarray, called *Maximal Standard Area* (MSA). The idea behind is the area of the region between the maximum and minimum values of expression levels that genes assume under the conditions contained in the bicluster. Thus, what it is measured is the area depicted by the maximal fluctuation of expression levels for each experimental condition. For each condition, the minimum and maximum values of expression level for all the genes contained in the bicluster are taken. These pairs of values define a band across all the conditions in the bicluster, and the area of this band is therefore the measure MSA. In the following of this section we will provide a detailed description of this measure.

Definition 2 (Bounds of bicluster $\mathcal{B}$) *We define the upper bound of a bicluster $\mathcal{B}$ for condition j as*

$$M_j(\mathcal{B}) = max_i\ b_{ij},\ \forall i$$

and similarly the lower bound of bicluster $\mathcal{B}$ for condition j as

$$m_j(\mathcal{B}) = min_i\ b_{ij},\ \forall i$$

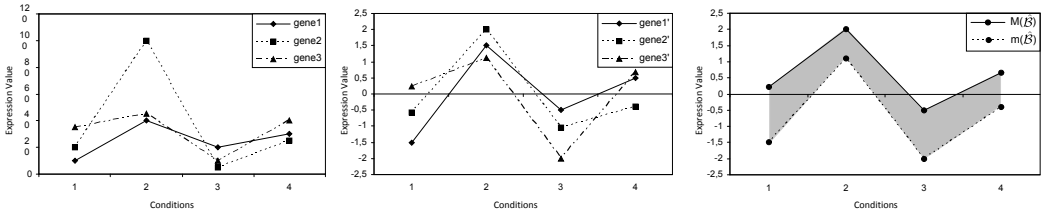
We can now define the proposed measure:

Definition 3 (MSA) *We define MSA($\mathcal{B}$) as the area delimited by the bounds for each condition in the standardized bicluster as follows:*

$$MSA(\mathcal{B}) = \sum_{j=1}^{|J|-1} \left| \frac{M_j(\hat{\mathcal{B}}) - m_j(\hat{\mathcal{B}}) + M_{j+1}(\hat{\mathcal{B}}) - m_{j+1}(\hat{\mathcal{B}})}{2} \right|$$

where $\hat{\mathcal{B}}$ is the standardized bicluster.

As an example, Figure 6.1(a) shows a bicluster $\mathcal{B}$ containing three genes and four conditions. In Figure 6.1(b) the resulting standardized bicluster $\hat{\mathcal{B}}$ is displayed. It can be noticed that the standardized genes assume closer values than the original ones. In particular, in Figure 6.1(a), the value assumed by the second gene under the second condition is about 2.5 times greater than the value assumed by the other two genes under the same condition. As a result of the standardization, this difference is much less stressed, while the general tendency of the three genes is maintained. Finally, Figure 6.1(c) shows $M(\hat{\mathcal{B}})$ and $m(\hat{\mathcal{B}})$ for each condition. MSA($\mathcal{B}$) is illustrated by the grey region.



(a) Bicluster $\mathcal{B}$ (b) Standardized Bicluster $\hat{\mathcal{B}}$ (c) M($\hat{\mathcal{B}}$), m($\hat{\mathcal{B}}$) and MSA

Figure 6.1: Example to illustrate the Maximal Standard Area (MSA). In (a) a bicluster example; in (b) its standardization; in (c) the MSA.

If the genes of a bicluster $\mathcal{B}$ have a perfectly coherent trend, then MSA($\mathcal{B}$) is equal to zero. On the contrary, MSA will assume higher values when the

genes are less correlated with each other, due to the fact $M(\hat{\mathcal{B}})$ and $m(\hat{\mathcal{B}})$ are more distant from each other. It follows that we assume that biclusters characterized by a low MSA are interesting for further biological studies.

Various experiments were carried out in which MSA was introduced into a evolutionary biclustering algorithm producing interesting results (Giraldez et al (2007)). Nevertheless, after several preliminary experiments regarding the use of VE (presented in next section) and MSA within evolutionary biclustering approaches (Pontes et al (2007a)), we concluded that VE outperforms MSA. We therefore continued our research focusing our efforts on VE.

6.3 Virtual Error (VE)

Virtual Error (VE) follows similar assumptions regarding the standardization process within the evaluation. The basic idea behind VE is to measure how genes follow the general tendency within the bicluster. This is because if all the genes of a bicluster follow the same tendency under a given set of conditions, then it means that they are activated/deactivated under the same experimental conditions. It follows that such a bicluster may be potentially biologically interesting.

In order to catch the general tendency of the genes across the conditions contained in the bicluster, we first calculate a new row from the genes of the bicluster, called *virtual pattern* or *virtual gene*, defined as follows:

Definition 4 (Virtual Pattern) *Given a bicluster $\mathcal{B}$, we define its virtual pattern/gene ρ as the set of elements $\rho = \{\rho_1, \rho_2, \ldots, \rho_{|J|}\}$, where ρ_j, $1 \leq j \leq |J|$, is defined as the mean of the j^{th} column or experimental condition:*

$$\rho_j = \frac{1}{|I|} \sum_{i=1}^{|I|} b_{ij} \tag{6.1}$$

Each of the elements of the virtual pattern represents the average value for all genes under a specific condition. Thus, if we graphically represent this values next to the real genes, the virtual one symbolizes the common tendency of the set of genes for the given bicluster.

Once the virtual gene ρ has been computed, we can assess how well a specific gene g_i of the bicluster follows the general tendency. In order to do this, we compute the differences between the expression level values of g_i and the values of ρ for each experimental condition of the bicluster.

However, computing such differences using the original expression values can yield to a misclassification of the bicluster. In fact,the range of values of

the expression values of the genes may be very far from each other. In order to remove or minimize these range differences, we will use the standardized gene expression values as explained above in section 6.1. By using this numerical transformation, we standardize the expression values of every gene, including the virtual one, scaling the values to a common range. This way, the standardized values of the virtual gene will be given by the following expression:

$$\hat{\rho_j} = \frac{\rho_j - \mu_\rho}{\sigma_\rho} \tag{6.2}$$

where μ_ρ and σ_ρ represent the mean and the deviation of the values in the virtual pattern, respectively.

We now define VE as the average value of all the differences between the standardized expression values and the standardized virtual pattern:

Definition 5 (Virtual Error) *The Virtual Error of a bicluster $\mathcal{B}$, denoted by $VE(\mathcal{B})$, is defined as:*

$$VE(\mathcal{B}) = \frac{1}{|I| \cdot |J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} |\hat{b}_{ij} - \hat{\rho}_j| \tag{6.3}$$

where $\hat{b}_{ij}$ is the standardized expression value of the element $b_{ij} \in \mathcal{B}$, and $\hat{\rho}_j$ is the standardized value of the element ρ_j in the virtual pattern ρ.

VE computes the differences between the real genes and the virtual one, once they have been standardized. Therefore, the more similar the genes are, the lower the value for VE. In fact, if a bicluster follows a perfect shifting or scaling pattern, VE is zero (see section 6.3.1 below). It follows that the lower the VE the better the bicluster.

A diagram on how VE is computed for any input bicluster is shown in Figure 6.2. The whole process is comprised by four different steps: calculation of the virtual pattern, standardization of both the virtual gene and the whole bicluster, and finally VE is given by the average of the differences between every standardized gene component and its corresponding standardized pattern element.

The smoothing effect of the standardization is clear in Figure 6.3 (b), where the range of values in the y-axis is significantly narrower than the original one shown in Figure 6.3 (a). The bicluster in the example has a VE value of 0.21, i.e near to zero. This result shows that the genes follows a very similar behaviour across the conditions, as Figure 6.3 (a) confirms. Therefore, this low value of VE indicates the good quality of the bicluster.

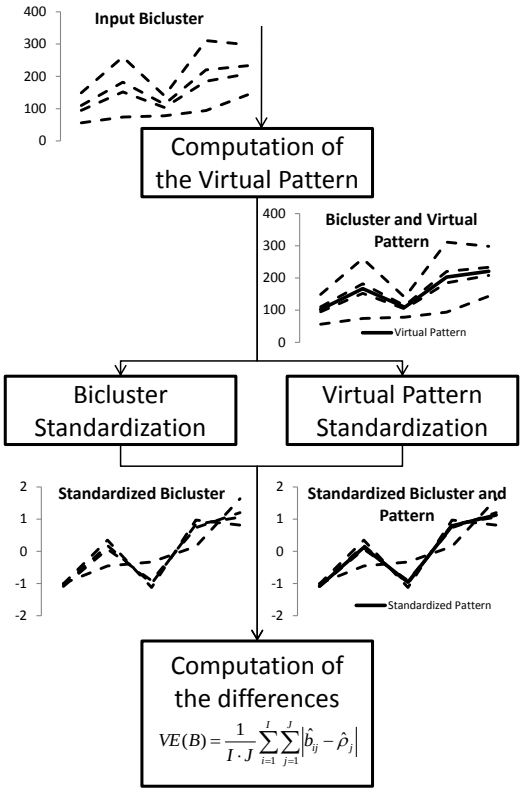
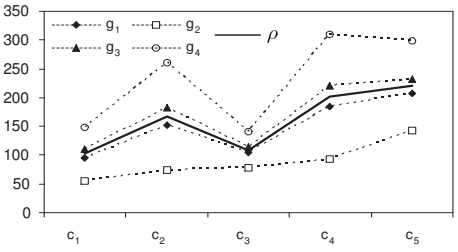


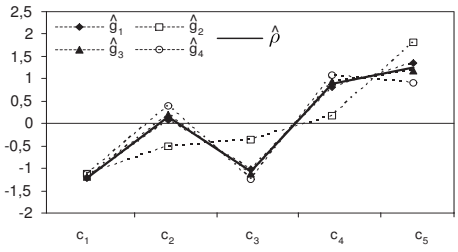
Figure 6.2: Virtual Error computation diagram.



$$\mathcal{B} = \begin{pmatrix} 95 & 152 & 104 & 185 & 208 \\ 56 & 74 & 78 & 94 & 143 \\ 110 & 182 & 114 & 220 & 233 \\ 149 & 260 & 140 & 311 & 298 \end{pmatrix}$$

$$\rho = (\ 102.5 \quad 167 \quad 109 \quad 202.5 \quad 220.5\ )$$

(a) Bicluster and the virtual pattern.



$$\hat{\mathcal{B}} = \begin{pmatrix} -1.22 & 0.07 & -1.02 & 0.82 & 1.34 \\ -1.12 & -0.51 & -0.37 & 0.17 & 1.83 \\ -1.20 & 0.20 & -1.12 & 0.93 & 1.19 \\ -1.13 & 0.39 & -1.25 & 1.09 & 0.91 \end{pmatrix}$$

$$\hat{\rho} = (\ -1.21 \quad 0.14 \quad -1.07 \quad 0.88 \quad 1.26\ )$$

(b) Standardized bicluster and the standardized virtual pattern.

Figure 6.3: Example to illustrate the Virtual Error.

In general, we can conclude that VE is robust when a bicluster follows a shifting or scaling pattern since, in these cases, the value of VE is equals to zero.

6.3.1 Virtual Error Analysis

Although several experiments carried out on real data microarrays have been carried out (Pontes et al (2007a)) that confirm VE validity for finding biclusters with shifting or scaling tendencies, here we state two theorems and their corresponding proofs demonstrating that VE is zero when a bicluster follows either a shifting or scaling pattern.

Theorem 1 *A bicluster presenting a perfect shifting pattern has Virtual Error equal to zero.*

Proof 1 *If $\mathcal{B}$ follows a perfect shifting pattern, then we can represent each element as $b_{ij} = \pi_i + \beta_j$.*

Applying two simple algebraic properties[1]*, it is easy to obtain the mean and deviation of each gene g_j as:*

$$\mu_{g_i} = \pi_i + \mu_\beta$$

$$\sigma_{g_i} = \sigma_\beta$$

We use this results to standardize b_{ij}:

$$\hat{b}_{ij} = \frac{b_{ij} - \mu_{g_i}}{\sigma_{g_i}} = \frac{\pi_i + \beta_j - \pi_i - \mu_\beta}{\sigma_\beta} = \frac{\beta_j - \mu_\beta}{\sigma_\beta} \tag{6.4}$$

Combining the same former properties, it is easy to obtain the mean and standard deviation for the virtual pattern as:

$$\mu_\rho = \mu_\pi + \mu_\beta$$

$$\sigma_\rho = \sigma_\beta$$

Finally, the standardized values of the virtual pattern are represented by:

$$\hat{\rho}_j = \frac{\rho_j - \mu_\rho}{\sigma_\rho} = \frac{\mu_\pi + \beta_j - \mu_\pi - \mu_\beta}{\sigma_\beta} = \frac{\beta_j - \mu_\beta}{\sigma_\beta} = \hat{b}_{ij} \tag{6.5}$$

[1]Being $f(x) = g(x) \times c_1 + c_2$, we can enumerate these two properties related to the arithmetic mean ($\mu_{f(x)}$) and the standard deviation ($\sigma_{f(x)}$) of $f(x)$ as: $\mu_{f(x)} = \mu_{g(x)} \times c_1 + c_2$ and $\sigma_{f(x)} = \sigma_{g(x)} \times c_1$.

This result points out that when a bicluster follows a perfect shifting pattern, the standardized virtual pattern is equal to all the real genes after standardizing them. This means that VE is equal to zero for every bicluster with a perfect shifting pattern. ■

Theorem 2 *A bicluster presenting a perfect scaling pattern has Virtual Error equal to zero.*

Proof 2 *If $\mathcal{B}$ follows a perfect scaling pattern, then we can represent each element as $b_{ij} = \pi_i \times \alpha_j$.*

Following the same reasoning than for Theorem 1, the mean and deviation of each gene g_i are the following:

$$\mu_{g_i} = \pi_i \times \mu_\alpha$$

$$\sigma_{g_i} = \pi_i \times \sigma_\alpha$$

We use these results to standardize the values of the bicluster:

$$\hat{b}_{ij} = \frac{b_{ij} - \mu_{g_i}}{\sigma_{g_i}} = \frac{\pi_i \times \alpha_j - \pi_i \times \mu_\alpha}{\pi_i \times \sigma_\alpha} = \frac{\alpha_j - \mu_\alpha}{\sigma_\alpha} \tag{6.6}$$

We next obtain the mean and standard deviation for the virtual pattern:

$$\mu_\rho = \mu_\pi \times \mu_\alpha$$

$$\sigma_\rho = \mu_\pi \times \sigma_\beta$$

And the standardized values of the virtual pattern:

$$\hat{\rho}_j = \frac{\rho_j - \mu_\rho}{\sigma_\rho} = \frac{\mu_\pi \times \alpha_j - \mu_\pi \times \mu_\alpha}{\mu_\pi \times \sigma_\alpha} = \frac{\alpha_j - \mu_\alpha}{\sigma_\alpha} = \hat{b}_{ij} \tag{6.7}$$

As we can observe, the result of the last equation shows that when a bicluster follows a perfect scaling pattern, the standardized virtual pattern is equal to all the real genes after standardizing them. Therefore, we can state that VE is equal to zero for every bicluster with a perfect scaling pattern. ■

6.4 Evolutionary Biclustering with Virtual Error

In order to further test the effectiveness of VE, we also incorporated it into a multi-objective environment using Multi-Objective Evolutionary Biclustering (MOEB). Experiments on real data sets and their biological validation

show that VE yields the algorithm at finding interesting biclusters, as shown below. We also compared the results with those obtained using the same evolutionary approach but guiding the search with MSR. Following sections present the biclustering algorithm used in these experiments as well as the obtained results.

6.4.1 Sequential Multi-objective Biclustering

VE has been introduced as a quality measure that can be used to guide an optimization heuristic in order to discover biclusters in an expression matrix. However, the problem cannot be addressed by only optimizing the VE of biclusters. In fact, this approach may lead to the discovery of uninteresting sub-matrices. For instance, flat biclusters will have a low value of VE, or, again, biclusters containing few genes and conditions will typically have lower values of VE, if compared to biclusters characterized by higher volume. The same hold for the MSR. This is because the more genes or conditions are contained in a bicluster, the less likely the genes are to follow the same behaviour. Such biclusters are not very interesting, and, in order to solve this issue, other properties of the biclusters are usually optimized. In particular, we are interested in finding biclusters with high volume, good quality (being quality measured by an appropriate metric such as VE, MSA or MSR) and relatively high gene variance. Thus, we can individuate at least three objectives to be optimized and these objectives are usually in conflict with each other. For this reason, the problem of finding biclusters in an expression matrix can be straightforwardly seen as a multi-objective problem. Moreover, by addressing this problem as a multi-objective problem, it is not necessary to combine all the objectives into single cost function, which might become complicated, especially when both maximization and minimization are involved.

The algorithm used in this work is called SMOB (*Sequential MultiObjective Biclustering*), and is outlined in Algorithm 3. The algorithm is similar to SEBI (*Sequential Evolutionary BIclustering*), which was introduced in Divina & Aguilar-Ruiz (2006) and adopts a sequential covering strategy. Unlike SEBI, where a single-objective EA was used, SMOB invokes several times a multi-objective evolutionary algorithm. Each time the MOEA is called, a bicluster is returned. Biclusters returned are stored in a list, where the number of elements in the output list represents the number of found solutions. Since SMOB iterates until certain criteria is met, this number will be dependant on the stopping criteria. In our experiments we have defined an input parameter for specifying the number of desired solutions. In Divina & Aguilar-Ruiz (2006) a threshold δ on MSR was used in order to reject

biclusters. In SMOB we do not use any threshold because several objectives are optimized at the same time, and thus biclusters cannot be rejected based on a bad result of a single objective.

Algorithm 3: SMOB for sequential covering

input : $\mathcal{M}$: expression matrix
output: L: list of biclusters

```
1  load Expression Matrix M
2  L ← {}
3  matrix of weights W ← {}
4  bicluster b
5  repeat
6  |  b = MOEB (M) if b is not null then
7  |  |  L ← L ⊕ b
8  |  |  adjust weights of M
9  |  else
10 |  |  end_cond met
11 |  if max_iter is reached then
12 |  |  end_cond met
13 until end_cond is met;
14 return L
```

Algorithm 4: Procedure **MOEB**

input : $\mathcal{M}$: expression matrix, $\mathcal{W}$: matrix of weights
output: Best individual in population

```
1  initializePopulation
2  evaluatePopulation
3  repeat
4  |  selectParents
5  |  recombinePairOfParents
6  |  mutateResultingOffspring
7  |  evaluateNewPopulation
8  |  selectIndividualsForNextGen
9  until max_iter is reached;
10 best_ind ← bestIndividualInPopulation
11 return best_ind
```

SMOB adopts a sequential covering strategy, where the MOEB procedure is called n times, and each time a bicluster is returned (see Algorithm 3). The returned bicluster is stored in a list L that contains all the biclusters found so far. When MSR is used as an objective, the returned bicluster is stored in the list only if its MSR is lower than the threshold δ. Notice

that the sequential coverage strategy adopted is such that the order in which biclusters are discovered does not reflect their quality nor their biological relevance. Next subsections details how the common characteristics of EC have been implemented in this study.

Biclusters Codification and Initialization

The encoding of biclusters used in this approach is the one proposed in Divina & Aguilar-Ruiz (2006), where bit strings are used. A bit is associated to each gene and each condition of the expression matrix. If a bit is set to one, it means that the relative gene/condition belongs to the bicluster, otherwise it does not.

Individual initialization is the first task performed by MOEB procedure (line 1 in Algorithm 4). In this work, individuals are initialized in the following way: first, the number of genes $|I|$ and of conditions $|J|$ contained in the biclusters are randomly determined. Then, $|I|$ bits corresponding to genes and $|J|$ bits corresponding to conditions are randomly selected. The selected bits are set to one, which means that the relative gene/condition is contained in the bicluster encoded by the individual. We perform this initialization instead of a pure random initialization of bit-strings, because in that way the initial biclusters would contain all about the same number of genes and conditions.

Individual Evaluation

Evaluation is performed after the initialization (line 2 in Algorithm 4) and also after the new population has been created using the genetic operators (line 7 in Algorithm 4). In a multi-objective approach, evaluation needs to be carried out for all the individuals since the fitness is dominance-dependent and thus the fitness of the same individual in two different generations may vary.

We have individuate four objectives to be optimized: MSR, VE, gene variance and the volume of the biclusters. However, as it will be described in Section 6.4.2, we will use three different settings of the algorithm, where only three objectives will be optimized at a time. The objectives that will always be considered are the volume and the gene variance.

The evaluation in this work adopts a strategy similar to NSGA (see section 5.4.1), where individuals are divided into different non-dominated fronts, and individuals belonging to the same front have the same starting fitness $rank(\mathcal{B})$. For instance, a non-dominated individual will have a rank value

equals to zero. The fitness of an individual $\mathcal{B}$ is then defined as:

$$\phi(\mathcal{B}) = rank(\mathcal{B}) + sh(\mathcal{B}) + P(\mathcal{B}) \tag{6.8}$$

where $sh(\mathcal{B})$ is the phenotypic sharing (Zitzler & Thiele (1999)) and $P(\mathcal{B})$ is used in order to avoiding overlapping among biclusters. In our implementation $sh(\mathcal{B})$ is the minimal euclidean distance, computed on the objectives to the other individuals as follows:

$$sh(\mathcal{B}) = min\left(\sqrt{\sum_{i=1}^{n}(\mathcal{B}_i - \mathcal{B}_{2i})^2}\right) \tag{6.9}$$

where $\mathcal{B}_2$ represents every individual in the population different from $\mathcal{B}$ and $\mathcal{B}_i, \mathcal{B}_{2i}$ are the objectives used.

In order to avoid overlapping among biclusters, a weights-based approach similar to the one defined in section A.1.2 is used, where a weight $\mathcal{W}(b_{ij})$ is associated with each element b_{ij} of the bicluster in the original expression matrix. These weights are initialized to zero in line 3 of the Algorithm 3 and are adjusted every time a bicluster is returned after each call of MOEB (line 8 of SMOB). Therefore, $\mathcal{W}(b_{ij})$ is equals to the number of biclusters stored in L that contain the same element of the input expression matrix. When evaluating a bicluster the weights of its elements are used in order to penalize biclusters overlapping with elements of L.

When a bicluster $\mathcal{B}$ is evaluated inside MOEB, a penalty in the form of equation 6.10 is added to the fitness of $\mathcal{B}$, where $V(\mathcal{B})$ is the volume of $\mathcal{B}$. In this way, if a bicluster has low volume and it covers elements of the expression matrix that are already contained in many biclusters already found, $P(\mathcal{B})$ will be high. On the other hand, if the bicluster has a high volume and it overlaps with few biclusters, the penalty will be lower. If the bicluster $\mathcal{B}$ does not overlap with any bicluster then $P(\mathcal{B})$ is zero.

$$P(\mathcal{B}) = 1 - \frac{V(\mathcal{B}) - \sum_{i,j \in \mathcal{B}} \mathcal{W}(b_{ij})}{V(\mathcal{B})} \tag{6.10}$$

Selection

Individuals are selected with a tournament mechanism (see section 5.1.3), where a group of four individuals is picked at random and in an independent way from the population. The fittest individual of the pool is afterwards selected as a parent for applying genetic operators and creating offspring. Tournament selection mechanism will be carried out as many times as parents are needed (line 4 in Algorithm 4).

Elitism is also applied by letting the non-dominated individuals survive to the next generation. However, only a maximum of half population is allowed to the next generation. This is to prevent the case where all the population is non-dominated and thus copied to the next generation. In this case the evolutionary process would stop, since only individuals belonging to the previous generation will be inserted in the population.

Genetic Operators

Genetic operators are applied in lines 5 and 6 in Algorithm 4. Several crossover and mutation operators are defined in order to recombine and renovate the next generation individuals.

For recombining parents, three crossover operators are used with different probabilities: one-point crossover, two-point crossover and uniform crossover. However, the application of the uniform crossover is the one having the highest probability. Uniform crossover is preferred to the other two crossovers because one-point and two-point crossover would prohibit certain combinations of bits to be crossed over together.

Resulting offspring are mutated in three different ways: using a classical mutation operator, one that can add a row and one that can add a column. We consider columns and rows separately, because typically there are many more columns than rows, thus considering them together, would give more probability of mutation to columns than to rows.

6.4.2 Experimental Results on Real Data Microarrays

In order to assess the validity our VE in a multi-objective environment, we have conducted experiments on nine datasets shown in Table 6.1. The embryonal dataset was preprocessed as by Tavazoie et al (1999), where each entry of the original dataset was substituted by its normalized value between 0 and 600. All the other datasets were preprocessed as by Cheng & Church (2000). The most important preprocessing operation regards missing values. They are replaced with random values, although it is known the existing risk that these random numbers can affect the discovery of biclusters (Yang et al (2002)). The expectation was that these random values would not form recognizable patterns.

The aim of the experimentation is to assess the validity of VE as a quality measure. In particular, we aim to test whether VE can yield the discovery of biclusters characterized by higher gene variance and volume. Moreover, we are interested in comparing the results achieved by using VE as a measure of the quality of a bicluster against the results obtained when MSR is used.

Dataset	Name	#genes	#cond.	Ref.
Yeast	Yeast *Saccharomyces cerevisiae* cell cycle	2884	17	Cho et al (1998)
Human	Human B–cells	4026	96	Alizadeh et al (2000)
Colon	Colon Cancer	2000	62	Alon et al (1999)
Malaria	Malaria *Plasmodium parasites* life cycle	3719	16	Le Roch et al (2003)
Embryonal	Embryonal tumors of the central nervous syst.	7129	60	Pomeroy et al (2002)
Leukemia	Leukemia	7129	72	Golub et al (1999)
RatCNS	Rat Central Nervous System	112	9	Wen et al (1998)
Steminal	Steminal Cells	26127	30	Boyer et al (2006)
PBM	Peripheral Blood Monocytes	2329	139	Hartuv et al (2000)

Table 6.1: Datasets used in the experimentation.

Dataset	δ for SMOB-δ	δ for SMOB-Δ
Yeast	300	3400
Human	1200	23000
Colon	500	3300
Malaria	600	19000
Embryonal	1800	10000
Leukemia	1800	23130700
RatCNS	5	11
Steminal	10	130
PBM	0.3	1.3

Table 6.2: Values of δ used in the settings SMOB-δ and SMOB-Δ.

In order to do this, we compare the algorithm using VE against a version that uses as an objective MSR instead of VE. Another point we want to test, is whether the use of a too low threshold δ may prevent the algorithm from finding interesting biclusters, since the search is limited to biclusters with MSR lower than δ .

In the experimentation, we use three settings of the algorithm, that differ for the objectives subject of optimization:

- SMOB-VE. In this setting the objectives are: volume, gene variance and VE.
- SMOB-δ. In this setting the objectives are: volume, gene variance and MSR, with δ shown in the second column of Table 6.2.
- SMOB-Δ. In this setting the objectives are: volume, gene variance and MSR, with δ shown in the third column of Table 6.2.

In both SMOB-δ and SMOB-Δ, if the algorithm returns a bicluster whose MSR is higher than δ, such bicluster is rejected. As it can be noticed, the only difference between settings SMOB-Δ and SMOB-δ lies in the value used

Parameter	Value
Generations	100
Population size	200
Crossover probability	0.85
Mutation probability	0.2
Tournament size	4

Table 6.3: Parameter settings for the algorithm.

for the threshold δ. SMOB-Δ was included for testing the limitations, in terms of biclusters found, that the use of a low δ can impose on an algorithm.

The values of δ used in SMOB-δ on the human and the yeast dataset were taken from Cheng & Church (2000), while for the other datasets they were established using a procedure suggested in the same work. The values of δ used in SMOB-Δ were determined as follows. For each dataset, we first run the experiments with SMOB-VE. We calculated the MSR of all the bicluster found, and then we selected the highest as δ to be used in SMOB-Δ(third column in Table 6.2). In this way we can test whether the use of δ prevented SMOB-δ from discovering some interesting biclusters, only because their MSR was higher than the used δ. Therefore, SMOB-Δ could obtain similar biclusters to those produced by SMOB-VE guaranteeing in this way a fair comparison.

In order to perform a fair comparison, we use the same parameter setting in all the versions of the algorithm. This setting is given in Table 6.3. The values of these parameters were obtained after a number of preliminary runs aimed at testing different parameter settings.

On each dataset, we obtained 100 biclusters for each setting of the algorithm. The average MSR and VE are reported in Tables 6.4 and 6.5, respectively. Table 6.6 reports both the average gene variance and the average volume. Standard deviation is reported next to each result. In order to test the statistical significance of the results, we applied the Student's T-test of difference of means with confidence level of 1%. In the tables, a minus (plus) symbol next to a result indicates that the average is statistically significantly lower (higher) than the average obtained by SMOB-VE on a given dataset. So, for instance, the average MSR obtained by SMOB-δ on the yeast dataset is significantly lower than the average MSR obtained by SMOB-VE.

In order to present the results in a clearer way, we have not included in the tables information about the statistically significance of the differences of results obtained by SMOB-δ and SMOB-Δ. However, we can say that in Table 6.4, all the results obtained by SMOB-Δ, but the results obtained on the RatCNS datasets, are significantly higher than those obtained by SMOB-δ. The average VE of biclusters obtained by SMOB-Δ is significantly higher

Dataset	MSR				
	SMOB-VE	SMOB-δ		SMOB-Δ	
Yeast	$1419.6_{\pm 513.8}$	$272.0_{\pm 24.9}$	-	$1062.0_{\pm 274.8}$	-
Human	$16441.1_{\pm 3323.2}$	$1103.5_{\pm 86.7}$	-	$10742.5_{\pm 2321.3}$	-
Colon	$2491.0_{\pm 408.9}$	$455.2_{\pm 45.5}$	-	$2107.2_{\pm 306.3}$	-
Malaria	$14456.1_{\pm 2227.3}$	$449.7_{\pm 174.6}$	-	$12095.5_{\pm 2298.1}$	-
Embryonal	$1506.0_{\pm 1932.6}$	$398.2_{\pm 344.2}$	-	$802.97_{\pm 1243.1}$	-
Leukemia	$81.0e5_{\pm 45.0e5}$	$1533.1_{\pm 169.2}$	-	$49.2e5_{\pm 29.3e5}$	-
RatCNS	$3.43_{\pm 2.6}$	$1.63_{\pm 1.3}$	-	$2.17_{\pm 2.0}$	-
Steminal	$43.87_{\pm 37.2}$	$4.07_{\pm 2.1}$	-	$13.13_{\pm 12.9}$	-
PBM	$0.47_{\pm 0.1}$	$0.19_{\pm 0.1}$	-	$0.30_{\pm 0.1}$	-

Table 6.4: Average MSR obtained on each dataset.

Dataset	VE				
	SMOB-VE	SMOB-δ		SMOB-Δ	
Yeast	$0.83_{\pm 0.05}$	$0.73_{\pm 0.08}$	-	$0.86_{\pm 0.05}$	+
Human	$0.90_{\pm 0.04}$	$0.82_{\pm 0.07}$	-	$0.92_{\pm 0.04}$	+
Colon	$0.53_{\pm 0.04}$	$0.33_{\pm 0.07}$	-	$0.53_{\pm 0.03}$	
Malaria	$0.73_{\pm 0.06}$	$0.48_{\pm 0.18}$	-	$0.75_{\pm 0.05}$	+
Embryonal	$0.70_{\pm 0.08}$	$0.88_{\pm 0.07}$	+	$0.89_{\pm 0.08}$	+
Leukemia	$0.82_{\pm 0.08}$	$0.72_{\pm 0.09}$	-	$0.95_{\pm 0.06}$	+
RatCNS	$0.58_{\pm 0.18}$	$0.71_{\pm 0.17}$	+	$0.73_{\pm 0.18}$	+
Steminal	$0.70_{\pm 0.10}$	$0.98_{\pm 0.10}$	+	$1.0_{\pm 0.10}$	+
PBM	$0.28_{\pm 0.10}$	$0.34_{\pm 0.10}$	+	$0.35_{\pm 0.10}$	+

Table 6.5: Average VE obtained on each dataset.

than those obtained by SMOB-δ on five dataset, and in particular on the yeast, human, colon, malaria and the leukemia dataset. As far as the gene variance is concerned, SMOB-Δ obtained significantly higher results on all the datasets but the RatCNS dataset. The average volume characterizing biclusters found by SMOB-Δ is significantly higher on five dataset: yeast, human, colon, malaria and leukemia.

As it can be noticed from Table 6.4, SMOB-δ obtains the lowest values of MSR. This result was expected, since MSR is one of the objective subject to optimization in SMOB-δ. Moreover, in this setting a small threshold is used in order to reject biclusters. However, when this threshold is relaxed, as in SMOB-Δ the average MSR of the biclusters obtained by SMOB-VE is comparable, even if slightly higher in general. The fact that the MSR obtained by SMOB-Δ is lower than that obtained by SMOB-VE is due to the fact that SMOB-Δ considers the MSR as an objective to be optimized, whilst SMOB-VE

Dataset	Gene Variance			Volume		
	SMOB-VE	SMOB-δ	SMOB-Δ	SMOB-VE	SMOB-δ	SMOB-Δ
Yeast	1661.7±502.2	408.5±77.2 -	1245.3±294.5 -	606.4±216.7	226.1±77.3 -	461.4±95.8 -
Human	17395.7±3376.4	1412.2±168.6 -	11412.4±2476.8 -	1430.6±473.2	362.6±106.7 -	1128.3±260.5 -
Colon	5597.6±617.4	2836.7±1026.7 -	4876.2±548.6 -	1402.7±514.4	197.5±85.1 -	1172.4±334.5 -
Malaria	20245.6±2433.4	2667.3±3765.7 -	16707.6±2914.7 -	628.5±179.2	54.6±31.5 -	461.9±108.2 -
Embryonal	1683.3±2126.2	428.8±365.9 -	849.4±1304.1 -	1121.6±401.7	1456.1±577.3 +	1423.6±519.26 +
Leukemia	8.94e6±4.82e6	2391.3±660.1 -	5.27e6±3.10e6 -	1110.0±348.0	441.7±141.2 -	1132.2±220.4
RatCNS	4.6±3.4	2.1±1.6 -	2.7±2.3 -	128.5±63.4	128.5±81.1	128.6±69.5
Steminal	50.8±40.9	4.4±2.3 -	13.8±13.6 -	999.6±336.0	1312.5±371.8 +	1286.7±399.4 +
PBM	1.3±0.2	0.5±0.2 -	0.8±0.2 -	1427.9±616.1	1659.7±701.2	1744.8±687.9 +

Table 6.6: Average gene variance and volume obtained on each dataset.

does not. Moreover, the results obtained by SMOB-Δ are in general significantly higher than those obtained by SMOB-δ. Only on the RatCNS dataset the two algorithms obtain an average MSR that is not statistically significant. This result shows that a too low threshold restrict the search performed by the algorithm too tightly. This fact can be clearly seen by inspecting Table 6.6. This table presents on the left part the average row variance and on the right part the average volume obtained on the 100 biclusters found for each dataset. We can notice that SMOB-Δ obtains biclusters characterized by higher gene variance and volume than those discovered by SMOB-δ.

As Table 6.5 shows, SMOB-δ obtained the lowest values of VE on five datasets. This may seem odd, since SMOB-δ does not consider VE as an objective to be optimized. Nevertheless, this is explained by the fact that low values of MSR correspond to low values of VE. Moreover, in SMOB-δ a low threshold was used to limit the values of MSR. This kind of threshold is not used in SMOB-VE. Notice that VE is not the only objective being optimized by SMOB-VE, being the other ones the volume and the gene variance. This fact, in combination with the above considerations, explains why the values of VE obtained by SMOB-VE are, on average, higher than those obtained by SMOB-δ. It is worth to note that when the threshold δ is relaxed, the results obtained by SMOB-VE are significantly better than those provided by SMOB-Δ.

Due to the presence of shifting patterns in biclusters with low MSR, low values of MSR correspond to very low values of VE. However, the opposite is not true, i.e., low values of VE do not correspond to low values of MSR. This is explained by the presence of scaling patterns, which do not affect VE, but have the effect of remarkably increasing the values of MSR, as proven in Aguilar-Ruiz (2005).

From Table 6.6, we can notice that, on the two common objectives (i.e., gene variance and volume) subject of optimization by all the three settings of the algorithm, SMOB-VE obtains the best results. In particular the gene variance obtained by SMOB-VE is much higher than the gene variance of the

biclusters found by SMOB-δ. This is because gene variance is in conflict with MSR. In fact, the presence of scaling patterns have the effect of incrementing the gene variance. On the other hand, MSR is also incremented by the presence of scaling patterns. Thus, since both the gene variance and the MSR are being optimized at the same time in SMOB-δ most scaling patterns are rejected. This is because such patterns would lead to biclusters with a MSR higher than the used threshold δ. It is also interesting to notice that SMOB-VE obtains biclusters with higher gene variance and volume than those obtained by SMOB-Δ. This confirms the fact that VE is more successful than MSR in guiding the search performed by the algorithm towards the discovery of more interesting biclusters. In fact, a low threshold imposed on MSR prevents the algorithm from finding biclusters containing certain patterns, and even if this threshold is increased, the algorithm obtains biclusters that are less interesting that those obtained when VE is used as one of the objectives.

Biological Validation

This section presents a biological validation of the results obtained by the three settings of the algorithm on three datasets: Embryonal, Leukemia and Steminal. We have selected these datasets because they were the ones having the highest percentage of gene names that could be found in GO (Ashburner et al (2000)) and because of their significance. In order to validate the results, we first use the *Gene Functional Dissimilarity* (GFD) measure (Díaz-Díaz & Aguilar-Ruiz (2011)) and then, for each method, the number of significant biclusters, according to GO (see section 4.6), is extracted.

GFD assigns a numerical value, between zero and one, to the gene set contained in a bicluster for each of the three GO sub-ontologies (*Molecular Function* (MF), *Cellular Component* (CC) and *Biological Process* (BP)). The value assigned to a biclusters represents the functional cohesion of the genes, where lower values represent higher functional similarity. We decided to use GFD since this measure presents the advantage that it can identify the most common function for all of the genes involved in a biological process.

Table 6.7 shows the average GFD for all the biclusters obtained on the three datasets by the three settings of the algorithm, according to each GO sub-ontology. Standard deviation is also reported next to the averages. In order to test whether the results are significantly different, we performed a two-tailed t-test. Results of this test, with confidence levels of 5% and 1%, are reported in the right part of the table. Thus, for instance, the average GFD obtained on the Embryonal dataset by SMOB-VE and SMOB-δ is significantly different for the MF and CC sub-ontologies with a confidence level of 1%, while for the BP ontology the difference is significant with a 5% confidence

Dataset	GO's Sub-Ontology	Average GFD			Statistical Significance (T-Test)			
		SMOB-VE	SMOB-δ	SMOB-Δ	SMOB-VE vs SMOB-δ		SMOB-VE vs SMOB-Δ	
					< 0.05	< 0.01	< 0.05	< 0.01
Embryonal	MF	0.425± 0.100	0.464± 0.099	0.474± 0.106	×	×	×	×
	BP	0.603± 0.077	0.626± 0.068	0.633± 0.067	×		×	×
	CC	0.389± 0.098	0.446± 0.068	0.442± 0.083	×	×	×	×
Luekemia	MF	0.442± 0.098	0.505± 0.114	0.471± 0.074	×	×	×	
	BP	0.642± 0.050	0.648± 0.077	0.652± 0.050				
	CC	0.439± 0.081	0.473± 0.102	0.444± 0.067	×	×		
Steminal	MF	0.617± 0.044	0.632± 0.038	0.641± 0.036	×	×	×	×
	BP	0.710± 0.036	0.710± 0.023	0.715± 0.022				
	CC	0.492± 0.071	0.527± 0.052	0.528± 0.051	×	×	×	×

Table 6.7: Average GFD values for each of the three sub-ontologies of GO.

Dataset	p-value	Number of Biclusters		
		SMOB-VE	SMOB-δ	SMOB-Δ
Embryonal	< 0.01	1	2	5
	< 0.05	16	6	10
Leukemia	< 0.01	4	2	2
	< 0.05	12	6	9
Steminal	< 0.01	11	1	2
	< 0.05	27	10	4

Table 6.8: Number of significant biclusters for the three GO ontologies, at two different levels.

level. From the table, we can notice that SMOB-VE obtains the lowest GFD in all the cases but one: on the Steminal dataset for the BP ontology, where the three settings of the algorithm obtains basically the same results. This implies that, on average, the genes contained in the biclusters obtained on these datasets by using VE present a stronger functional similarity than those obtained by using MSR. It is also interesting to notice that in some cases, e.g., on the Leukemia dataset for the MF sub-ontology, SMOB-Δ obtains a lower value of GFD than SMOB-δ. This means that in such cases the average functional cohesion is stronger for the biclusters discovered by relaxing the threshold δ used with MSR. Thus, in some cases, a too strict threshold may prevent the algorithm from finding interesting biclusters.

To further analyse the results, we have used the ontologizer tool (Bauer et al (2008)) to directly compute the most significantly enriched GO terms associated to the set of genes of every bicluster. Results of this analysis have been obtained using the methodology described in section 4.6 and are reported in table 6.8. This table shows the number of biclusters with at least one significant GO term in any ontology, at two different levels: 1% and 5%. For Leukemia and Steminal datasets, SMOB-VE discovers more significant bi-

clusters than SMOB-δ and SMOB-Δ with both levels. As far as the Embryonal dataset is concerned, SMOB-VE outperforms the other two versions of the algorithm when the p-value considered is lower than 0.05. In this case, of all the biclusters discovered by SMOB-VE, 16 are significant, while only 6 biclusters found by SMOB-δ are significant and SMOB-Δ finds 10 significant biclusters. If the threshold on the p-value is lowered to 0.01, only 1 bicluster is significant for SMOB-VE, whereas the other two settings finds 2 and 5 significant biclusters.

6.5 VE^t: Transposed Virtual Error

Although VE has been proven to be efficient for finding significantly enriched biclusters when using it into an evolutionary strategy, it is not able to capture shifting and scaling simultaneous tendencies. In this section we present an enhanced version of VE, named VE^t, from *Transposed Virtual Error*, analytically proving that VE^t is zero for those biclusters with perfect shifting and scaling patterns. This variation of VE has been motivated by the work of Cho (2010), where several numeric transformations have been applied to the data in order to detect both kind of patterns.

VE^t is computed similarly to VE but considering the transposed bicluster. The idea here is to create the virtual pattern in the condition dimension. We will name this virtual pattern *Virtual Condition*, in order to differentiate it from the pattern in VE. Afterwards, the differences between the standardized values for every condition and the standardized virtual condition are measured in the same way as in VE. In the following, we explain how to create the virtual condition for a certain bicluster $\mathcal{B}$, in order to compute VE^t.

Definition 6 (Virtual Condition) *Given a bicluster $\mathcal{B}$ with a set I of $|I|$ genes and a set J of $|J|$ conditions, we define its virtual condition as a collection of $|I|$ elements ρ_i, each of them defined as the mean of the i^{th} row representing a gene:*

$$\rho_i = \frac{1}{|J|}\sum_{j=1}^{|J|} b_{ij} \tag{6.11}$$

This way, each element of the virtual condition represents a meaningful value for all the conditions, regarding each gene. Once the virtual condition has been created, the next task would consist of quantifying the way in which all the experimental conditions in the bicluster are similar to it. In

order to perform an appropriate comparison, we first carry out a standardization of the virtual condition and of every experimental condition in the bicluster. This standardization allows us to capture the differences among the tendencies, with independence of the numerical values.

Nevertheless, the standardization procedure we carry out for the computation of VE^t differs from the one used in the case of VE. Here, the standardization is performed on the condition dimension, being the elements $\hat{b_{ij}}$ of the standardized bicluster $\hat{\mathcal{B}}$ obtained as follows: $\hat{b_{ij}} = \frac{b_{ij}-\mu_{c_j}}{\sigma_{c_j}}$, where σ_{c_j} and μ_{c_j} represent the standard deviation and the arithmetic average of all the expression values for condition j, respectively.

As it has already been said, the virtual condition needs also to be standardized. Equation 6.12 shows how the values of the standardized virtual condition are obtained, where ρ_i refers to the virtual condition value for gene i, while μ_ρ and σ_ρ refer to the average and the deviation of the values of the virtual condition, respectively.

$$\hat{\rho}_i = \frac{\rho_i - \mu_\rho}{\sigma_\rho} \tag{6.12}$$

Definition 7 (Transposed Virtual Error) *Given a bicluster $\mathcal{B}$, and its corresponding virtual condition ρ, Transposed Virtual Error (VE^t) can be defined as the mean of the numerical differences between each standardized condition and the values of the standardized virtual condition for each gene:*

$$VE^t(\mathcal{B}) = \frac{1}{|I| \cdot |J|} \sum_{i=1}^{i=|I|} \sum_{j=1}^{j=|J|} (\hat{b_{ij}} - \hat{\rho}_i) \tag{6.13}$$

Next, we present three theorems and their proofs that demonstrate the strength of VE^t with regard to the shifting and scaling patterns.

6.5.1 Analytical Analysis

This section includes formal proofs that bear out the hypothesis that VE^t is zero for those biclusters with perfect shifting and scaling patterns, either separately or simultaneously.

Theorem 3 *A bicluster presenting a perfect shifting pattern has VE^t equal to zero.*

Proof 3 *Let $\mathcal{B}$ be a bicluster with a perfect shifting pattern, then it is possible to refer to its elements as $b_{ij} = \pi_i + \beta_j$. Applying the two same simple*

arithmetic properties described above[1]*, the mean and the deviation for each condition* c_j *can be expressed by:*

$$\mu_{c_j} = \mu_\pi + \beta_j$$

$$\sigma_{c_j} = \sigma_\pi$$

where μ_π *and* σ_π *represent the mean and the deviation of the* π *values, respectively. Using these results we obtain the standardize values for* b_{ij}*:*

$$\hat{b}_{ij} = \frac{b_{ij} - \mu_{c_j}}{\sigma_{c_j}} = \frac{\pi_i + \beta_j - \mu_\pi - \beta_j}{\sigma_\pi} = \frac{\pi_i - \mu_\pi}{\sigma_\pi}$$

Combining the former properties[1] *it is easy to express the mean and standard deviation for the virtual condition as:*

$$\mu_\rho = \mu_\pi + \mu_\beta$$

$$\sigma_\rho = \sigma_\pi$$

Finally, the standardized values for the virtual condition are the following:

$$\hat{\rho}_i = \frac{\rho_i - \mu_\rho}{\sigma_\rho} = \frac{\pi_i + \mu_\beta - \mu_\pi - \mu_\beta}{\sigma_\pi} = \frac{\pi_i - \mu_\pi}{\sigma_\pi} = \hat{b}_{ij}$$

As it can be seen above, the standardized virtual condition is equal to all the real conditions after being standardized. Therefore, VE^t *has been proven to be zero for those biclusters with perfect shifting patterns.* ■

Theorem 4 *A bicluster presenting a perfect scaling pattern has* VE^t *equal to zero.*

Proof 4 *Let* $\mathcal{B}$ *be a bicluster following a perfect scaling pattern, then its elements can be expressed by* $b_{ij} = \pi_i \times \alpha_j$*. Following the same reasoning that in the former proof, the mean and deviation of each condition* c_j *are:*

$$\mu_{c_j} = \alpha_j \times \mu_\pi$$

$$\sigma_{c_i} = \alpha_j \times \sigma_\pi$$

From these results we obtain the standardized values for b_{ij}*:*

$$\hat{b}_{ij} = \frac{b_{ij} - \mu_{c_j}}{\sigma_{c_j}} = \frac{\pi_i \times \alpha_j - \alpha_j \times \mu_\pi}{\alpha_j \times \sigma_\pi} = \frac{\pi_i - \mu_\pi}{\sigma_\pi}$$

Next we obtain the mean and deviation for the values of the virtual condition:

$$\mu_\rho = \mu_\pi \times \mu_\alpha$$
$$\sigma_\rho = \mu_\alpha \times \sigma_\pi$$

And finally the standardized values for the virtual condition are:

$$\hat{\rho}_i = \frac{\rho_i - \mu_\rho}{\sigma_\rho} = \frac{\pi_i \times \mu_\alpha - \mu_\pi \times \mu_\alpha}{\mu_\alpha \times \sigma_\pi} = \frac{\pi_i - \mu_\pi}{\sigma_\pi} = \hat{b}_{ij}$$

As in the previous proof, we obtain that the standardized values for the virtual condition are equal to de standardized values for all the real experimental conditions. As a consequence, VE^t will be zero for every bicluster with a perfect scaling pattern. ■

Theorem 5 *A bicluster presenting a perfect combined pattern (shifting and scaling) has VE^t equal to zero.*

Proof 5 *If $\mathcal{B}$ contains a perfect combined pattern, its values can be represented by $b_{ij} = \pi_i \times \alpha_j + \beta_j$. Using the same arithmetic properties as in the former proofs, the mean and deviation for each condition c_i are:*

$$\mu_{c_j} = \alpha_j \times \mu_\pi + \beta_j$$
$$\sigma_{c_j} = \alpha_j \times \sigma_\pi$$

And the standardized values for b_{ij} can be expressed as:

$$\hat{b}_{ij} = \frac{b_{ij} - \mu_{c_j}}{\sigma_{c_j}} = \frac{\pi_i \times \alpha_j + \beta_j - \alpha_j \times \mu_\pi + \beta_j}{\alpha_j \times \sigma_\pi} = \frac{\pi_i - \mu_\pi}{\sigma_\pi}$$

The mean and deviation for the virtual condition are the following:

$$\mu_\rho = \mu_\pi \times \mu_\alpha + \mu_\beta$$
$$\sigma_\rho = \mu_\alpha \times \sigma_\pi$$

And the standardized values for the virtual condition:

$$\hat{\rho}_i = \frac{\rho_i - \mu_\rho}{\sigma_\rho} = \frac{\pi_i \times \mu_\alpha + \mu_\beta - \mu_\pi \times \mu_\alpha - \mu_\beta}{\mu_\alpha \times \sigma_\pi} = \frac{\pi_i - \mu_\pi}{\sigma_\pi} = \hat{b}_{ij}$$

Again, the standardized values for the virtual condition match up with the standardized values for the original conditions. Therefore, VE^t will also be zero for those biclusters following a perfect shifting and scaling pattern. ■

These results confirm that VE^t is capable of recognizing combined patterns in gene expression data. While MSR is only capable of detecting shifting patterns, and VE cannot recognize both kind of patterns simultaneously, VE^t has been proven to go beyond this other two measures.

6.5.2 Noise Robustness Analysis

This section discusses the use of VE^t for bicluster evaluation, in comparison with other evaluation measures, such as VE and MSR. In particular, we study the value of VE^t for those biclusters in which the presence of patterns is not perfect. That is, when the tendency of the data in a bicluster is similar to a perfect pattern but does not completely match with the equation 6.13.

In order to check the behaviour of VE^t whenever a bicluster does not follow a perfect pattern, we add an additive term ε_{ij} to the combined pattern equation. The meaning of this new term corresponds to the error made by the assumption that the bicluster can be represented by a perfect pattern.

$$b_{ij} = \pi_i \times \alpha_j + \beta_j + \varepsilon_{ij} \tag{6.14}$$

It is possible therefore to study the variations produced to VE^t depending on the values of ε_{ij}. Nevertheless, it is not so simple due to the huge amount of situations depending on the distribution and the magnitude of the ε_{ij} values in the data matrix.

In two specific situations the value of VE^t will not be affected when the errors could be included in the former equation 6.14. These two cases correspond to those in which ε_{ij} values are either a constant or constants per conditions (rows). In both cases it is possible to eliminate the term ε_{ij} from the equation, since it can be considered to be a part of β_j.

Nevertheless, the cases in which ε_{ij} cannot not be included in the perfect pattern equation are very difficult to study analytically. For this reason, we have performed a test to check the tendency of the VE^t values with regard to the error values. This test consist of the addition of random errors to a synthetic bicluster with perfect shifting and scaling patterns. The original bicluster is the one shown in Figure 4.3. Specifically, we have generated 100 synthetic biclusters adding random errors to the bicluster in the figure, and we have repeated this process 200 different times, varying the amplitude of the errors from one time to another. We start adding negative errors in the range of $[-10, 0]$, and obtain 100 different biclusters. Then we decrease the amplitude by 0.1 and repeat the process (range $[-9.90, 0]$). Once the amplitude of the errors has reached the zero value, we start again generating biclusters with positive errors, increasing the amplitude from 0.1 up to 10. The whole process produced 100 sets of 100 biclusters with negative errors and 100 sets of 100 biclusters with positive errors (built using the same strategy as for negative). Therefore, the random errors have been drawn from an uniform distribution corresponding to the ranges. Note that the type of the error values is a double type. This introduces more diversity in the distribution of the error data.

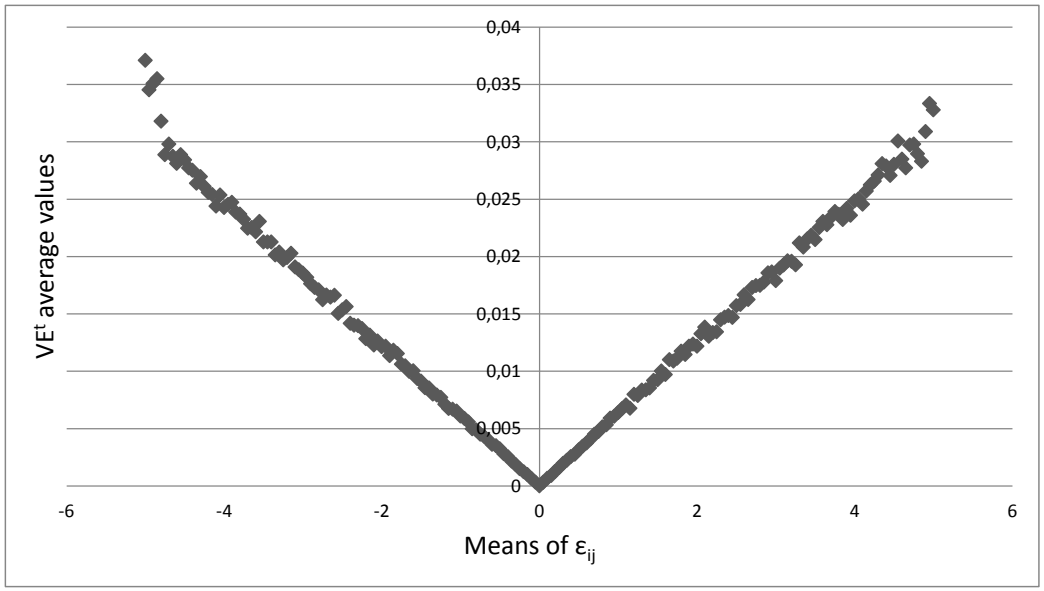


Figure 6.4: VE^t behaviour in biclusters with errors.

Within the process, we evaluate each produced bicluster using the three measures: MSR, VE and VE^t. Then we obtain the mean of each measure for each group of 100 biclusters of the same range of errors. This data has been represented in Figures 6.4, 6.5 and 6.6, where the x-axis represents the mean of the error for each amplitude (this value matches up with the value in the middle of the range of errors) and the y-axis corresponds to the mean of the specific measure for each figure.

From Figure 6.4 it is possible to observe that VE^t presents a linear decreasing tendency in relation to the amount of error in a bicluster. In other words, the similar a bicluster is to a perfect pattern, the lower its VE^t value will be, and we can establish a linear relationship between VE^t and the amount of error. Nevertheless, it is not possible to come to the same conclusion for either VE or MSR. Figures 6.5 and 6.6 depict the connection of the errors with VE and MSR, respectively. Although the general tendency seems to be that both measures are higher for biclusters with higher error values, we cannot establish any correspondence between them. In both figures it is possible to see some cases in which the mean of the biclusters with errors is lower than the original bicluster.

VE^t thus outperforms both MSR and VE efficiency for identifying behavioural patterns in synthetic data. Next chapter proves that VE^t is also efficient when working with real data from gene expression microarrays.

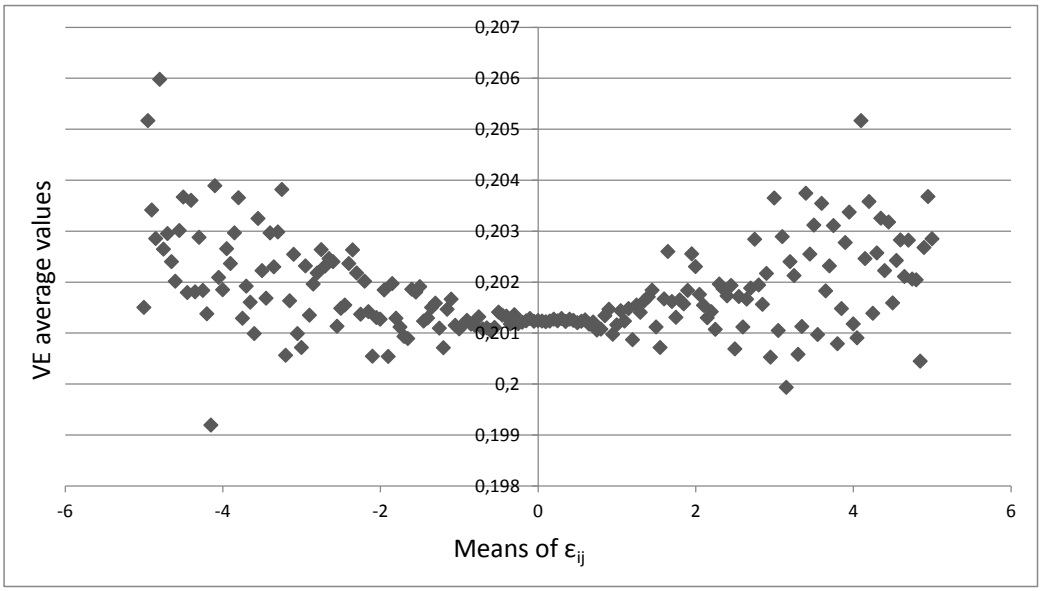


Figure 6.5: VE behaviour in biclusters with errors.

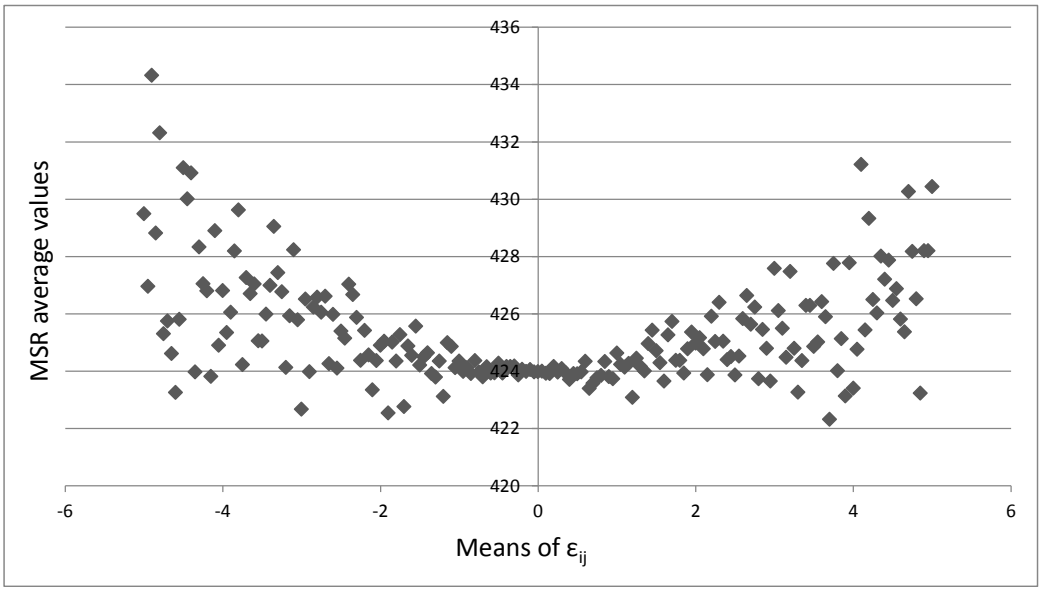


Figure 6.6: MSR behaviour in biclusters with errors.

6.6 Conclusions

In this chapter we have presented three different evaluation measures for biclusters, being all of them based on standardization procedures. The goal of an appropriate quality metric is to capture the tendencies of the expres-

sion levels according to the patterns described in chapter 4, where the more general situation has been proven to be depicted by the combined shifting and scaling pattern. The standardization mechanism we used allows the tendencies comparison, rather that their particular numerical values. Another advantage of using this mechanism is that there is no need of performing a previous standardization of the input data matrix.

According to the results of both analytical and experimental studies, the evaluation measure presented in last section (VE^t) is the more efficient one, since it is the only one able to capture shifting and scaling tendencies simultaneously in a bicluster. For a more elaborated discussion on the proposals in this chapter we refer the reader to the general conclusions in chapter 8.

Chapter 7

Evolutionary Biclustering based on Expression Patterns

In section 6.4, VE and MSR were incorporated into a multi-objective evolutionary biclustering algorithm. Nevertheless, a multi-objective approach based on the individuals dominances is not necessarily the best option for the biclustering problem. Using dominances, individual fitness is not conserved through generations. That is, individuals with the same genotype in different populations may not have the same fitness, due to the dominance concept effect on the evaluation. Furthermore, a multi-objective approach such as the one presented in the above chapter does not allow establishing any kind of prevalence among the different objectives. However, it may be interesting to prioritize some bicluster features over others. Depending on the microarray under study and also its biological or biomedical application, some users may prefer smaller biclusters but with a high gene variance, while other ones may prioritize the biclusters size over gene variance.

We present here a fully customizable evolutionary biclustering algorithm, named Evo-Bexpa (***Evo****lutionary* ***B****iclustering based on* ***Ex****pression* ***Pa****tterns*). VE^t has been used as the bicluster coherence evaluation measure, together with other objectives such as the bicluster volume, gene variance or overlapping level. The algorithm can be easily configurable towards obtaining results with the desired characteristics, according to the user preferences. Furthermore, new user-defined objectives can also be incorporated into the search without any difficulties, which makes our algorithm fully customizable. Moreover, Evo-Bexpa has been parallelized in two different spots which were the most expensive computationally.

Experiments on both synthetic and real datasets have been conducted, demonstrating Evo-Bexpa abilities to obtain meaningful biclusters. Synthetic experiments have been designed in order to compare Evo-Bexpa performance

with five other approaches when looking for perfect patterns. Experiments with four different real datasets also confirm the proper performing of our algorithm, whose results have been biologically validated through Gene Ontology.

7.1 Biclusters Evaluation in Evo-Bexpa

The problem of finding a single bicluster according to several objectives corresponds to a multi-objective optimization problem, in which two or more conflicting objectives need to be optimized. The strategy of constructing a single *Aggregate Objective Function* (AOF) has been adopted in order to solve this multi-objective problem. This way, it is possible to specify the relative relevance of each objective in the bicluster evaluation, allowing thus our algorithm to be configurable.

This section details the biclusters characteristics taken into account in their evaluation. In our approach we have individualised four different objectives, attending to the extent to which a bicluster follow a perfect correlation pattern, its size, overlapping amount among different solutions and mean gene variance.

Transposed Virtual Error (VE^t) explained in section 6.5 has been used as the quality measure for biclusters, being one of the most important objectives in the fitness. It is based on the concepts of expression patterns and quantifies the degree of correlation among genes in a bicluster. VE^t is always positive, being its optimal value equals to 0.

VE^t has been proven to be efficient to recognize both shifting and scaling patterns in biclusters either simultaneously or independently and it has also been proven to present a linear increasing behaviour when the amount of error in a bicluster gets bigger, measured according to the distance from its nearest perfect pattern.

7.1.1 Bicluster Volume

Bicluster volume is defined as the product of the number of genes and the number of samples. At this point we have two contrary objectives to be optimized. On the one hand, VE^t has to be minimized and normally the smaller a bicluster is, the lower VE^t will be. On the other hand, the volume has to be maximized and the general tendency is that bigger biclusters will have bigger values for VE^t. In order to design the volume term for the fitness we took into account the following issues:

- Use of a *logarithmic scale*. Little changes in the number of rows or columns would not have a significant effect, depending on the bicluster size.

- *Two separated terms* for number of genes and conditions. This is necessary for avoiding too unbalanced biclusters, but also desirable in order to allow to configure each dimension size independently. Note that biclusters in which one dimension is very small are more probable to be nearer to a perfect pattern, and therefore, they have low VE^t values. For this reason, it is preferable to optimize the size of both dimensions independently, thus avoiding obtaining biclusters made up of a great numbers of genes and only a few samples.

- *Fixed range*. The range of the values of the functions controlling both dimensions should not be dependant on any parameter value.

The final design of the term for the volume is the one shown in equation 7.1, where $|I|$ and $|J|$ refer to the number of genes and conditions, respectively, while w_g and w_c are the configuring parameters for both dimensions.

$$Vol(\mathcal{B}) = (\frac{-\ln(|I|)}{\ln(|I|) + w_g}) + (\frac{-\ln(|J|)}{\ln(|J|) + w_c}) \tag{7.1}$$

Those terms whose constant value (w_g or w_c) is greater decrease slower. Depending on the value of the constant used, the term will have more or less influence over the fitness function at the beginning of the algorithm, since initial biclusters are small and they grow along the evolutionary process. At a certain point, increasing the number of rows or columns for a certain solution would not compensate the lose of quality, according to the rest of objectives. The moment in which the algorithm stops increasing the size of the solutions and focuses on improving the quality depends on the value of the constants used. The smaller these constants are, the sooner the algorithm will stop increasing the size. Figure 7.1 represents the term for the number of genes in equation 7.1, for different values of the constant w_g. It can be clearly seen that for the smaller value of w_g represented ($w_g = 0.25$), the function decreases slower from a smaller value of the number of genes than for greater values of w_g.

Although we have found default values for the constants for both dimensions (rows and columns) that allow to obtain good solutions in every expression matrix we have tested, it is very easy to modify the fitness function in order to obtain solutions of different sizes if it is desirable. Increasing the constant associated to rows (w_g) will produce biclusters with greater

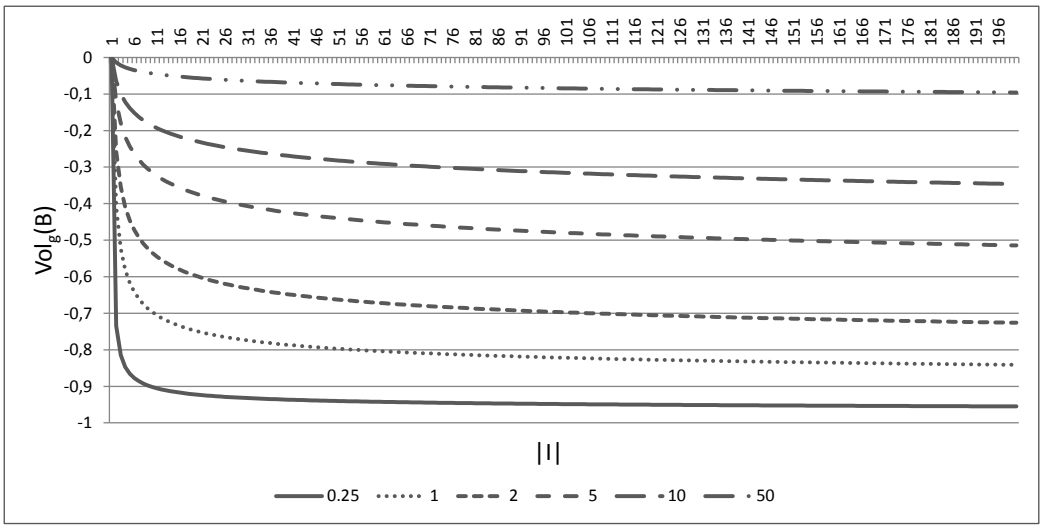


Figure 7.1: Genes size term in volume evaluation.

number of genes, while increasing the constant associated to columns (w_c) will produce biclusters with more experimental conditions.

7.1.2 Overlapping among Biclusters

Overlapping differs from VE^t and volume in the sense that it cannot be evaluated on a bicluster by itself.

As it has already been mentioned in previous chapters, overlapping among biclusters is usually permitted but controlled in the literature. The most popular way in which overlapping is controlled within the biclustering community is to replace the elements contained in each found bicluster with random numbers. This strategy was initially proposed by Cheng & Church (2000) and has been usually adopted in sequential approaches, where one bicluster is obtained at a time.

In order to avoid the problems derived from the random replacement strategy (see sections A.1.1 and A.3 in appendix A), in this work we have adopted a weight-based strategy , where a matrix of weights $\mathcal{W}$ the size of the microarray is initialized with zero values at the beginning of the algorithm. Every time a bicluster is found, the weight matrix is updated increasing by one those elements contained in the bicluster. In order to limit the overlap among biclusters, this matrix is used in the corresponding term of the fitness function as in equation 7.2, where I and J refers to the sets of rows and columns in the bicluster $\mathcal{B}$, respectively. $\mathcal{W}(e_{ij})$ corresponds to the weight

of e_{ij} in $\mathcal{W}$ and n_b is the order of the found bicluster. This way, we are being more permissive with the overlapping of latest biclusters. Appendix A includes a modification of the original CC algorithm (Cheng & Church (2000)) named CC-R, where this overlapping approach has been used to compare the results of CC algorithms with those of CC-R. CC-R follows a similar strategy to the one in CC, but replacing the random replacement by this weight-based approach.

$$Overlap(\mathcal{B}) = \frac{\sum_{i \in I, j \in J} \mathcal{W}(b_{ij})}{|I| \times |J| \times (n_b - 1)} \tag{7.2}$$

7.1.3 Gene variance

Biclustering was first defined by Hartigan (1972), although it wasn't applied to microarray data. The aim was to find a set of sub-matrices having zero variance, that is with constant values. Therefore, Hartigan used the variance of a bicluster to evaluate its quality. However, when working with gene expression data, it is preferable to obtain biclusters in which gene variances are high. This way, gene variance is used in biclustering of microarray data to avoid obtaining trivial biclusters, favouring those solutions in which genes exhibit high fluctuating trends. Gene variance of a bicluster is given by the mean of the variances of all the genes in it, as in equation 7.3.

$$GeneVar(B) = \frac{1}{|I| \cdot |J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} (b_{ij} - \mu_{g_i})^2 \tag{7.3}$$

Existing biclustering approaches deal with gene variance in different ways. For instance, Cheng and Church used a threshold value δ as an upper limit for their evaluation measure. This way, they search for biclusters with the maximum possible values for MSR below δ, rejecting thus trivial solutions in which there are no expression changes across the samples. Nevertheless, using such a limit presents a clear drawback, since δ has to be computed for each database before applying the algorithm (see section *Biclustering approaches based on Evaluation Measures*).

In our proposal, using VE^t as a single objective would produce biclusters in which gene variance is considerable low. However, if VE^t is combined with volume constraints favouring bigger solutions and overlapping control, the obtained results may not have so low variance. Despite this fact, we have also designed a term for controlling gene mean variance within the fitness function. This new term consists in the inverse of the gene mean variance

in equation 7.3, since biclusters with higher gene variances are preferred and the fitness is going to be minimized in the algorithm.

7.2 Evolutionary Algorithm

Evolutionary Algorithms (EAs) are classified as population-based meta-heuristics for combinatorial optimization, iteratively trying to improve a candidate solution with regard to a given measure of quality. In Evo-Bexpa, the measure of quality for biclusters will be given a combination of the objectives presented in the former section. EAs start with a set of possible solutions instead of a single one. This characteristic allow genetic algorithms to explore a larger subset of the whole space of solutions, at the same time as it helps them to avoid becoming trapped at a local optimum. These reasons made genetic algorithms very suited to the biclustering problem. Evo-Bexpa follows a sequential covering strategy, obtaining a single bicluster each time the evolutionary procedure (Bexpa) is executed. Therefore, it has to be run n times if n biclusters are desired, where n is an user-defined parameter.

Starting by an initial population representing different solutions, genetic algorithms select some individuals and recombine them to generate a new population of individuals. This process is repeated for a number of generations until the algorithm converges or certain criterion criteria is met.

Algorithms 5 and 6 show the pseudo-codes of both the sequential (Evo-Bexpa) and genetic (Bexpa) strategies, respectively. Evo-Bexpa (Algorithm 5) consist in iteratively invoking Bexpa as many times as biclusters are desired from an input matrix $\mathcal{M}$. An initially empty list L is used to store the biclusters returned after each Bexpa invocation (line 1 in Algorithm 5). The matrix of weights $\mathcal{W}$ explained in section for the overlapping control is also initialized with zero values in line 2. This way, after a new bicluster is obtained in the n_b iteration (line 5), it is stored in the list L (line 6) and $\mathcal{W}$ is also updated by incrementing in one unit those elements contained in the bicluster in $\mathcal{W}$. Finally, when the n biclusters have been found, the whole list is returned by Evo-Bexpa. Note that the order of the biclusters in the

output list does not reflect their quality nor their biological relevance.

Algorithm 5: Function **Evo-Bexpa**

input : $\mathcal{M}$: expression matrix,
n: number of biclusters
output: L: list of n biclusters

```
1 L ← {};
2 n_b ← 1;
3 matrix of weights W ← {};
4 while n_b <= n do
5     bicluster b ← Bexpa(M,W,n_b);
6     L ← L ⊕ b;
7     updateWeightMatrix(W,b);
8     n_b ← n_b + 1;
9 return L;
```

Algorithm 6: Function **Bexpa**

input : $\mathcal{M}$: expression matrix,
$\mathcal{W}$: matrix of weights,
n_b: order of bicluster
output: b: new bicluster

```
1  Pop ← initializePopulation();
2  while stopping criteria are not met do
3      NextPop ← {};
4      NextPop ← NextPop ⊕ best(Pop);
5      NextPop ← NextPop ⊕ mutate(best(Pop));
6      repeat
7          parents ← selectForCrossOver(Pop);
8          offspring ← crossover(parents);
9          mutateWithProbability(offspring);
10         NextPop ← NextPop ⊕ offspring;
11     until numCrosses is reached;
12     repeat
13         offspring ← selectParentForReplicate(Pop);
14         mutateWithProbability(offspring);
15         NextPop ← NextPop ⊕ offspring;
16     until numReplicates is reached;
17     Pop ← NextPop;
18 b ← best(Pop);
19 return b;
```

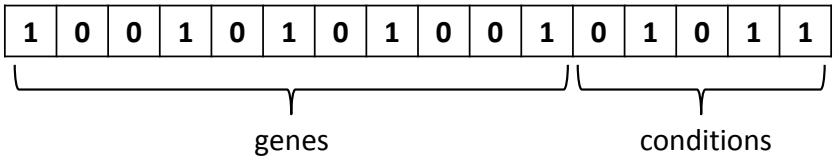


Figure 7.2: Binary string for individuals representation.

Bexpa (Algorithm 6) starts with the initialization of the population in line 1, followed by an iterative process for the search of a single bicluster. The whole process consist in repeatedly replace the current population with an evolved one, until certain criteria is met (lines 3 to 17). Each of this replacements is called a *generational change*, and it is explained in section 7.2.2, after the individuals encoding and initialization strategies are introduced.

In our Bexpa implementation, initial population and generational change strategies have been parallelized to improve time performance, aiming at maximizing the parallelization factor, represented by the time consumed of the equivalent non-parallel algorithm divided by the time consumed by the parallel one. In both cases, there exists no communication among the different activities, thus reducing synchronization and coordination times. Furthermore, we have no shared resources for writing, thus avoiding concurrent accesses.

7.2.1 Individual Encoding and Initialization

The first task when choosing a genetic algorithm for solving any problem is to decide an appropriate individual or chromosome representation for the possible solutions. We have adopted the same individual representation in other evolutionary biclustering works, where each bicluster is represented by a fixed sized binary string in which a bit is set to one if the corresponding gene or sample is present in the bicluster, and set to zero otherwise. Figure 7.2 shows an example of a bicluster individual made up of 5 genes and 3 experimental conditions, while the microarray it has been extracted of contains 11 genes and 5 conditions. This way, indexes from 1 to 11 in the string representation refer to the same indexes of the genes in the microarray. Nevertheless, condition indexes in the microarray might be computed by the difference of the total number of genes. For example, position 13 of the bicluster individual in Figure 7.2 corresponds to the second experimental condition in the microarray.

Initial population procedure is also essential in every evolutionary algo-

rithm. Depending on the adopted strategy, the algorithm may converge to different solutions. Also, a suitable initial population strategy can even speed up the convergence (Toğan & Daloğlu (2008)).

Other evolutionary biclustering approaches have adopted a totally random initial population generation (Mukhopadhyay et al (2009a)), where initial solutions are made up of a random number of elements (genes and samples) randomly chosen from the microarray, or also random strategies in which the chromosomes are made up of only one element (one gene and one condition)(Divina et al (2012)) from the microarray. In our experimental tests on synthetic data, we found that these kind of initializations did not give the algorithm an initial space solution good enough to come up to the best solution. Nevertheless, our algorithm always converged to the best solution when the initial population contained at least one 3×3 sized bicluster representing a partial solution. That is, this 3×3 sized bicluster is a submatrix of the solution. Therefore, the initial strategy we have adopted consists in randomly generating individuals which represent 3×3 sub-matrices, henceforth seeds. The key is to generate much more seeds than the size of the population and then select the best ones. In fact, it is quite easy to compute the number of seeds needed to increase the probability that some of them are part of the solution, if it is known beforehand. The probability of a randomly generated seed to be part of the solution can be computed as the number of possible seeds in the solution divided by the number of possible seeds in the whole data matrix, as in equation 7.4, where N, M, $|I|$ and $|J|$ are the number of rows and columns of the microarray data matrix and the solution, respectively.

$$\frac{Favorable\_seeds}{Total\_seeds} = \frac{\binom{|I|}{3} \times \binom{|J|}{3}}{\binom{N}{3} \times \binom{M}{3}} \tag{7.4}$$

Thus, our algorithm computes the number or seeds needed in order to at least one of them is a part of the solution. This procedure can only be performed with synthetic data, but it also gives us an idea of the number of seeds to generate in the case of real data.

The initialization procedure is carried out once per bicluster. Nevertheless, when looking for very small biclusters, or when the size of the microarray is very big, this initialization could take a very long time. We have therefore parallelized this phase using a pool of threads the size of the available cores in the computer, creating afterwards as many activities as cores. The task of each activity will be to generate a number of seeds such that the total number of seeds are generated by all the activities. This activities are therefore submitted to the pool of threads and the results are gathered up once they

have finished.

7.2.2 Generational Change

Generational change is the mechanism that allows the population to improve its individuals, according to the fitness function and trying to converge to the optimal solution (lines 3 to 17 in algorithm 6). For each generation, the new population is formed by incorporating individuals from the previous one in several ways: replicating themselves, being mutated, being crossed with other(s) individual(s) or combining some of these operators.

The next population in Bexpa is created by firstly adding the best individual of the current population to the next one, as it can be seen in line 4 of Algorithm 6. This process is called *elitism* and is usually applied in order to ensure the convergence of the algorithm (Coello (2006)). Also, a mutated copy of the best individual is incorporated into the next population (line 5). The rest of individuals are generated by selecting one or two individuals and applying crossover or/and mutation. Selection is based on the use of the fitness function together with a random component. In our approach, we have used tournament of size 3 as selection mechanism (Floreano & Mattiussi (2008)). A given percentage of the remaining individuals are generated by the crossover of two previously selected chromosomes (lines 6 to 10), while the rest of individuals that will complete the population correspond to replications (lines 11 to 14). The resulting offspring is mutated with a certain probability in both cases.

Three distinct crossover operators are used in our algorithm with equal probability: one-point crossover (Figure 7.3), two-points crossover (Figure 7.4), and uniform crossover. The first two operators select one or two positions at random, respectively, and interchanges the genetic code of the individual chromosome of both parents for creating the offspring, using the random positions as delimiters. In Figure 7.3, the first three elements of the offspring are taken from the first parent, while the rest of the elements are inherited from the second one, being the crossover position the third one. Figure 7.4, on the other hand, represents the two-points crossover, in which two different positions are used to create the new offspring. Elements before the first position and after the second one are taken from the first parent, while the elements in the middle are inherited from the second. In the uniform crossover, each position of the offspring is copied from one of the two parents, being both possibilities equally probable.

We have also applied two different mutation operators: the simple and the uniform ones, being the probability of the uniform mutator is much lower due to the fact that every position of the bit string is a candidate to be mutated in

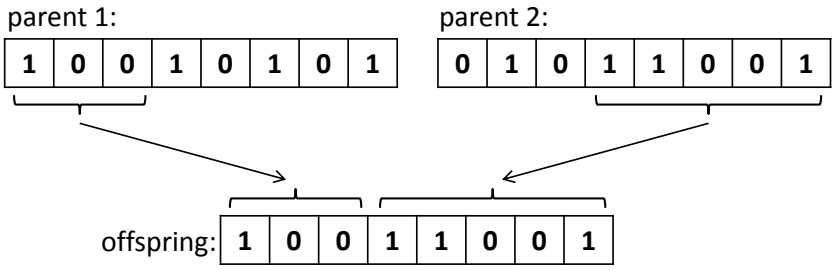


Figure 7.3: One-point crossover.

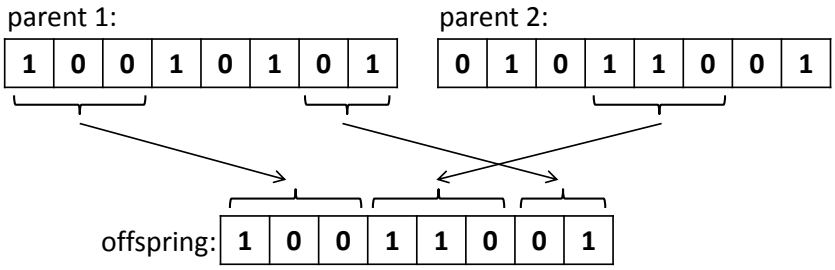


Figure 7.4: Two-points crossover.

the uniform mutation. Simple mutation, on the other hand, selects a random position of the individual chromosome and is mutated or not depending on the given probability. In both kind of mutations, if the selected candidate is finally mutated, its value is modified from 0 to 1 or the other way round, depending on its original value.

Bexpa iterates for a predefined number of generations, although if there is no significant improvement after a certain number of consecutive generations, the execution is stopped. Crossover and replications percentages, as well as mutation probabilities and the number of generations have been set experimentally, although all of them are input parameters for the evolutionary algorithm and can be modified by the user.

Computing the properties of the individuals detailed in section 7.1 is the most time-consuming task in the algorithm, since performing the evaluation of each chromosome implies mathematical operations over big-sized bit strings. In order to avoid unnecessary repeated computations, the evaluation is performed every time a new individual is produced, and all their values are stored as internal properties until the individual is discarded. For example, if a certain individual is replicated to the next generation without mutation, its fitness is the same and there is no need to compute it again. Fortunately, the whole generational change process can be implemented using parallel

computing, allowing us to parallelize the whole generational change process. We have used again a pool of threads the size the number of cores in the computer. In this case, each activity submitted to the pool would be responsible for the creation of a new chromosome, together with its evaluation.

7.2.3 Fitness Function

After the different objectives taken into account in the bicluster evaluation have been introduced, here we present the way in which they have been combined to form the fitness function used in our algorithm for the evaluation of the potential solutions. Although we have used the four different objectives described above in our experiments, the fitness function is easily configurable by adding new objectives in the form of a mathematical formula.

In the context of evolutionary algorithms, the fitness function is a particular type of objective function used to summarise, as a single figure of merit, how close a given design solution is in order to achieve the set aims.

Equation 7.5 depicts the final fitness function used in our algorithm. Note that the goal is to minimize the value of every term, in order to find big-sized biclusters with a low value of VE^t, high gene variance and hardly overlapped.

$$\begin{aligned}\Phi(\mathcal{B}) = \frac{VE^t(\mathcal{B})}{VE^t(\mathcal{M})} + w_s \cdot Vol(\mathcal{B}) + w_{ov} \cdot Overlap(\mathcal{B}) \\ + w_{var} \cdot \frac{1}{1 + GeneVar(\mathcal{B})}\end{aligned} \tag{7.5}$$

Every term is weighted, except VE^t which acts as the reference objective. Nevertheless, the value of VE^t for the bicluster has been divided by the VE^t value of the whole microarray. This is due to the fact that the range of values of VE^t depends of the values in each microarray. Although the algorithm pursuit to minimize it, the weight of the other terms of the fitness function would have to be recomputed when using a different microarray. In order to avoid this situation, we divide it by the VE^t value of the whole microarray ($\mathcal{M}$ refers to the microarray data matrix).

Modifying the weights associated to the different objectives leads the algorithm towards different kind of biclusters, according to their sizes, overlapping amount or gene variance. All weights have been designed in the same way; a lower value of a certain weight will result on biclusters with lower values for the corresponding characteristic, and vice versa. For example, a lower value of w_s will lead to small-sized biclusters, while bigger values of w_s will result on big-sized biclusters. In the results section we provide default values for every weight, which have been obtained experimentally and have

Parameter	Value
Generations	1500
Max. Generations without sign. improvement	150
Population size	200
Crossover probability	0.80
Mutation probability	0.20
Tournament size	3

Table 7.1: Evolutionary parameter setting for Bexpa.

produced meaningful results for all the databases under study. Also, we provide the user with a guidance on how the modification of the weights affect the different characteristic of the obtained biclusters.

Note that it is quite simple to add new objectives to the fitness. A new mathematical formula should be designed for each new bicluster feature to be taken into account. This formula will be minimized when inserted into the fitness function, and will also have a corresponding weight. In order to better control the effect on the results, it is preferable that the range of values were fixed, not dependant on the specific values of the microarray or bicluster.

7.3 Experimental Results and Discussion

This section presents a wide set of experiments performed to test the validity of Evo-Bexpa, both on synthetic and real data sets. The results have been compared with those obtained using five different approaches: OPSM(Ben-Dor et al (2003)), ISA(Bergmann et al (2003); Ihmels et al (2004)), xMotifs(Murali & Kasif (2003)), CC(Cheng & Church (2000)) and Bimax(Prelić et al (2006)) (see section 4 for a description of each approach). All these five algorithms have been executed using BicAT (*Biclustering Analysis Toolbox*)(Barkow et al (2006)).

Next subsection presents an in-depth analysis on the performance of Evo-Bexpa when modifying the different configuration parameters introduced in section 7.1, as well as a study on the different parameters for the algorithms in BicAT. In subsequent subsections, experiments carried out on both artificial and real data sets are described. In order to perform a fair comparison, we use the same evolutionary parameter setting in Bexpa, corresponding to the most common configurable parameters in any evolutionary algorithm. This setting is given in Table 7.1. The values of these parameters were obtained after a number of preliminary runs on synthetic data sets aimed at testing different parameter settings.

7.3.1 Analysis of Parameters

Each biclustering approach needs different parameters to run. Although default parameters are provided which should guide the algorithms towards reasonable results, there is no detailed description on how their variations affect the obtained bicluster, for any of them. In this subsection we first describe the input parameters for each of the algorithms in BicAT (OPSM, ISA, xMotifs, CC and Bimax), trying to clarify the characteristic of the resulting biclusters affected by the modification of the different parameters. After that, we present a study on the parameter sensitivity for Evo-Bexpa.

Analysis of Parameters for Algorithms in BicAT

Bimax uses an underlying binary data model which assumes two possible expression levels per gene. Therefore, as a preprocessing phase, it is compulsory to discretize the expression values to binary values at a specific threshold and with a specific scheme. All values above the threshold will be set to one, all those below to zero. The discretization scheme defines if only down or up-regulated genes (or both) will be considered.

The algorithm also takes as input parameters the minimum number of genes and samples for the output biclusters. By specifying larger lower bounds, fewer biclusters will be returned, reducing thus the computing time. Default values for both the minimum number of genes and conditions have been set to two.

CC algorithm takes as input parameter two different thresholds, δ as the upper limit for MSR, which has already been mentioned, and $\alpha > 1$ as a threshold for the multiple node deletion phase. δ presents two main drawbacks: its value depends on the input microarray and has to be computed beforehand (there is no common default value), and also the use of δ blocks the algorithm from obtaining meaningful solutions Aguilar-Ruiz (2005); Pontes et al (2009). Default value for α parameter has been set to 1.2, and the authors claim that when it is properly selected, the multiple node deletion phase is usually extremely fast. Nevertheless, there is no explanation on how does this value affect the results. There are no criteria for finding an efficient value for α either.

CC also receives as an input parameter the number of biclusters to obtain, since it is based on a sequential covering strategy, as well as Evo-Bexpa.

OPSM approach is based on the formulation of a probabilistic model of the expression data. As finding the best model is infeasible for real data, Ben-Dor et al. use partial models and grow them iteratively. The algorithm takes as input paramater the number of partial models passed for each iteration ℓ.

According to the authors, increasing ℓ would improve results, although it will come at a cost of a higher running time. Nevertheless, it is no clear in which aspect does the modification of ℓ affect the obtained biclusters (size, quality or other) in real data. Furthermore, they do not provide any instruction on how to select an appropriate value for ℓ.

The *Iterative Signature Algorithm* (ISA) receives three different input parameters. T_g and T_c are thresholds for the resolution of the modular decomposition of both genes and conditions, respectively. T_c is said to have a minor effect on the results, and was set to 2 in all the analyses. T_g was varied from 1.8 to 4.0 in steps of 0.1, in order to analyse the resulting stringency of co-regulation between the genes. The default value for T_g can be assumed as 2.0. Although the authors perform an analysis on the influence of T_g on the results on a specific datasetIhmels et al (2004), it is not straightforward to see what will the influence be for any other datasets.

The third input parameter for ISA is the number of starting points that the algorithm uses for randomly selecting a set of genes and iteratively refining this set until the genes and conditions in it are mutually consistent and match the definition of a transcription module. Authors claim that using a sufficiently large number of initial sets it is possible to determine all the modules corresponding to a particular pair of thresholds. The default value for this parameter is set to 100.

xMOTIFs looks for biclusters in which genes are expressed in the same state across all samples. In order to differentiate biologically interesting states, a maximum p-value parameter is used, considering only those states whose p-value is less than the parameter $(1 \times 10E - 9)$. Another parameter α determines the minimum number of samples for biclusters, given as a fraction of the total number of conditions, being its default value 0.05. Murali and Kasif also make use of inner parameters to the algorithm such as the number of seeds (n_s), the number of determinants (n_d) and the size of the discriminating set (s_d), as in Procopiuc et al (2002). The authors claim that the quality of the results does not change much when those are slightly varied.

Analysis of Evo-Bexpa Parameter Sensibility on Real DataSets

Input parameters for Evo-Bexpa were detailed in former sections. The number of parameters will depend on the number of objectives or bicluster characteristics to optimize. In this approach, we have used 5 different configuration parameters, which control the volume (w_g, w_c and w_s), the amount of overlapping (w_{ov}) and the gene variance (w_{var}).

Default values for Evo-Bexpa have been set experimentally by using

a benchmark database and trying to reproduce previous results for this database in the literature. Also, solutions with low proportions of the number of genes and high percentage of the total number of samples have been favoured for the setting.

In order to deduce the parameter influence on each characteristic, we have tested Evo-Bexpa modifying each configuration parameter from −100% to +100% its value, in intervals of ±25%. Table 7.2 shows all the used values, where the central row gives the default ones. So it means running Evo-Bexpa 8 additional times per parameter, using each value of table 1 for each weight, while maintaining the other weights at their default values, this represents 41 experiments for each dataset. Furthermore, we have chosen four different microarrays to study the parameters influence in diverse scenarios (see table 7.4). All in all, for the purpose of the parameter influence analysis, a total of 164 experiments over real datasets have been carried out, being 100 the number of biclusters to be obtained in each execution. All weights must have a positive value, being 0.0 the value for which the corresponding objective exerts no influence on the results. However, they can be set to any positive value, even above +100% their default values, if more influence of any bicluster characteristic is desired.

	w_g	w_c	w_s	w_{ov}	w_{var}
−100%	0.0	0.0	0.0	0.0	0.0
−75%	0.0625	0.125	1.25	1.25	0.025
−50%	0.125	0.25	2.5	2.5	0.05
−25%	0.1875	0.375	3.75	3.75	0.075
Default	0.25	0.5	5.0	5.0	0.1
+25%	0.3125	0.625	6.25	6.25	0.125
+50%	0.375	0.75	7.5	7.5	0.15
+75%	0.4375	0.875	8.75	8.75	0.175
+100%	0.5	1.0	10.0	10.0	0.2

Table 7.2: Experimental values for configuration parameters

In the following, parameter analysis is only presented for *Embryonal tumours of the central nervous system* dataset (Pomeroy et al (2002)), in view of results for the other datasets are similar and do not contribute any novelty to the study.

Figures 7.5 to 7.9 represent the variations of the means and deviations for all the different objectives (VE^t, number of genes, number of conditions, overlap and gene variance) when modifying each configuration parameter. Each figure is made up of four different graphics which depict the influence

of a certain weight over the aforementioned bicluster aspects. The main graphic of each figure shows the variations of the means and deviations for the main aspect affected by the weight modifications. Abscissa axis refers to the specific weight values, according to table 7.2, while the ordinates axis depends on the configuration parameter under study. For example, in the first three figures (7.5 to 7.7), vertical axis corresponds to the means of the number of elements in the biclusters (genes or samples). At the right side of the main graphic, the way in which the variations of the parameter affects the other characteristics has also been represented.

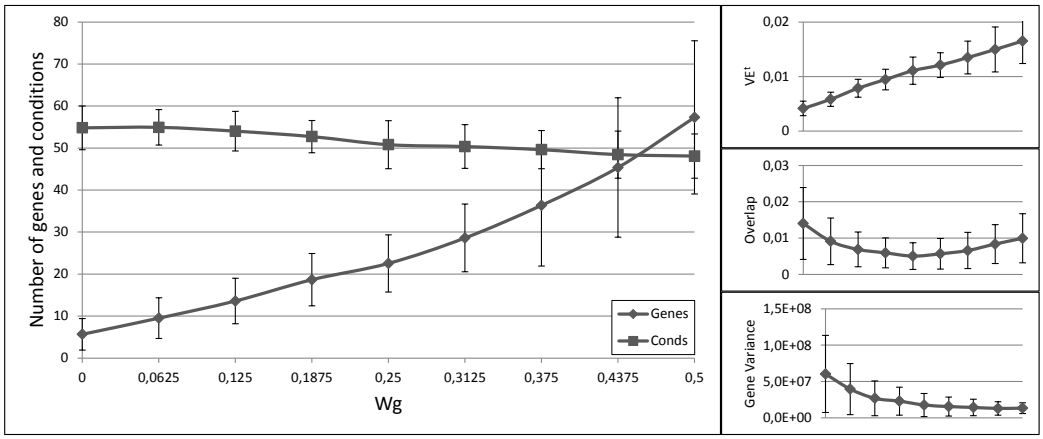


Figure 7.5: w_g influence over the different bicluster features.

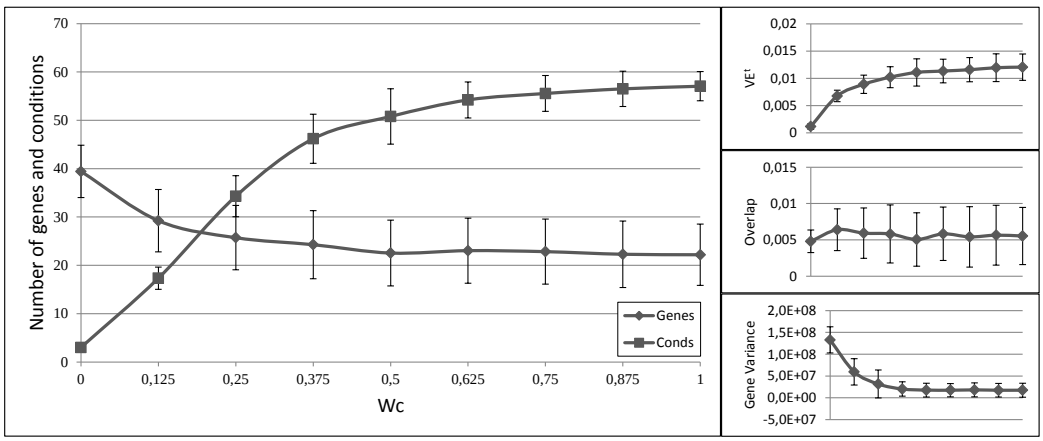


Figure 7.6: w_c influence over the different bicluster features.

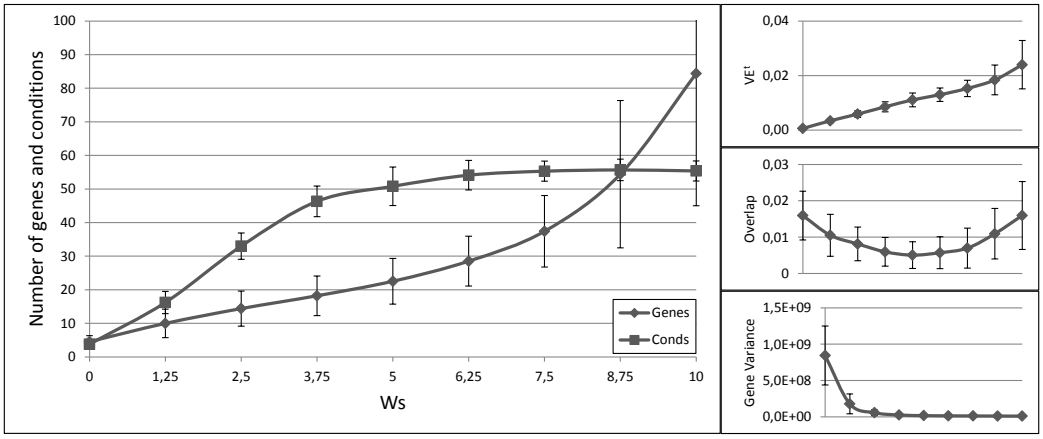


Figure 7.7: w_s influence over the different bicluster features.

Means of the 100 biclusters represent the general tendency of the results. Nevertheless, deviations cannot be disregarded. This is due to the fact that although it is possible to favour some properties in the solutions, results provided by Evo-Bexpa are diverse, obtaining thus biclusters in which their properties vary in a range around the reported mean.

Although the modification of any configuration parameter not only affects its corresponding aspect, it can be clearly seen that the greatest variations in any characteristic are obtained by increasing or decreasing its associated weight. Furthermore, some objectives are related in a negative way. Mean gene variance, for instance, would be decreased if bicluster size is increased or the overlap decreases. Therefore, it would be a good practice to slightly correct gene variance parameter when size or overlap parameters are adjusted, or vice-versa. Other characteristics have different behaviours when adjusting any other weights. The mean of the number of genes and conditions is quite stable when modifying W_{ov} in Figure 7.8, as well as overlap mean when w_c is adjusted, in Figure 7.6. In general, VE^t increases whenever greater sizes or less overlapping is preferred. It was the expected behaviour, since bigger solutions would produce higher values of VE^t, unless they were closer to a perfect combined pattern. On the contrary, biclusters with higher mean gene variance would have lower values of VE^t, due to the reduction of their sizes when higher variances are required.

Table 7.3 presents a summary of the configuration parameters influences over the different bicluster characteristics. Each row represents the influence of each configuration parameter in the first column over the characteristics in the first row, where the behaviour of each row has been observed when

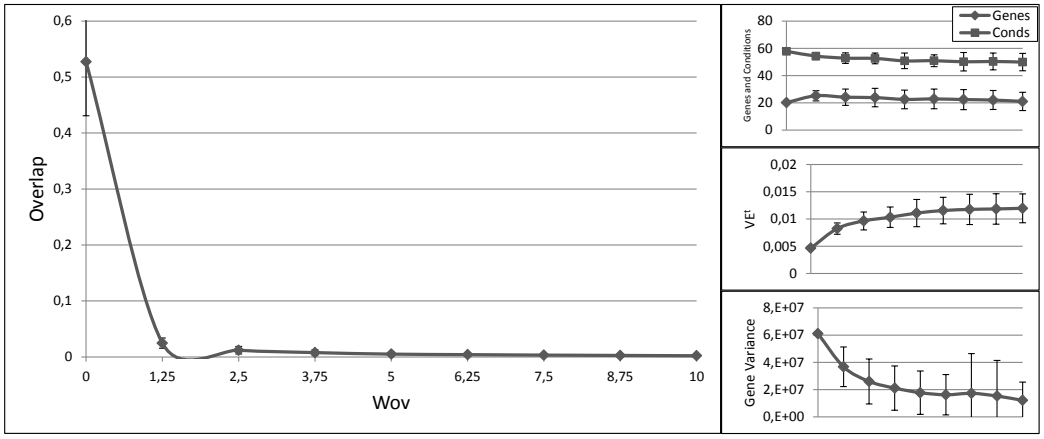


Figure 7.8: w_{ov} influence over the different bicluster features.

increasing the corresponding weight from 0.0 to its maximum experimented value (see Table 7.2). Symbol $\uparrow\uparrow$ represents large increments, $\uparrow$ medium increments and $\downarrow\downarrow$ large decrements. Symbol $=$ stands for no significant variations, $\searrow\nearrow$ depicts a decreasing behaviour for low values of the weight, turning to increasing for higher values of the parameter, while $\nearrow\searrow$ depicts the contrary situation.

This table has been elaborated using the four real datasets in Table 7.4 and the aforementioned variations of the weights. This way, Table 7.3 represents the common behaviour observed in all the datasets under study.

weight	VE^t	#genes	#conditions	overlap	mean gene variance
w_g	$\uparrow\uparrow$	$\uparrow\uparrow$	$=$	$\searrow\nearrow$	$\downarrow\downarrow$
w_c	$\uparrow$	$\searrow=$	$\uparrow\uparrow$	$=$	$\downarrow\downarrow$
w_s	$\uparrow\uparrow$	$\uparrow\uparrow$	$\uparrow\uparrow$	$\searrow\nearrow$	$\downarrow\downarrow$
w_{ov}	$\uparrow$	$=$	$=$	$\downarrow\downarrow$	$\downarrow\downarrow$
w_{var}	$\downarrow\downarrow$	$\downarrow\downarrow$	$\nearrow\searrow$	$\searrow\nearrow$	$\uparrow\uparrow$

Table 7.3: Qualitative influence of the configuration parameters over the different objectives.

In short, Evo-Bexpa parametrization allows the user to specify preferences on biclusters features, by adjusting the corresponding weight(s). The recommended procedure consist in first run the algorithm using the default configuration, correcting afterwards those weights needed to reach the desired results in terms of the objectives. In order to select an appropriate correction, Figures 7.5 to 7.9, together with the information in Table 7.3

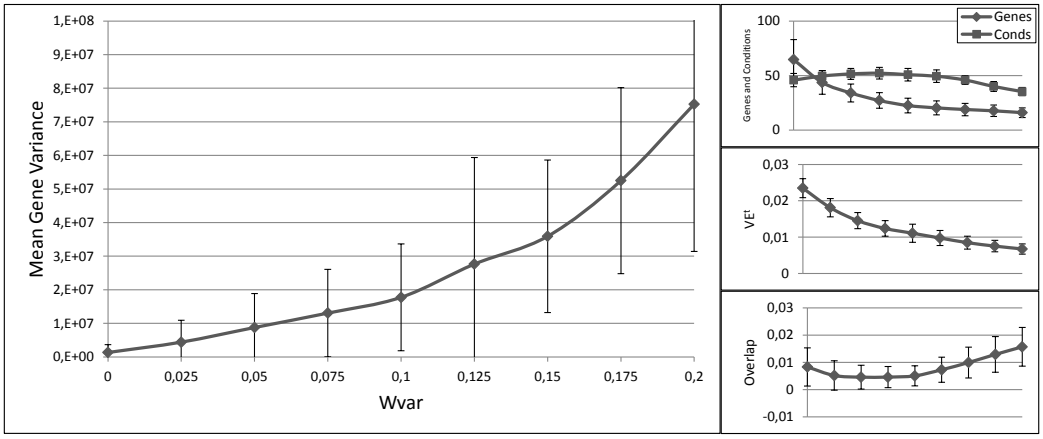


Figure 7.9: w_{var} influence over the different bicluster features.

should be used, being aware of the implications that each weight variation has on the other bicluster aspects.

7.3.2 Synthetic Data Experiments

In order to test the effectiveness of Evo-Bexpa to find biclusters following shifting and scaling patterns, we have carried out a set of experiments inspired on the works of Mukhopadhyay et al (2009b) and Bozdağ et al (2010), where perfect synthetic biclusters with shifting or scaling tendencies were inserted into artificial data sets. In a more general purpose, we have used combined patterns (shifting and scaling simultaneously) for biclusters generation. These biclusters have been hidden in several artificial data matrices, with uniform random distributions.

We have chosen the size of one of the most tested benchmark microarrays in biclustering: yeast *Saccharomyces cerevisiae cell cycle expression* dataset Cho et al (1998), made up of 2884 genes and 17 samples, for the generation of artificial matrices. We have also defined several sizes (genes $\times$ conditions) for the inclusion of perfect biclusters: 20×10, 60×12, 100×13, 150×15 and 200×16. For each of these sizes we have generated a perfect bicluster according to a combined shifting and scaling pattern. Each of these 5 different sized perfect biclusters has been inserted into 5 different random in silico microarrays in random positions. Thus, a total of 25 different case studies constitute the first set of experiments, in which no noise has been introduced.

Furthermore, we have also generated the same number of experiments adding noise to the data with random values generated from normal distribution, with mean equals to 0 and deviation equals to 0.25. All in all,

there are 50 different experiments, 25 in which the biclusters follow a perfect pattern and 25 in which random noise has been included into the data.

For each of the experiments, we have run the following 6 different biclustering algorithms: OPSM, ISA, xMotifs, CC, Bimax and Evo-Bexpa presented in this work. We have used default parameters to run all of them.

In order to check the extent to which the bicluster obtained by each algorithm adjusts to the solution we have used match scores indexes for both genes and conditions (Prelić et al (2006)) as performance measure. Let $\mathcal{B}_1(I_1, J_1)$ and $\mathcal{B}_2(I_2, J_2)$ be two biclusters, then gene match score is defined as $S_I(I_1, I_2) = \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|}$ and condition match score is defined as $S_J(J_1, J_2) = \frac{|J_1 \cap J_2|}{|J_1 \cup J_2|}$. Both indexes vary from 0, when both set of genes (or conditions) are disjoint, to 1, when the sets totally match. This way, match score indexes can be use to compute the degree of similarity of the sets of genes and conditions of two biclusters. We have, therefore, compare each bicluster obtained with the corresponding solution for all the executions using the six former algorithms.

Figure 7.10 displays the gene and condition match scores of the executions of the six algorithms. X-axis represents gene match scores and Y-axis represents condition match scores. Each dot in the graphic refers the comparison of a bicluster found by each algorithm and the equivalent solution. According to the gene and condition match scores definitions, the dots in the right top corner of the graphic correspond to those obtained biclusters which have a better match with its equivalent solution. OPSM, CC and Evo-Bexpa are the algorithms with better results. In the case of Evo-Bexpa, there exists exactly five biclusters which are not correctly found, and whose scores indexes are below 0.6 and 0.3 for the conditions and genes sets respectively. We have studied these results and have found that they correspond to the five experiments in which the hidden biclusters are smaller (20×10) and noise has been introduced. Only OPSM finds better solutions than Evo-Bexpa in these experiments, while for the other cases Evo-Bexpa outperforms both OPSM and CC. In fact, we have conducted a statistical test which confirms that Evo-Bexpa outperforms the other five algorithms in finding perfect shifting and scaling behaviours in synthetic data.

Match Score can also be used for measuring the degree of similarity of two biclusters using former genes and conditions match scores indexes. This way, we have used the bicluster match score index in order to rank the effectiveness of the algorithms. Bicluster match score is defined as $\sqrt{S_I(I_1, I_2) \times S_J(J_1, J_2)}$, and varies from 0, when the biclusters $\mathcal{B}_1$ and $\mathcal{B}_2$ are disjoints, to 1, when $\mathcal{B}_1$ and $\mathcal{B}_2$ completely match.

Since our results do not follow a normal distribution, we have applied Friedman as a non-parametrical test to carry out a comparison which involves

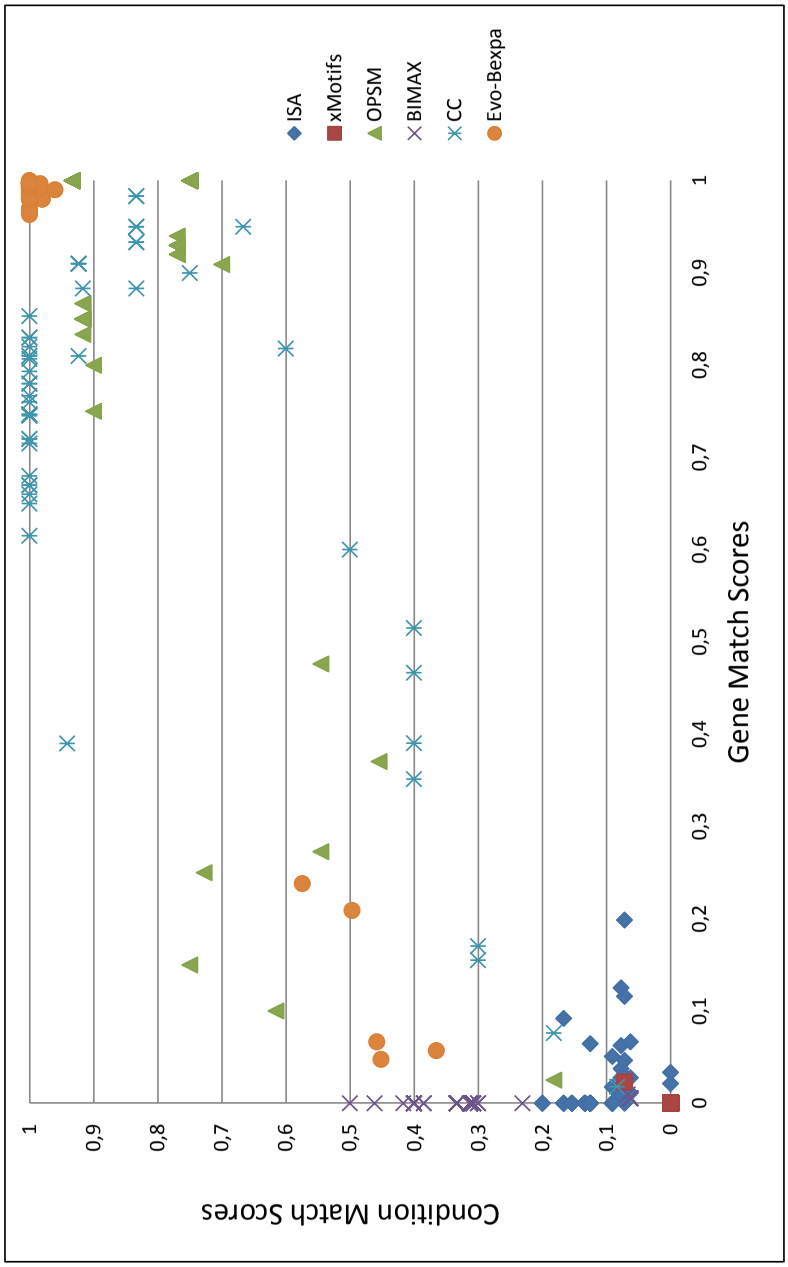


Figure 7.10: Gene and condition match scores for ISA, xMotifs, OPSM, BIMAX, CC and Evo-Bexpa in synthetic experiments.

six different methods. Friedman test ensures us that the results obtained by the six algorithms are statistically different, with a p-value of 1.16^{-10}. Also, the raking provided by Friedman suggests the following order: Evo-Bexpa, OPSM, CC, ISA, Bimax, xMotifs, which seems to be in concordance with the representation in Figure 7.10. Furthermore, we have also performed a post-hoc procedure in order stablish a comparison two by two using our algorithm as the control method. In this comparison, we obtained for each of the other five algorithms a p-value less than the alpha values returned by five different post-hoc procedures (Holm, Holland, Rom, Finner and Li), which certifies that our proposal Evo-Bexpa outperforms the other five algorithms in this empirical study with a significance less than 0.05. STATService (Parejo et al (2012)) has been used in order to perform these statistical tests.

7.3.3 Experiments on Real DataSets

Experiments on four different real microarrays have been conducted using Evo-Bexpa and the five algorithms contained in Bicat toolbox: OPSM, CC, ISA, Bimax and xMotifs. Table 7.4 specifies the details of the datasets, including theirs sizes as well as references to their corresponding publications. Yeast dataset is the smallest, made up of 2884 genes and 17 samples, and represents one the most used dataset for comparison of biclustering techniques. In fact, it is considered as a benchmark dataset for many researches. Leukemia dataset is the one containing the higher number of samples, while Steminal acts as the most unbalanced microarray, with the mayor number of genes (26127) and only 30 samples.

Dataset	Name	#genes	#cond.	Ref.
Yeast	Yeast *Saccharomyces cerevisiae* cell cycle	2884	17	Cho et al (1998)
Embryonal	Embryonal tumors of the central nervous syst.	7129	60	Pomeroy et al (2002)
Leukemia	Leukemia	7129	72	Golub et al (1999)
Steminal	Steminal Cells	26127	30	Boyer et al (2006)

Table 7.4: Datasets used in the experimentation.

Table 7.5 presents the results for each dataset and algorithm using default parameters in all cases. Results are represented by the number of biclusters obtained, means and deviations of their volume (number of genes and experimental conditions), and means and deviations of their VE^t values and gene variance. Results have also been grouped by dataset, being the first five rows those corresponding to the executions of OPSM, ISA, CC, Bimax and Evo-Bexpa for Yeast microarray, respectively. Unfortunately, Bicat implementations of xMotifs and Bimax approaches did not work properly for every dataset in Table 7.4. Specifically, xMotifs could not be performed for

Yeast and Steminal datasets, due to unexpected runtime errors. xMotifs could neither be executed for Leukemia dataset, since it does not support more than 64 samples, according to Bicat toolbox. In the case of Bimax, we did not obtain results for either Embryonal, Leukemia or Steminal datasets in reasonable time. This fact might be related to the datasets sizes, since the mean of the biclusters sizes for the Yeast dataset using Bimax is greater than 75% of the size of the whole microarray, and the computational cost of generating quality biclusters of similar proportions for the other datasets may be unfeasible. Nevertheless, Bimax has run properly for the Yeast dataset and synthetic data matrices of the same size.

Dataset	Algorithm	NumBic	Genes	Conditions	VE^t	Mean Gene Variance
Yeast 2884x17	OPSM	14	496.1± 791.1	8.6± 4.4	0.189± 0.051	$1.39 \times 10^5 \pm 2.23 \times 10^5$
	ISA	0	-	-	-	-
	CC	100	34.0± 64.2	7.6± 3.2	0.151± 0.048	$3.20 \times 10^3 \pm 3.16 \times 10^3$
	Bimax	56	2297.6± 26.1	15.3± 0.5	0.207± 0.004	$1.42 \times 10^5 \pm 5.41 \times 10^3$
	Evo-Bexpa	100	44.0± 33.7	11.8± 3.9	0.051± 0.027	$9.81 \times 10^2 \pm 5.00 \times 10^2$
ET 7129x60	OPSM	12	1151.5± 1809.1	7.8± 4.1	0.155± 0.072	$2.51 \times 10^8 \pm 4.22 \times 10^8$
	ISA	20	377.0± 191.0	2.7± 0.8	0.145± 0.053	$5.98 \times 10^8 \pm 3.85 \times 10^8$
	xMotifs	1000	2134.6± 830.4	5.0± 0.0	0.135± 0.021	$9.61 \times 10^7 \pm 3.74 \times 10^7$
	CC	100	53.1± 100.4	11.9± 7.9	0.310± 0.082	$6.14 \times 10^4 \pm 2.06 \times 10^5$
	Evo-Bexpa	100	22.5± 6.8	50.8± 5.7	0.011± 0.003	$1.78 \times 10^7 \pm 1.59 \times 10^7$
Leukemia 7129x72	OPSM	12	924.3± 1633.3	7.8± 4.3	0.103± 0.047	$1.75 \times 10^8 \pm 2.89 \times 10^8$
	ISA	34	253.1± 172.1	3.1± 1.1	0.147± 0.049	$3.31 \times 10^8 \pm 2.17 \times 10^8$
	CC	100	53.5± 232.4	13.9± 8.6	0.265± 0.067	$4.32 \times 10^4 \pm 8.53 \times 10^4$
	Evo-Bexpa	100	18.4± 2.9	63.3± 5.9	0.008± 0.002	$4.46 \times 10^6 \pm 2.97 \times 10^6$
Steminal 26127x30	OPSM	27	1170.6± 3274.1	16.2± 8.8	0.399± 0.163	$1.73 \times 10^7 \pm 5.55 \times 10^7$
	ISA	0	-	-	-	-
	CC	100	179.1± 813.6	13.0± 3.1	0.219± 0.071	$4.84 \times 10^4 \pm 1.19 \times 10^5$
	Evo-Bexpa	100	33.8± 17.9	26.3± 2.4	0.009± 0.004	$4.80 \times 10^5 \pm 2.69 \times 10^5$

Table 7.5: Summary of experimental results for the microarrays in table 7.4.

Bimax generated 56 biclusters for Yeast dataset, all of them of a very big size, containing almost the totality of the elements, both genes and conditions. In fact, although we did not measure overlap for the algorithms in Bicat, it must be certainly high, since biclusters are made up of a mean of almost 2300 genes (out of 2884) and 15,16 or 17 experimental conditions (out of 17). Studying correlations in this kind of biclusters is almost as difficult as studying the whole dataset. It would even be easier to analyse the genes and/or samples not contained in the biclusters.

Murali and Kasif's xMotifs generates 1000 biclusters for the Embryonal Tumours dataset, all of them have 5 samples and a decreasing number of genes with the biclusters indexes. The first bicluster is made up of 4593 genes, more than the half of the whole dataset, while bicluster number 999 consist of 576 genes. We consider the number of biclusters to be cumbersome for any post analysis, even more if it needs to be carried out manually. Also, the number of genes per bicluster may again result too high for any specific

study.

ISA only found biclusters for Embryonal Tumours (20) and Leukemia (12) microarrays. In both cases they are obtained with a decreasing number of genes and conditions, being the second one a very low value, which we consider almost useless in biclustering analyses (2 or 3 samples per bicluster). The number of genes varies from 661 to 81 for the Embryonal database and from 707 to 83 for Leukemia. For both datasets the biclusters obtained by ISA have the greatest gene variance.

OPSM, together with CC and Evo-Bexpa produced results for the four datasets. OPSM biclusters are characterized for having the greatest deviation on the number of genes. In fact, OPSM bicluster's sizes vary from a bicluster containing a few genes and a great number of samples to the contrary: almost the whole set of genes and very few samples (2×17 to 2422×2 for Yeast, 2×16 to 5491×2 for ET, 2×17 to 5208×2 for Leukmia and 6×30 to 15332×2 in the Steminal case). From the biological point of view, only a small portion of the obtained biclusters are interesting: those in the intermediate situations.

CC algorithm allows the user to choose the number of biclusters to obtain, being 100 its default value. It is a sequential process in which random data is inserted into the matrix. For these reasons, first biclusters are in general greater than the following ones, being the smallest ones the last 10 biclusters. VE^t values are quite high, specially for Embryonal Tumours (the mean of VE^t is 0.3098) and Leukemia (VE^t mean is 0.2652) datasets, where biclusters sizes are not as big as to favour this range of values. Also, results produced by CC are rather flat, since their gene variance is in the majority of the cases the lowest of all the algorithms.

Default parameters for Evo-Bexpa in Table 7.2 have been adjusted to produce biclusters with a very low proportion of genes but a high proportion of samples, although there exists considerable diversity in the results, as shown by the deviation. Only for the Yeast dataset Evo-Bexpa obtains the biclusters with the lowest values of gene variance, while VE^t is always much lower, as preferred. In fact, VE^t values for the biclusters found by Evo-Bexpa is smaller than 0.1, for all datasets, whereas no other algorithm finds biclusters with such a low VE^t level. This is a very good achievement of our approach given the importance of VE^t as a quality measure for quantifying all kind of patterns in gene expression data. Furthermore, although VE^t values increase for bigger biclusters or those with lower levels of overlapping, it can be seen in figures 2 to 6 that they are never greater than the biclusters VE^t for the other approaches. The order in which biclusters are found with Evo-Bexpa is not relevant, although if the weights associated to the overlapping and size are too high Evo-Bexpa will produce big submatrices with no overlap, increasing thus VE^t considerably for the latest solutions.

The great advantage of Evo-Bexpa with regard to the other algorithms is its ability to adjust the result characteristics to user defined parameters. Next subsection presents biological validation for biclusters obtained by Evo-Bexpa, using the same parameter configuration introduced in section 7.3.1, which confirms the validity of our approach.

7.3.4 Biological Assessment

In this section we present the biological validation results of Evo-Bexpa biclusters using the biological information from *Gene Ontology* (GO). This validation has been carried out as explained in section 4.6, consisting in getting all the GO terms annotated to any of the genes in the bicluster $\mathcal{B}$ and then apply a statistical significance test to determine if each term appearance is relevant or not. A bicluster is said to be significantly enriched at any confident level if there exists at least one GO term for which genes in the bicluster are significantly annotated.

Among all the existing tools for the analysis of gene expression data using GO (Khatri & Drăghici (2005)) we have chosen Ontologizer by Bauer et al (2008) for assessing Evo-Bexpa biclusters due to its novelty (it has been recently updated) and its suitability for performing the validation of a great number of biclusters as a batch process. We have set Ontologizer up in order to carry out a Term-for-Term analysis using Fisher's exact test together with the Bonferroni multiple test correction, corresponding to the most common configuration for bicluster validation in the literature.

In order to check the influence of Evo-Bexpa configuration parameters on the biological validation of the obtained biclusters, we have represented in Figures 7.11 to 7.15 the number of significant biclusters (ordinates axis) for each of the experiments detailed in section 7.3.1 and for each dataset, where abscissa axis refer to the specific weight value, and the adjusted p-value has been set to 0.05.

The main conclusion we can come up to is that there is no a common behaviour embraced by the four different data sets and for each configuration parameter. For example, the number of significant biclusters for the Yeast dataset increases significantly whenever the number of genes (w_g) or conditions (w_c) are increased, as well as the overall size (w_s). Nevertheless, when the overlap gets more penalized (Figure 7.14), the number of significant biclusters for the Yeast dataset decreases. This is due to the fact that biclusters sizes are affected by the overlapping weight, in the reverse way (the more restrictive the overlapping amount is, the less elements the biclusters contains). Figure 7.15 shows that variance weight variations do not significantly affect the number of significant biclusters in the Yeast dataset.

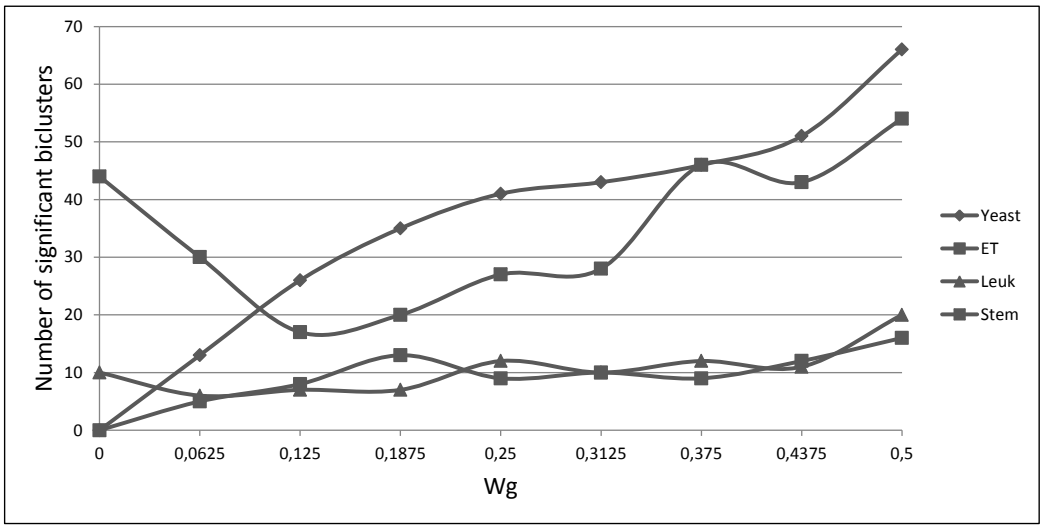


Figure 7.11: Number of significant biclusters for different w_g values.

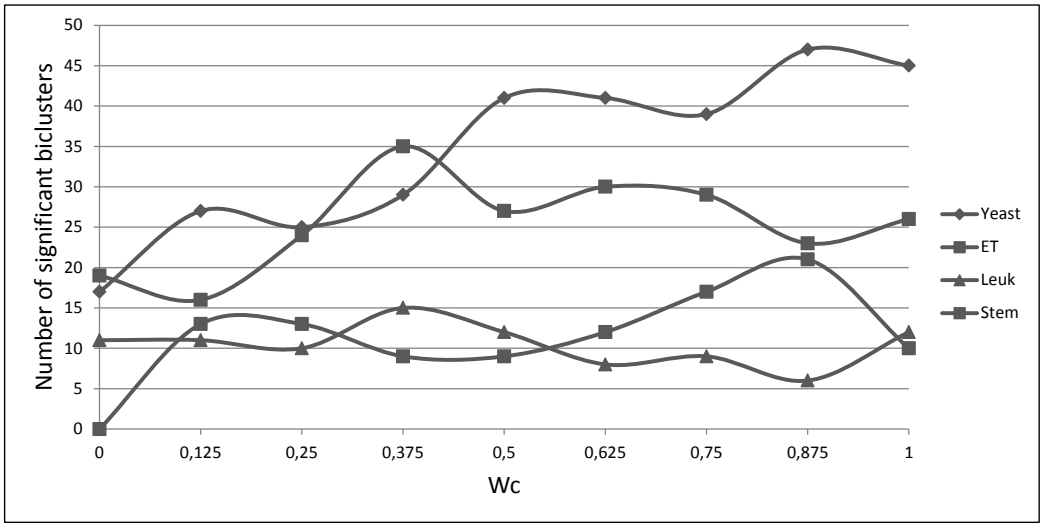


Figure 7.12: Number of significant biclusters for different w_c values.

Steminal dataset is the second one presenting more variations on the number of significant biclusters when modifying the parameters values. It is worth to note that Steminal is the only dataset for which the number of significant biclusters varies significantly from more to less when increasing the mean gene variance. This is related to the fact that when higher gene variances in biclusters are required, the size of the obtained biclusters decrease,

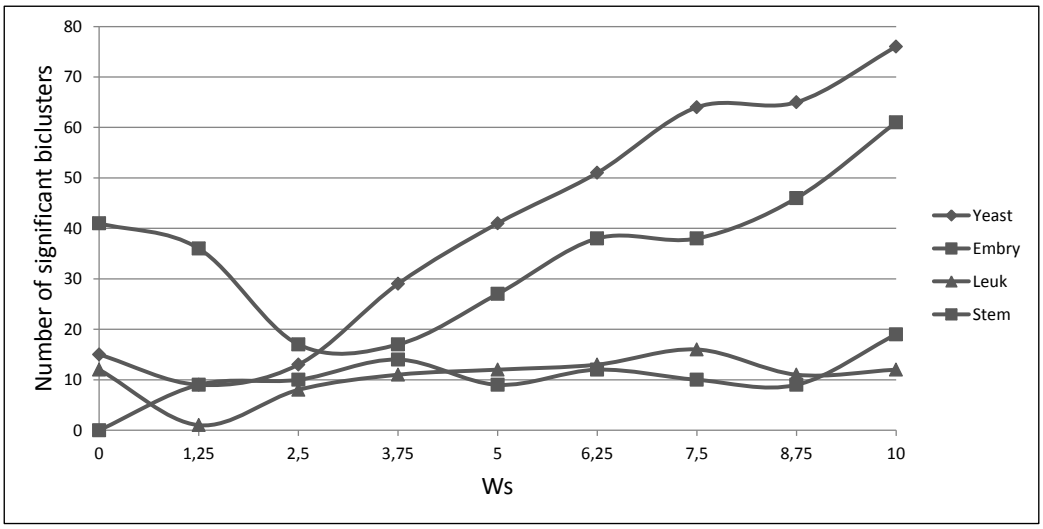


Figure 7.13: Number of significant biclusters for different w_s values.

as explained in section 7.3.1.

For Embryonal and Leukemia data sets the number of significant biclusters is quite lower than for the other two data sets, in all the cases. In fact, it rarely exceeds 20%. For both of them there no exist great variations when modifying the different parameter values. It is interesting to mark that no significant biclusters were found for the Embryonal datasets when w_g, w_c or w_s are set to zero (Figures 7.11 to 7.13).

In order to study the level of specificity (see section 4.6) of the terms to which Evo-Bexpa biclusters have been annotated we have carried out three different validations: taking the whole hierarchical graph into account, and limiting the validation with the levels 3 to 6 and 4 to 7, both inclusive.

Table 7.6 presents the validation results for Evo-Bexpa biclusters using the default configuration. For each type of validation the number of significant biclusters and the mean of their significant terms are given, for two different adjusted p-value values: 0.01 and 0.05. As it can be seen in Table 7.6, the number of significant biclusters slightly varies from the validation with the whole graph to the limited validations, meaning that the majority of biclusters obtained by Evo-Bexpa contain genes that are not frequently annotated to too general or too specific terms. For the Embryonal Tumours dataset the number of significant biclusters does not even decrease with the limited validations. The mean of significant terms to which genes in the biclusters of the previous column have been annotated is also smaller for the hierarchically limited validations. This was an expected result since those

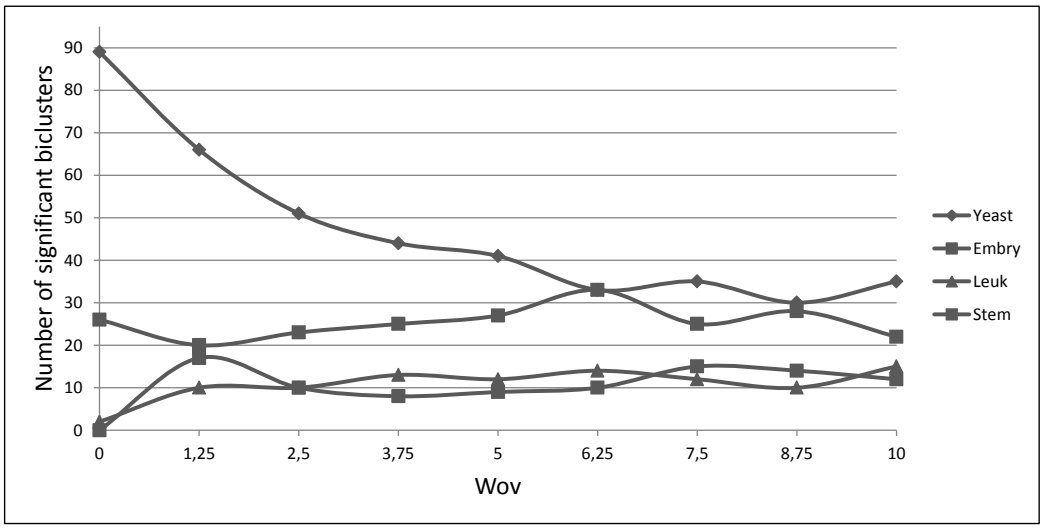


Figure 7.14: Number of significant biclusters for different w_{ov} values.

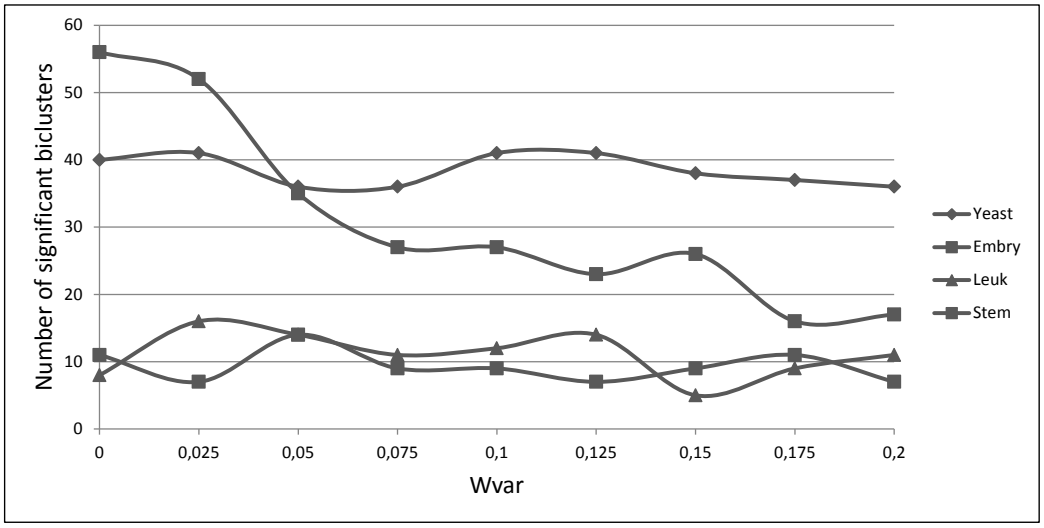


Figure 7.15: Number of significant biclusters for different w_{var} values.

terms which are not in the specified levels have not been taken into account in the study. Nevertheless, we consider the reduction on the number of significant biclusters and terms to be minimal, locating Evo-Bexpa biclusters in the central part of the GO graph.

In general, the validation carried out supports Evo-Bexpa effectiveness for biclustering microarray data. In fact, significant biclusters have been ob-

		All levels		Levels 3 to 6		Levels 4 to 7	
	p-value	#Bics	$\overline{\#Terms}$	#Bics	$\overline{\#Terms}$	#Bics	$\overline{\#Terms}$
Yeast	0.01	32	5.969	32	5.125	31	4.387
	0.05	41	6.878	40	5.625	40	5.225
ET	0.01	3	3.000	3	3.000	3	2.666
	0.05	9	3.778	9	3.444	9	3.333
Leukemia	0.01	4	1.000	3	1.000	3	1.000
	0.05	12	2.333	11	1.454	11	1.818
Steminal	0.01	18	4.056	16	2.750	13	2.231
	0.05	27	7.111	26	5.269	25	4.200

Table 7.6: Validation results with GO hierarchy level limitations.

tained for each dataset at 0.01 and 0.05 levels provided that configuration parameters are not set to zero. This fact also suggest the appropriateness of the different chosen objectives in this work. Although the number of significant biclusters may vary in a different way for different datasets when modifying the different configuration parameters, Evo-Bexpa significant biclusters correspond to significant terms in the central part of the GO graph. This means Evo-Bexpa succeeds at finding biclusters whose significant terms have an intermediate level of specificity.

7.4 Conclusions

In this chapter we have presented a new evolutionary algorithm for biclustering of gene expression data named Evo-Bexpa. Evo-Bexpa makes use of VE(^{t}) presented in chapter 6 as the bicluster quality metric for guiding the search. Furthermore, other objectives have been added to the individual evaluation, being all of them configurable with parameters. This means that the user has the possibility of choosing the importance of each objective, depending on the desired results. We have also presented the results of an extensive experimental set on both synthetic and real datasets, comparing our approach with others existing in the literature. For a more elaborated discussion on the proposals in this chapter we refer the reader to the general conclusions in chapter 8.

Chapter 8

Conclusions and Future Works

This chapter summarizes the main conclusions of the work carried out during the development of this PhD Thesis. Furthermore, we also describe future works that we think will be a very interesting continuation of the ones carried out.

8.1 Conclusions

Biclustering algorithms for microarray data aim at discovering functionally related gene sets under different subsets of experimental conditions. Due to the problem complexity and the characteristics of microarray datasets, heuristic searches are usually used instead of exhaustive algorithms. Also, the comparison among different techniques is still a challenge since the obtained results vary in relevant features such as the number of genes or conditions, which makes it difficult to carry out a fair comparison. Several quality measures for biclusters have been proposed together with different search heuristics. Nevertheless, none of the proposed quality measures is able to recognize a perfect shifting and scaling pattern in a bicluster, which describes the most general situation of gene correlation, according to the literature. The availability of such a measure would also be useful for comparing the results of different biclustering approaches. This way, we centred first our research upon the development of evaluation measures for biclusters based on the expression patterns concepts.

Moreover, evolutionary environments have been extensively used in biclustering, due to its appropriateness to the problem, where populations of potential solutions allow the exploration of a greater portion of the search space. Nevertheless, none of the existing approaches allow the user to choose the objectives involved in the search and their relevances in the search. In

this sense, we have also worked towards the development of a customizable evolutionary biclustering approach, where different objectives might be weighted by the user depending on the desired results preferences. This objectives include volume, gene variance or overlapping level among biclusters. Furthermore, it would also be very easy for the user to incorporate new objectives into the search.

Our contributions of these two areas of research are summarized below.

8.1.1 Bicluster Evaluation Measures

Our evaluation measures are all based on a previous standardization of the bicluster, where the main idea is to shift the expression levels of all the genes in the bicluster to a common range. The standardization mechanism will also be responsible for soften every gene or condition behaviour, since the most important aspect is to characterize its tendency rather than its particular numerical values. Furthermore, using an evaluation measure with this characteristic would not be necessary to carry out a normalization process of the data matrix before the search. Our two main contributions in this field are *Virtual Error* (VE) and *Transposed Virtual Error* (VE^t):

Virtual Error (VE)

The basic idea behind VE is to measure how genes follow the general tendency within the bicluster. In order to capture this general tendency, a *virtual gene* is first computed as a gene consisting of the arithmetic mean of all the genes in the bicluster. Afterwards, both the bicluster and the virtual gene are standardized, and VE measures the differences between every standardized gene and the standardized virtual one.

Using this simple strategy, VE has been proven to be capable of dealing with both shifting and scaling patterns, though not simultaneously. Moreover, unlike other metrics in the literature, VE does not need the use of a threshold for rejecting biclusters, so that an algorithm using VE has less constraints through the search.

In order to assess the validity of VE, we conducted experiments on nine microarray datasets. For this, we have incorporated VE into a multi-objective evolutionary strategy, where the other objectives were the gene variance and the volume of the bicluster. We have compared the results of the previous algorithm with two other settings of the algorithm. These last two settings use the most popular measure for biclusters (MSR) instead of VE. Specifically, the first setting uses the recommended threshold for rejecting biclusters, while in the latter version this thresholds was removed. This last version

of the algorithm was used in order to test whether the use of a too small threshold would prevent the algorithm from finding interesting biclusters.

From these experiments we concluded that VE yields the algorithm obtaining good results on all the datasets. In general, biclusters found by VE are characterized by a greater volume and gene variance. An interesting result is that when using MSR for guiding the search together with the recommended threshold for each data set, the average VE obtained is lower than when using VE. This is easily explained by the fact that low values of MSR correspond to low values of VE. We have also confirmed that the use of a threshold may prevent an algorithm from finding good results, when its value is too small.

We have also conducted a biological validation of the results obtained on three datasets. From this validation, it emerges that VE yields the discovery of biclusters whose genes have a stronger functional coherence than when using MSR. Moreover, when the algorithm used VE, it discovers more significant biclusters, according to the adjusted p-value.

In general, we can conclude that VE is an effective measure for assessing the quality of biclusters. In particular, VE is effective at recognizing biclusters containing both shifting and scaling patterns as quality biclusters. The same conclusions does not hold for MSR, which is negatively influenced by the presence of scaling patterns. It follows that VE can be used effectively within any heuristics for finding biclusters in gene expression data.

Transposed Virtual Error ($\mathbf{VE}^t$)

VE^t represents an enhanced version of VE, allowing finding biclusters with both shifting and scaling patterns simultaneously in gene expression data. Since no previous evaluation measure for biclusters is able of identifying this kind of pattern, VE^t constitutes an important contribution to the topic.

VE^t is computed similarly to VE but considering the transposed bicluster. The idea here is to create a *virtual condition* instead of a virtual gene. Afterwards, the differences between the standardized values for every condition and the standardized virtual condition are measured in the same way as in VE.

Analytical proofs demonstrate the capability of VE^t for detecting any kind of perfect pattern in gene expression data. Furthermore, we have also proved that VE^t presents a linear relationship with the amount of error in a bicluster, improving thus the behaviour of VE when a bicluster approximates but does not follow a perfect pattern.

VE^t has also been incorporated into a evolutionary search strategy and tested against five microarray datasets. Since the algorithm used in this

experimentation constitutes a different contribution in this PhD Thesis, experimental results will be concluded later in this section.

8.1.2 Evolutionary Biclustering of Gene Expression Data

Although searching for biclusters in microarrays usually involves using several objectives, a multi-objective approach based on the individuals dominances is not necessarily the best option. Using dominances, individual fitness is not conserved through generations, and individuals with the same genotype in different populations may not have the same fitness. Furthermore, a multi-objective approach does not allow establishing any kind of prevalence among the different objectives. However, depending on the microarray under study and its applications, it may be interesting to prioritize some bicluster features over others.

In order to overcome these issues, we have developed a new evolutionary algorithm for biclustering of gene expression data named Evo-Bexpa. There exist two main advantages over other existing approaches: the use of an evaluation measure able to detect shifting and scaling patterns (VE^t), and the possibility of specifying user preferences on some characteristics of the results, specifically the number of genes, number of conditions, overlapping amount and gene mean variances. This way, if any previous information related to the microarray under study is available, the search can be guided towards the preferred types of biclusters. Furthermore, other objectives can also be easily incorporated into the search, as well as any objective may be ignored by setting its weight to zero. Default values for the configuration parameters are given in order to provide the user with quality results. Moreover, an experimental study has been performed on four real datasets in order to study the parameters sensibility and their influence over the different features. This study concludes with an useful guide on how to customize the algorithm depending on the user preferences.

Experimental results on both synthetic and real datasets confirm the validity of our approach, where the results have been compared to those obtained by five well-known biclustering algorithms. Evo-Bexpa has been proven to outperform ISA, xMotifs, OPSM, BIMAX and CC in synthetic experiments, where match scores indexes have been used for comparing the obtained results with the solution. Regarding the experiments on real datasets, Evo-Bexpa results have been biologically validated using different levels in Gene Ontology hierarchy. This validation shows that significant biclusters obtained by Evo-Bexpa correspond to neither too general or specific

GO terms.

8.2 Future Works

As for future work, we are planning to design an improved version of Evo-Bexpa which incorporates a local search for enhancing several solutions with biological information. Furthermore, we are also planning to develop a more robust system for the biclusters validation and interpretation, based on the integration of different information sources. Both ideas are described in the following.

8.2.1 Memetic Evo-Bexpa

Biological information from diverse databases, such as Gene Ontology (GO) or KEGG (Kyoto Encyclopaedia of Genes and Genomes), is currently being used for the validation of the biclusters obtained after applying any search heuristic. Nevertheless, we consider it would be very interesting to use this kind of information to guide the search towards meaningful solutions. The idea would be to add a new objective to the bicluster evaluation based on its biological significance. Using Evo-Bexpa it seems relatively simple to carry out this task, since it has been specially designed to configure new objectives. Nevertheless, several issues needs to be resolved beforehand, such as connections to databases and time performance.

In order to perform biological validation of biclusters, not all available resources and databases on the Internet have the same connection procedure to access to the information. Some of them are only accessible via web browsers, while some others allow connection via web services or even the databases files are public for download. Furthermore, it is not frequent that this databases themselves carry out the statistical tasks needed for the validation of a group of genes. Nevertheless, there exists plenty of tools developed by external cooperators that perform different kinds of validations. This way, the first task for improving Evo-Bexpa objectives using biological information would be to study the different databases available as well as their accessing possibilities and their corresponding existing tools.

Another important issue is related to the time performance for individuals evaluation. Typically, individual evaluation is critical in evolutionary algorithms, since it is performed many times for each generation. This way, accessing to external data bases and carrying out the statistical operations involved in the bicluster validation for each individual would be very expensive computationally. For this reason, we think it would be a good idea to

perform this validation-based evaluation only to a small number of individuals, either in each generation or only in one of several populations. These approximations would be very appropriate for an implementation based on memetic algorithms (see section 5.4.2), also known as an hybridization of an EA with local search.

8.2.2 Integrated Biological Validation and Interpretation of Biclusters

Different public biological databases for the verification and interpretation of microarray analyses have been presented in section 3.2.4. We also discussed the inconvenience of having various sources of information, varying from data formats to the specific contents. Integrating all these pieces of information would significantly enrich both the validation and interpretation of experimental results obtained from data mining techniques in the biological context. In this sense, we plan to improve our validation mechanism, where biclusters are validated using information retrieved from *Gene Ontology* (GO) solely.

First step in this work would consist in making an in-depth study on the many biological databases existing on the Internet, pointing out their more interesting characteristics for our purpose. This study would also include those existing integrated databases which join data from several sources either in a navigational, warehousing or federations way. Furthermore, special attention must be paid to accessing protocols and related software for each alternative. With all this information gathered up, it would be easier to come up to a decision on the best strategy for a complete validation and interpretation of gene expression biclusters.

Bibliography

Agrawal R, Gehrke J, Gunopulos D, Raghavan P (2005) Automatic subspace clustering of high dimensional data. Data Mining and Knowledge Discovery 11(1):5–33

Aguilar-Ruiz JS (2005) Shifting and scaling patterns from gene expression data. Bioinformatics 21:3840–3845

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

Alizadeh AA, B M, Davis RE, et al (2000) Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. Nature 403:503–511

Alon U, Barkai N, Notterman DA, Gishdagger K, Ybarradagger S, Mackdagger D, Levine AJ (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proceedings of the National Academy of Sciences of the United States of America 96(12):6745–6750

Alterovitz G, Xiang M, Mohan M, Ramoni M (2007) Go pad: the gene ontology partition database. Nucleic Acids Research 35(Database-Issue):322–327

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

Angiulli F, Cesario E, Pizzuti C (2008) Random walk biclustering for microarray data. Information Sciences 178(6):1479–1497

Ashburner M, Ball C, Blake J, Botstein D, Butler H, Cherry J, Davis A, Dolinski K, Dwight S, Eppig J, et al (2000) Gene ontology: tool for the unification of biology. The Gene Ontology. Nature Genetics 25:25–29

Asyali MH, Colak D, Demirkaya O, Inan MS (2006) Gene expression profile classification: A review. Current Bioinformatics 1:55–73

Ayadi W, Elloumi M, Hao J (2009) A biclustering algorithm based on a bicluster enumeration tree: application to dna microarray data. BioData mining 2(1):9

Ayadi W, Elloumi M, Hao J (2012) Pattern-driven neighborhood search for biclustering of microarray data. BMC bioinformatics 13(Suppl 7):S11

Baldi P, Hatfield GW (2002) DNA Microarrays and Gene Expression From Experiments to Data Analysis and Modeling. Cambridge University Press

Baldi P, Long AD (2001) microarray expression data : regularized t-test. Bioinformatics 17(6):509–519

Barkow S, Bleuler S, Prelic A, Zimmermann P, Zitzler E (2006) Bicat: a biclustering analysis toolbox. Bioinformatics 22(10):1282–1283

Barlow H (1989) Unsupervised learning. Neural Computing 1(3):295–311

Barrett T, Edgar R (2006) Gene expression omnibus: Microarray data storage, submission, retrieval, and analysis. Methods in enzymology 411:352–369

Bauer S, Grossmann S, Vingron M, Robinson P (2008) Ontologizer 2.0—a multifunctional tool for go term enrichment analysis and data exploration. Bioinformatics 24(14):1650–1651

Bauer S, Gagneur J, Robinson P (2010) Going bayesian: model-based gene set analysis of genome-scale data. Nucleic acids research 38(11):3523–3532

Bellazzi R, Masseroli M, Murphy S, Shabo A, Romano P (2012) Clinical bioinformatics: challenges and opportunities. BMC Bioinformatics 13(Suppl 14):S1

Ben-Dor A, Bruhn L, Friedman N, Nachman I, Schummer M, Yakhini Z (2000) Tissue classification with gene expression profiles. Journal of Computational Biology 7(3-4):559–583

Ben-Dor A, Chor B, Karp RM, Yakhini Z (2003) Discovering local structure in gene expression data: The order-preserving submatrix problem. Journal of Computational Biology 10(3-4):373–384

Ben-Hur A, Elisseeff A, Guyon I (2001) A stability based method for discovering structure in clustered data 7:6–17

Bergmann S, Ihmels J, Barkai N (2003) Iterative signature algorithm for the analysis of large-scale gene expression data. Physical review E 67(3):031,902

Beyer H, Schwefel H (2002) Evolution strategies–a comprehensive introduction. Natural computing 1(1):3–52

Bhardwaj N, Lu H (2005) Correlation between gene expression profiles and protein–protein interactions within and across genomes. Bioinformatics 21(11):2730–2738

Bishop CM (2007) Pattern Recognition and Machine Learning (Information Science and Statistics), 1st edn. Springer

Bland J, Altman D (1995) Calculating correlation coefficients with repeated observations: Part 2—correlation between subjects. British Medical Journal 310(6980):633

Bleuler S, Prelić A, Zitzler E (2004) An EA framework for biclustering of gene expression data. In: Congress on Evolutionary Computation (CEC-2004), IEEE, pp 166–173

Blickle T, Thiele L (1996) A comparison of selection schemes used in evolutionary algorithms. Evolutionary Computation 4(4):361–394

Blouin C, Perry S, Lavell A, Susko E, Roger AJ (2009) Reproducing the manual annotation of multiple sequence alignments using a svm classifier. Bioinformatics 25(23):3093–3098

Bolstad BM, Irizarry RA, Astrand M, Speed TP (2003) A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. Bioinformatics 19(2):185–193

Bosman P, de Jong E (2006) Combining gradient techniques for numerical multi-objective evolutionary optimization. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM, pp 627–634

Boyer L, Plath K, Zeitlinger J, Brambrink T, Medeiros L, Lee T, Levine S, Wernig M, Tajonar A, Ray M, et al (2006) Polycomb complexes repress developmental regulators in murine embryonic stem cells. Nature 441(7091):349–353

Bozdağ D, Parvin J, Catalyurek U (2009) A biclustering method to discover co-regulated genes using diverse gene expression datasets. Bioinformatics and Computational Biology pp 151–163

Bozdağ D, Kumar AS, Catalyurek UV (2010) Comparative analysis of biclustering algorithms. In: Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology, ACM, New York, NY, USA, BCB '10, pp 265–274

Brazma A, Hingamp P, Quackenbush J, Sherlock G, Spellman P, Stoeckert C, Aach J, Ansorge W, Ball C, Causton H, et al (2001) Minimum information about a microarray experiment (miame)-toward standards for microarray data. Nature genetics 29(4):365–372

Brazma A, Parkinson H, Sarkans U, Shojatalab M, Vilo J, Abeygunawardena N, Holloway E, Kapushesky M, Kemmeren P, Lara G, et al (2003) Arrayexpress—a public repository for microarray gene expression data at the ebi. Nucleic Acids Research 31(1):68–71

Breitling R, Herzyk P (2005) Rank-based methods as a non-parametric alternative of the t-statistic for the analysis of biological microarray data. J Bioinformatics and Computational Biology 3(5):1171–1190

Breitling R, Armengaud P, Amtmann A, Herzyk P (2004) Rank products: a simple, yet powerful, new method to detect differentially regulated genes in replicated microarray experiments. FEBS Letters 573(1-3):83–92

Bryan K, Cunningham P, Bolshakova N (2006) Application of simulated annealing to the biclustering of gene expression data. Information Technology in Biomedicine, IEEE Transactions on 10(3):519–525

Bucca G, Laing E, Mersinias V, Allenby N, Hurd D, Holdstock J, Brenner V, Harrison M, Smith CP (2009) Development and application of versatile high density microarrays for genome-wide analysis of streptomyces coelicolor: characterization of the hspr regulon. Genome Biol 10(1):R5

Buness A, Ruschhaupt M, Kuner R, Tresch A (2009) Classification across gene expression microarray studies. BMC Bioinformatics 10:453

Busygin S, Prokopyev OA, Pardalos PM (2008) Biclustering in data mining. Computers & OR 35(9):2964–2987

Cano C, Adarve L, López J, Blanco A (2007) Possibilistic approach for biclustering microarray data. Computers in Biology and Medicine 37(10):1426–1436

Cantú-Paz E (2000) Selection intensity in genetic algorithms with generation gaps. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 911–918

Carlos A, Coello C, Christiansen A (1998) Two new ga-based methods for multiobjective optimization. Civil Engineering Systems 15(3):207–243

Carmona-Saez P, Pascual-Marqui R, Tirado F, Carazo J, Pascual-Montano A (2006) Biclustering of gene expression data by non-smooth non-negative matrix factorization. BMC bioinformatics 7(1):78

de Castro L, Von Zuben F (2001) ainet: an artificial immune network for data analysis. Data mining: a heuristic approach 1:231–259

Chain P, Kurtz S, Ohlebusch E, Slezak T (2003) An applications-focused review of comparative genomics tools: capabilities, limitations and future challenges. Briefings in bioinformatics 4(2):105–123

Charnes A, Cooper W (1957) Management models and industrial applications of linear programming. Management Science 4(1):38–91

Chen L, Xuan J, Riggins RB, Wang Y, Hoffman EP, Clarke R (2010) Multilevel support vector regression analysis to identify condition-specific regulatory networks. Bioinformatics 26(11):1416–1422

Chen Y, Liu C (1994) Multiobjective var planning using the goal-attainment method. In: Generation, Transmission and Distribution, IEE Proceedings-, IET, vol 141, pp 227–232

Chen Y, Dougherty ER, Bittner ML (1999) Ratio-based decisions and the quantitative analysis of cDNA microarray images. Journal of Biomedical Optics 2(4)

Cheng Y, Church GM (2000) Biclustering of expression data. In: Proceedings of the 8th International Conference on Intellingent Systemns for Molecular Biology, La Jolla, CA, pp 93–103

Chipperfield A, Fleming P (1995) Gas turbine engine controller design using multiobjective genetic algorithms. In: Genetic Algorithms in Engineering Systems: Innovations and Applications, 1995. GALESIA. First International Conference on (Conf. Publ. No. 414), IET, pp 214–219

Cho H (2010) Data transformation for sum squared residue. In: Proceedings of the 14th Pacific-Asia conference on Advances in Knowledge Discovery

and Data Mining - Volume Part I, Springer-Verlag, Berlin, Heidelberg, PAKDD'10, pp 48–55

Cho R, Campbell M, Winzeler E, Steinmetz L, Conway A, Wodicka L, Wolfsberg T, Gabrielian A, Landsman D, Lockhart D, Davis R (1998) A genome-wide transcriptional analysis of the mitotic cell cycle. Molecular Cell 2:65–73

Choe SE, Boutros M, Michelson AM, Church GM, Halfon MS (2005) Preferred analysis methods for Affymetrix GeneChips revealed by a wholly defined control dataset. Genome biology 6(2):R16

Coelho GP, de Franca FO, Zuben FJV (2009) Multi-objective biclustering: When non-dominated solutions are not enough. Journal of Mathematical Modelling and Algorithms 8(2):175–202

Coello C (2006) Evolutionary multi-objective optimization: a historical view of the field. Computational Intelligence Magazine, IEEE 1(1):28 – 36

Coello C, Christiansen A (2000) Multiobjective optimization of trusses using genetic algorithms. Computers & Structures 75(6):647–660

Coello C, Lamont G, Van Veldhuizen D (2006) Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation), vol 5. Springer-Verlag New YorK, Secaucus, NJ, USA

Cohen J (2004) Bioinformatics - an introduction for computer scientists. ACM Computing Surveys 36:122–158

Conesa A, Nueda MJ, Ferrer A, Talon M (2006) masigpro: a method to identify significantly differential expression profiles in time-course microarray experiments. Bioinformatics 22(9):1096–102

Crow J, Kimura M (1979) Efficiency of truncation selection. Proceedings of the National Academy of Sciences 76(1):396–399

Curtis R, Oresic M, Vidal-Puig A (2005) Pathways to the analysis of microarray data. TRENDS in Biotechnology 23(8):429–435

Darwin C (1888) The origin of species by means of natural selection. TY Crowell

Dasgupta D, McGregor D (1992) sga: A structured genetic algorithm. University of Strathclyde

David J Reiss NSB, Bonneau R (2006) Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks. BMC Bioinformatics 7:280

De Jong K, Sarma J (1993) Generation gaps revisited. Foundations of genetic algorithms 2:19–28

Deb K, Agrawal S, Pratap A, Meyarivan T (2000) A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. Lecture notes in computer science 1917:849–858

Devaux Y, Azuaje F, Vausort M, Yvorra C, Wagner D (2010) Integrated protein network and microarray analysis to identify potential biomarkers after myocardial infarction. Functional & integrative genomics 10(3):329–337

Dharan S, Nair AS (2009) Biclustering of gene expression data using reactive greedy randomized adaptive search procedure. BMC Bioinformatics 10(S-1)

Díaz-Díaz N, Aguilar-Ruiz J (2011) Go-based functional dissimilarity of gene sets. BMC bioinformatics 12(1):360

DiMaggio P, McAllister S, Floudas C, Feng X, Rabinowitz J, Rabitz H (2008) Biclustering via optimal re-ordering of data matrices in systems biology: rigorous methods and comparative studies. BMC Bioinformatics 9(1):458

Divina F, Aguilar-Ruiz JS (2006) Biclustering of expression data with evolutionary computation. IEEE Transactions on Knowledge & Data Engineering 18(5):590–602

Divina F, Pontes B, Giráldez R, Aguilar-Ruiz JS (2012) An effective measure for assessing the quality of biclusters. Computers in Biology and Medicine 42(2):245–256

Dopazo J, Carazo JM (1997) Phylogenetic reconstruction using an unsupervised growing neural network that adopts the topology of a phylogenetic tree. J Mol Evol 44:226–233

Draghici S (2002) Statistical intelligence: effective analysis of high-density microarray data. Drug Discov Today 7(11 Suppl)

Duval B, Hao JK (2010) Advances in metaheuristics for gene selection and classification of microarray data. Briefings in Bioinformatics 11(1):127–141

Eisen M, Spellman P, Brown P, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Sciences 95(25):14,863–14,868

Eren K, Deveci M, Küçüktunç O, Çatalyürek Ü (2012) A comparative analysis of biclustering algorithms for gene expression data. Briefings in Bioinformatics

Eshelman L (1991) The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. Foundations of genetic algorithms 1(265-283):108

Ester M, peter Kriegel H, S J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. AAAI Press, pp 226–231

Euskirchen GM, Rozowsky JS, Wei CL, Lee WH, Zhang ZD, Hartman S, Emanuelsson O, Stolc V, Weissman S, Gerstein MB, Ruan Y, Snyder M (2007) Mapping of transcription factor binding regions in mammalian cells by ChIP: Comparison of array- and sequencing-based technologies. Genome Res 17(6):898–909

Evans MR (2012) Philosophical Transactions of the Royal Society B: Biological Sciences 367(1586):181–190

Falcon S, Gentleman R (2007) Using gostats to test gene lists for go term association. Bioinformatics 23(2):257–258

Fayyad U, Piatetsky-shapiro G, Smyth P (1996) From data mining to knowledge discovery in databases. AI Magazine 17:37–54

Fielding A (2007) Cluster and classification techniques for the biosciences. Cambridge University Press

Firpi HA, Ucar D, Tan K (2010) Discover regulatory dna elements using chromatin signatures and artificial neural network. Bioinformatics 26(13):1579–1586

Fleischmann RD, Adams MD, White O, Clayton RA, Kirkness EF, Kerlavage AR, Bult CJ, Tomb JF, Dougherty BA, Merrick JM (1995) Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. Science (New York, NY) 269(5223):496–512

Floreano D, Mattiussi C (2008) Bio-Inspired Artificial Intelligence. Massachusetts Institute of Technology

Fogel L, Owens A, Walsh M (1966) Artificial intelligence through simulated evolution

Fonseca C, Fleming P, et al (1993) Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: Proceedings of the fifth international conference on genetic algorithms, Citeseer, vol 1, p 416

Friedberg R (1958) A learning machine: Part i. IBM Journal of Research and Development 2(1):2–13

Gallo C, Carballido J, Ponzoni I (2009) Bihea: A hybrid evolutionary approach for microarray biclustering. Advances in Bioinformatics and Computational Biology pp 36–47

Gan X, Liew A, Yan H (2008) Discovering biclusters in gene expression data based on high-dimensional linear geometries. BMC bioinformatics 9(1):209

Gasch AP, Eisen MB (2002) Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. Genome Biology 3(11):research0059.10,059.22

Getz G, Levine E, Domany E (2000) Coupled two-way clustering analysis of gene microarray data. Proceedings of the National Academy of Sciences 97(22):12,079

Ghosh A, Tsutsui S (2003) Advances in evolutionary computing: theory and applications. Springer

Giard J, Ambroise J, Gala JL, Macq B (2009) Regression applied to protein binding site prediction and comparison with classification. BMC Bioinformatics 10:276

Giraldez R, Divina F, Pontes B, Aguilar-Ruiz J (2007) Evolutionary search of biclusters by minimal intrafluctuation. In: IEEE International Fuzzy Systems Conference, 2007., IEEE, pp 1–6

Goldberg D, Korb B, Deb K (1989) Messy genetic algorithms: Motivation, analysis, and first results. Complex systems 3(5):493–530

Goldberg DE (1989) Genetic Algorithms in Search, Optimization & Machine Learning. Addison Wesley

Golub T, Slonim D, Tamayo P, Huard C, Gaasenbeek M, Mesirov J, Coller H, Loh M, Downing J, Caligiuri M, Bloomfield C, Lander E (1999) Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286:531–37

Grossmann S, Bauer S, Robinson P, Vingron M (2007) Improved detection of overrepresentation of gene-ontology annotations with parent–child analysis. Bioinformatics 23(22):3024–3031

Gu J, Bourne PE (2009) Structural Bioinformatics. Wiley-Blackwell

Gu J, Liu J (2008) Bayesian biclustering of gene expression data. BMC genomics 9(Suppl 1):S4

Hajela P, Lin C (1992) Genetic search strategies in multicriterion optimal design. Structural and Multidisciplinary Optimization 4(2):99–107

Handl J, Knowles J, Kell DB (2005) Computational cluster validation in post-genomic data analysis. Bioinformatics 21(15):3201–3212

Harder T, Borg M, Boomsma W, Røgen P, Hamelryck T (2012) Fast large-scale clustering of protein structures using gauss integrals. Bioinformatics 28(4):510–515

Hardiman G (2008) Applications of microarrays and biochips in pharmacogenomics. Methods in Molecular Biology 448:21

Hartemink AJ, Gifford DK, Jaakkola TS, Young RA (2002) Combining location and expression data for principled discovery of genetic regulatory network models. Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing pp 437–449

Hartigan J (1972) Direct clustering of a data matrix. Journal of the American Statistical Association 67(337):123–129

Hartigan J (1975) Clustering Algorithms. John Wiley and Sons, New York

Hartuv E, Schmitt A, Lange J, Meier-Ewert S, Lehrach H, Shamir R (2000) An algorithm for clustering cdna fingerprints. Genomics 66:249–256

Hassanzadeh O (2009) Automated protein structure classification: A survey. CoRR abs/0907.1990

Haubold B, Wiehe T (2004) Comparative genomics: methods and applications

Hautaniemi S, Kharait S, Iwabu A, Wells A, Lauffenburger DA (2005) Modeling of signal–response cascades using decision tree analysis. Bioinformatics 21(9):2027–2035

Hawkins T, Kihara D (2007) Function prediction of uncharacterized proteins. Journal of bioinformatics and computational biology 5(1):1–30

Hazelhurst S, Lipták Z (2011) Kaboom! a new suffix array based algorithm for clustering expression data. Bioinformatics 27(24):3348–3355

Holland J (1975) Adaptation in natural and artificial systems. 53, University of Michigan press

Horn J, Nafpliotis N, Goldberg D (1994) A niched pareto genetic algorithm for multiobjective optimization. In: Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on, Ieee, pp 82–87

Huang LT, Gromiha MM, Ho SY (2007) iptree-stab: interpretable decision tree based method for predicting protein stability changes upon mutations. Bioinformatics 23(10):1292–1293

Huang Q, Tao D, Li X, Liew AWC (2012) Parallelized evolutionary learning for detection of biclusters in gene expression data. IEEE/ACM Transactions on Computational Biology and Bioinformatics 9:560–570

Ideker T, Galitski T, Hood L (2001) A new approach to decoding life: systems biology. Annu Rev Genomics Hum Genet 2:343–372

Ihmels J, Bergmann S, Barkai N (2004) Defining transcription modules using large-scale gene expression data. Bioinformatics 20:1993–2003

Iida K, Nishimura I (2002) Gene expression profiling by dna microarray technology. Critical Reviews in Oral Biology & Medicine 13(1):35–50

Ishibuchi H, Murata T (1996) Multi-objective genetic local search algorithm. In: Proceedings of IEEE International Conference on Evolutionary Computation, 1996., IEEE, pp 119–124

Jacq B (2001) Protein function from the perspective of molecular interactions and genetic networks. Briefings in bioinformatics 2(1):38–50

Jancura P, Mavridou E, Pontes B, Marchiori E (2011) Describing the orthology signal in a ppi network at a functional, complex level. Bioinformatics Research and Applications pp 209–226

Jensen LJ, Bateman A (2011) The rise and fall of supervised machine learning techniques. Bioinformatics 27(24):3331–3332

Jiang D, Tang C, Zhang A (2004) Cluster analysis for gene expression data: A survey. IEEE Trans Knowl Data Eng 16(11):1370–1386

Johnson AD (2009) Single-nucleotide polymorphism bioinformatics: a comprehensive review of resources. Circulation Cardiovascular genetics 2(5):530–536

Kerr MK, Martin M, Churchill GA (2000) Analysis of variance for gene expression microarray data. Journal of computational biology : a journal of computational molecular cell biology 7(6):819–837

Khatri P, Drăghici S (2005) Ontological analysis of gene expression data: current tools, limitations, and open problems. Bioinformatics 21(18):3587–3595

Kirkpatrick S, Gelatt Jr C, Vecchi M (1983) Optimization by simulated annealing. science 220(4598):671–680

Kluger Y, Basri R, Chang J, Gerstein M (2003) Spectral bicluster of microarray data: Coclustering genes and conditions. Genome Research 13:703–716

Knowles J, Corne D (2005) Memetic algorithms for multiobjective optimization: issues, methods and prospects. Recent advances in memetic algorithms pp 313–352

Koonin EV, Wolf YI (2006) Evolutionary systems biology: links between gene evolution and function. Current Opinion in Biotechnology 17(5):481–487

Kota P, Ding F, Ramachandran S, Dokholyan NV (2011) Gaia: automated quality assessment of protein structure models. Bioinformatics 27(16):2209–2215

Koza JR (1992) Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems), 1st edn. A Bradford Book

Kozhenkov S, Baitaluk M (2012) Mining and integration of pathway diagrams from imaging data. Bioinformatics 28(5):739–742

Kuninger D, Kuzmickas R, Peng B, Pintar J, Rotwein P (2004) Gene discovery by microarray: identification of novel genes induced during growth factor-mediated muscle cell survival and differentiation. Genomics 84(5):876–889

Kuosmanen P, Astola J, Agaian S (1994) On rank selection probabilities. Signal Processing, IEEE Transactions on 42(11):3255–3258

Lazzeroni L, Owen A (2002) Plaid models for gene expression data. Statistica Sinica 12(1):61–86

Le Novère N, Hucka M, Mi H, Moodie S, Schreiber F, Sorokin A, Demir E, Wegner K, Aladjem MI, Wimalaratne SM, Bergman FT, Gauges R, Ghazal P, Kawaji H, Li L, Matsuoka Y, Villéger A, Boyd SE, Calzone L, Courtot M, Dogrusoz U, Freeman TC, Funahashi A, Ghosh S, Jouraku A, Kim S, Kolpakov F, Luna A, Sahle S, Schmidt E, Watterson S, Wu G, Goryanin I, Kell DB, Sander C, Sauro H, Snoep JL, Kohn K, Kitano H (2009) The Systems Biology Graphical Notation. Nature biotechnology 27(8):735–741

Le Roch KG, Zhou Y, Blair PL, Grainger M, Moch KJ, Haynes DJ, La D, Holder AA, Batalov S, Carucci DJ, Winzeler EA (2003) Discovery of gene function by expression profiling of the malaria parasite life cycle. Science 301(5639):1503–1508

Lesk A (2008) Introduction to Bioinformatics. Oxford

Lettieri T (2006) Recent applications of dna microarray technology to toxicology and ecotoxicology. Environmental health perspectives 114(1):4

Li G, Ma Q, Tang H, Paterson AH, Xu1 Y (2009) Qubic: a qualitative biclustering algorithm for analyses of gene expression data. Nucleic acids research 37(15):101–101

Linder R, Dew D, Sudhoff H, Theegarten D, Remberger K, Pöppl SJ, II MW (2004) The 'subsequent artificial neural network' (sann) approach might bring more classificatory power to ann-based dna microarray analyses. Bioinformatics 20(18):3544–3552

Liu J, Li Z, Hu X, Chen Y (2009) Biclustering of microarray data with mospo based on crowding distance. BMC bioinformatics 10 Suppl 4(Suppl 4):S9

Liu X, Wang L (2007) Computing the maximum similarity bi-clusters of gene expression data. Bioinformatics 23:50–56

Lönnstedt I, Speed T (2001) Replicated microarray data. Statistica Sinica 12:31–46

Loewe L (2009) A framework for evolutionary systems biology. BMC Systems Biology 3(1):27

Lourenço VM, Pires AM, Kirst M (2011) Robust linear regression methods in association studies. Bioinformatics 27(6):815–821

Lu Y, Zhou Y, Qu W, Deng M, Zhang C (2011) A lasso regression model for the construction of microrna-target regulatory networks. Bioinformatics 27(17):2406–2413

Madeira SC, Oliveira AL (2004) Biclustering algorithms for biological data analysis: A survey. IEEE Transactions on Computational Biology and Bioinformatics 1:24–25

Magi A, Benelli M, Gozzini A, Girolami F, Torricelli F, Brandi ML (2010) Bioinformatics for next generation sequencing data. Genes 1(2):294–307

Maglott D, Ostell J, Pruitt K, Tatusova T (2011) Entrez gene: gene-centered information at ncbi. Nucleic acids research 39(suppl 1):52–57

Malik R, Franke L, Siebes A (2006) Combination of text-mining algorithms increases the performance. Bioinformatics 22(17):2151–2157

Mardis ER (2008) The impact of next-generation sequencing technology on genetics. Trends Genet 24(3):133–41

Masso M, Vaisman II (2008) Accurate prediction of stability changes in protein mutants by combining machine learning with structure based computational mutagenesis. Bioinformatics 24(18):2002–2009

Metzker ML (2009) Sequencing technologies — the next generation. Nature Reviews Genetics 11(1):31–46

Mi Z, Shen K, Song N, Cheng C, Song C, Kaminski N, Tseng GC (2010) Module-based prediction approach for robust inter-study predictions in microarray data. Bioinformatics 26(20):2586–2593

Michalewicz Z (1998) Genetic algorithms + data structures = evolution programs. Springer

Miele V, Penel S, Daubin V, Picard F, Kahn D, Duret L (2012) High-quality sequence clustering guided by network topology and multiple alignment likelihood. Bioinformatics 28(8):1078–1085

Miller B, Goldberg D (1995) Genetic algorithms, tournament selection, and the effects of noise. Urbana 51:61,801

Mitra S, Hayashi Y (2006) Bioinformatics with soft computing. IEEE Transactions on Systems, Man and Cybernetics 36(5)

Mitra S, Das R, Banka H, Mukhopadhyay S (2009) Gene interaction - an evolutionary biclustering approach. Information Fusion 10(3):242–249

Moscato P (1989) On genetic crossover operators for relative order preservation. C3P Report 778

Mühlenbein H, Schlierkamp-Voosen D (1993) Predictive models for the breeder genetic algorithm i. continuous parameter optimization. Evolutionary computation 1(1):25–49

Mukhopadhyay A, Maulik U, Bandyopadhyay S (2009a) Finding multiple coherent biclusters in microarray data using varible string length multiobjective genetic algorithm. IEEE Transactions on Information Technology in Biomedicine 13(6)

Mukhopadhyay A, Maulik U, Bandyopadhyay S (2009b) A novel coherence measure for discovering scaling biclusters from gene expression data. Journal of Bioinformatics and Computational Biology 7(5):853–868

Mukhopadhyay A, Maulik U, Bandyopadhyay S (2010) On biclustering of gene expression data. Current Bioinformatics 5:204–216

Murakami Y, Mizuguchi K (2010) Applying the naive bayes classifier with kernel density estimation to the prediction of protein-protein interaction sites. Bioinformatics 26(15):1841–1848

Murali TM, Kasif S (2003) Extracting conserved gene expression motifs from gene expression data. In: Pacific Symposium on Biocomputing, pp 77–88

Nepomuceno J, Troncoso A, Aguilar-Ruiz J, et al (2011) Biclustering of gene expression data by correlation-based scatter search. BioData mining 4(3)

Ng KLS, Mishra SK (2007) De novo svm classification of precursor microrna from genomic pseudo hairpins using global and intrinsic folding measures. Bioinformatics 23(11):1321–1330

Nueda MJ, Conesa A, Westerhuis JA, Hoefsloot HC, Smilde AK, Talon M, Ferrer A (2007) Discovering gene expression patterns in time course microarray experiments by anova-sca. Bioinformatics 23(14):1792–800

Obayashi S (2002) Pareto solutions of multipoint design of supersonic wings using evolutionary algorithms. Adaptive Computing in Design and Manufacture V pp 3–16

Orengo C, Jones D, Thornton J (2003) Bioinformatics: genes, proteins and computers. BIOS Scientific Publishers Ltd

Panchenko A, Przytycka T (2008) Protein-protein Interactions and Networks: Identification, Computer Analysis and Prediction. Springer-Verlag London

Papanikolaou N, Pafilis E, Nikolaou S, Ouzounis CA, Iliopoulos I, Promponas VJ (2011) Biotextquest: a web-based biomedical text mining suite for concept discovery. Bioinformatics 27(23):3327–3328

Papp B, Notebaart R, Pál C (2011) Systems-biology approaches for predicting genomic evolution. Nature Reviews Genetics 12(9):591–602

Parejo JA, García J, Ruiz-Cortés A, Riquelme JC (2012) In: VIII Congreso Expañol sobre Metaheurísticas, Algoritmos Evolutivos y Bio-inspirados

Pavan M, Pelillo M (2003) A new graph-theoretic approach to clustering and segmentation. In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on, IEEE, vol 1, pp I–145

Peng H (2008) Bioimage informatics: a new area of engineering biology. Bioinformatics (Oxford, England) 24(17):1827–1836

Perez-Diez A, Morgun A, Shulzhenko N (2005) Microarrays for cancer diagnosis and classification. Eurekah Bioscience 1(6):503–509

Pevsner J (2009) Bioinformatics and Functional Genomics. Wiley-Blackwell

Ping H, Tzu L, Fang Sheu Y, Chuan T, Tang Y (2007) Finding homologous sequences in genomic databases

Pomeroy SL, Tamayo P, Gaasenbeek M, Sturla LM, Angelo M, McLaughlin ME, Kim JYH, Goumnerova LC, Black PM, Lau C, Allen JC, Zagzag D, Olson JM, Curran T, Wetmore C, Biegel JA, Poggio T, Mukherjee S, Rifkin R, Califano A, Stolovitzky G, Louis DN, Mesirov JP, Lander ES, Golub TR (2002) Prediction of central nervous system embryonal tumour outcome based on gene expression. Nature 415:436–442

Pongpanich M, Sullivan PF, Tzeng JY (2010) A quality control algorithm for filtering snps in genome-wide association studies. Bioinformatics 26(14):1731–1737

Pontes B, Divina F, Giráldez R, Aguilar-Ruiz J (2007a) Virtual error: a new measure for evolutionary biclustering. Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics pp 217–226

Pontes B, Giráldez R, Divina F, Martínez-Álvarez F (2007b) Evaluación de biclusters en un entorno evolutivo. In: IV Taller nacional de minería de datos y aprendizaje (TAMIDA), pp 1–10

Pontes B, Giráldez R, Divina F, Martínez-Álvarez F (2007c) Evaluación de biclusters mediante intra-fluctuaciones mínimas: un enfoque multi-objetivo. In: I Jornadas de Algoritmos Evolutivos y Metaheurísticas (JAEM 2007), pp 121–128

Pontes B, Divina F, Giráldez R, Aguilar-Ruiz J (2008) A novel approach for avoiding overlapping among biclusters in expression data. In: Eighth International Conference on Hybrid Intelligent Systems, 2008, IEEE, pp 813–818

Pontes B, Divina F, Giráldez R, Aguilar-Ruiz J (2009) Improved biclustering on expression data through overlapping control. International Journal of Intelligent Computing and Cybernetics 2(3):477–493

Pontes B, Giráldez R, Aguilar-Ruiz J (2010a) Measuring the quality of shifting and scaling patterns in biclusters. Pattern Recognition in Bioinformatics pp 242–252

Pontes B, Giráldez R, Aguilar-Ruiz JS (2010b) Ve^t: Una medida para la detección de patrones de desplazamiento y escalado en biclusters. In: VII Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB 2010), pp 429–436

Pontes B, Giráldez R, Aguilar-Ruiz J (2011) Evolutionary biclustering based on expression patterns. In: 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011), IEEE, pp 537–542

Pop M, Salzberg SL (2008) Bioinformatics challenges of new sequencing technology. Trends in Genetic 24(3):142–149

Prelić A, Bleuler S, Zimmermann P, et al (2006) A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22:1122–1129

Pritsker M, Liu Y, Beer M, Tavazoie S (2004) Whole-genome discovery of transcription factor binding sites by network-level conservation. Genome research 14(1):99–108

Procopiuc CM, Jones M, Agarwal PK, Murali TM (2002) A monte carlo algorithm for fast projective clustering. In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, ACM, New York, NY, USA, SIGMOD '02, pp 418–427

Qi Y, Zhang Y (2009) Text mining for bioinformatics: State of the art review. 2009 2nd IEEE International Conference on Computer Science and Information Technology (2):398–401

Quackenbush J (2002) Microarray data normalization and transformation. Nat Genet 32:496–501

Raman K (2010) Construction and analysis of protein-protein interaction networks. Automated Experimentation 2(1):2

Ramaswamy S, Tamayo P, Rifkin R, Mukherjee S, Yeang CH, Angelo M, Ladd C, Reich M, Latulippe E, Mesirov JP, Poggio T, Gerald W, Loda M, Lander ES, Golub TR (2001) Multiclass cancer diagnosis using tumor gene expression signatures. Proceedings of the National Academy of Sciences of the United States of America 98(26):15,149–15,154

Reis-Filho JS (2009) Next-generation sequencing. Breast Cancer Research 11(3)

Rosen GL, Reichenberger ER, Rosenfeld AM (2011) Nbc: the naïve bayes classification tool webserver for taxonomic classification of metagenomic reads. Bioinformatics 27(1):127–129

Sanger F, Air GM, Barrell BG, Brown NL, Coulson AR, Fiddes JC, Hutchison CA, Slocombe PM, Smith M (1977) Nucleotide sequence of bacteriophage phiX174 DNA. Nature 265(5596):687–695

Schachtner R, Lutter D, Knollmüller P, Tomé AM, Theis FJ, Schmitz G, Stetter M, Vilda PG, Lang EW (2008) Knowledge-based gene expression classification via matrix factorization. Bioinformatics 24:1688–1697

Schaffer J (1985) Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the 1st international Conference on Genetic Algorithms, L. Erlbaum Associates Inc., pp 93–100

Schena M, Shalon D, Davis R, Brown P (1995) Quantitative monitoring of gene expression patterns with a complementary dna microarray. Science 70:467–470

Segal E, Taskar B, Gash A, Friedman N, Koller D (2001) Rich probabilistic models for gene expression. Bioinformatics 17:243–252

Shannon W, Culverhouse R, Duncan J (2003) Analyzing microarray data using cluster analysis. Pharmacogenomics 4(1):41–52

Shen HB, Chou KC (2006) Ensemble classifier for protein fold pattern recognition. Bioinformatics 22(14):1717–1722

Shendure J, Ji H (2008) Next-generation dna sequencing. Nature Biotechnology 26(10):1135–1145

Sheng Q, Moreau Y, De Moor B (2003) Biclustering microarray data by gibbs sampling. Bioinformatics 19(suppl 2):196–205

Sherry ST, Ward M, Kholodov M, Baker J, Phan L, Smigielski EM, Sirotkin K (2001) dbsnp: the ncbi database of genetic variation. Nucleic Acids Research 29(1):308–311

Slonim DK (2002) From patterns to pathways: gene expression data analysis comes of age. Nature genetics 32 Suppl:502–508

Słowik A, Białko M (2004) Modified version of roulette selection for evolution algorithms–the fan selection. Artificial Intelligence and Soft Computing-ICAISC 2004 pp 474–479

Smyth GK, Smyth GK (2004) Linear models and empirical bayes methods for assessing differential expression in microarray experiments. Statistical applications in genetics and molecular biology 3(1):3

Soldatova LN, King RD (2005) Are the current ontologies in biology good ontologies? Nature Biotechnology 23(9):1095–1098

Srinivas N, Deb K (1994) Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2(3):221–248

Stein L, et al (2003) Integrating biological databases. Nature Reviews Genetics 4(5):337–345

Stelzl U, Worm U, Lalowski M, Haenig C, Brembeck F, Goehler H, Stroedicke M, Zenkner M, Schoenherr A, Koeppen S, Timm J, Mintzlaff S, Abraham C, Bock N, Kietzmann S, Goedde A, Toksöz E, Droege A, Krobitsch S, Korn B, Birchmeier W, Lehrach H, Wanker E (2005) A human protein-protein interaction network: a resource for annotating the proteome. Cell 122(6):957–68

Sturn A, Quackenbush J, Trajanoski Z (2002) Genesis: cluster analysis of microarray data. Bioinformatics 18(1):207–208

Suchard MA, Rambaut A (2009) Many-core algorithms for statistical phylogenetics. Bioinformatics 25(11):1370–1376

Swedlow JR, Goldberg IG, Eliceiri KW (2009) Bioimage Informatics for Experimental Biology. Annual Review of Biophysics 38(1):327–346

Tanay A, Sharan R, Shamir R (2002) Discovering statistically significant biclusters in gene expression data. Bioinformatics 18:136–144

Tanay A, Sharan R, Shamir R (2005) Biclustering algorithms: A survey. Handbook of computational molecular biology 9:26–1

Tang C, Zhang A (2005) Interrelated two-way clustering and its application on gene expression data. International Journal on Artificial Intelligence Tools 14(4):577

Tavazoie S, Hughes J, Campbell M, Cho R, Church G (1999) Systematic determination of genetic network architecture. Nature Genetics 22:281–285

Teng L, Chan L (2008) Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. Signal Processing Systems 50(3):267–280

Tibshirani R, Hastie T, Eisen M, Ross D, Botstein D, Brown P (1999) Clustering methods for the analysis of dna microarray data. technical report, Dept of Health Research and Policy, Dept of Genetics, and Dept of Biochemestry, Standford Univ

Tilstone C (2003) Dna microarrays: Vital statistics. Nature 424:610–612

Toğan V, Daloğlu AT (2008) An improved genetic algorithm with initial population strategy and self-adaptive member grouping. Comput Struct 86:1204–1218

Tusher VG, Tibshirani R, Chu G (2001) Significance analysis of microarrays applied to the ionizing radiation response. Proceedings of the National Academy of Sciences of the United States of America 98(9):5116–5121

Uramoto N, Matsuzawa H, Nagano T, Murakami A, Takeuchi H, Takeda K (2004) A text-mining system for knowledge discovery from biomedical documents. IBM Systems Journal 43(3):516–533

Velculescu VE, Zhang L, Vogelstein B, Kinzler KW (1995) Serial analysis of gene expression. Science 270(5235):484–487

Venter JC, Adams MD, Myers EW, Li PW, Mural RJ, Sutton GG, Smith HO, Yandell M, Evans CA, Holt RA (2001) The Sequence of the Human Genome. Science 291(5507):1304–1351

Viklund H, Elofsson A (2008) Octopus: improving topology prediction by two-track ann-based preference scores and an extended topological grammar. Bioinformatics 24(15):1662–1668

Wade CM, Giulotto E, Sigurdsson S, Zoli M, Gnerre S, Imsland F, Lear TL, Adelson DL, Bailey E, Bellone RR, Blöcker H, Distl O, Edgar RC, Garber M, Leeb T, Mauceli E, MacLeod JN, Penedo MCT, Raison JM, Sharpe T, Vogel J, Andersson L, Antczak DF, Biagi T, Binns MM, Chowdhary BP, Coleman SJ, Della Valle G, Fryc S, Guérin G, Hasegawa T, Hill EW, Jurka J, Kiialainen A, Lindgren G, Liu J, Magnani E, Mickelson JR, Murray J, Nergadze SG, Onofrio R, Pedroni S, Piras MF, Raudsepp T, Rocchi M, Røed KH, Ryder OA, Searle S, Skow L, Swinburne JE, Syvänen AC, Tozaki T, Valberg SJ, Vaudin M, White JR, Zody MC, Platform BIGS, Team BIWGA, Lander ES, Lindblad-Toh K (2009) Genome Sequence, Comparative Analysis, and Population Genetics of the Domestic Horse. Science 326(5954):865–867

Wang H, Wang W, Yang J, Yu P (2002) Clustering by pattern similarity in large data sets. In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, ACM, pp 394–405

Wang JTL, Zaki MJ, Toivonen H, Shasha D (eds) (2005) Data Mining in Bioinformatics. Springer

Wang W, Yang J, Muntz RR (1997) Sting: A statistical information grid approach to spatial data mining. In: Proceedings of the 23rd International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, VLDB '97, pp 186–195

Watson JD (2003) DNA The Secret of Life. Alfred A. Knopf

Wen X, Fuhrman S, Michaels G, Carr D, Smith S, Barker J, Somogyi R (1998) Large-scale temporal gene expression mapping of central nervous system development. National Acad Sciences, vol 95, pp 334–339

Whitley D, et al (1989) The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In: Proceedings of the third international conference on Genetic algorithms, Citeseer, vol 1, pp 116–121

Winsor GL, Lam DKW, Fleming L, Lo R, Whiteside MD, Yu NY, Hancock REW, Brinkman FSL (2011) Pseudomonas genome database: improved comparative analysis and population genomics capability for pseudomonas genomes. Nucleic Acids Research 39(Database-Issue):596–600

Witten I, Frank E (2005) Data Mining: Practical Machine Learning Tools And Techniques. Morgan Kaufmann Series in Data Management Systems, Morgan Kaufman

Wood IA, Visscher PM, Mengersen KL (2007) Classification based upon gene expression data. Bioinformatics 23(11):1363–1370

Xu R, Wunsch D, et al (2005) Survey of clustering algorithms. IEEE Transactions on Neural Networks 16(3):645–678

Xu X, Lu Y, Tung AKH, Wang W (2006) Mining shifting-and-scaling co-regulation patterns on gene expression profiles. In: 22nd International Conference on Data Engineering (ICDE'06), pp 89–99

Yang J, Wang W, Wang H, Yu PS (2002) δ–clusters: Capturing subspace correlation in a large data set. In: Proceedings of the 18th IEEE Conference on Data Engineering, pp 517–528

Yang J, Wang H, Wang W, Yu PS (2005) An improved biclustering method for analyzing gene expression profiles. International Journal on Artificial Intelligence Tools 14:771–790

Yang WH, Dai DQ, Yan H (2011) Finding correlated biclusters from gene expression data. IEEE Transactions on Knowledge and Data Engineering 23:568–584

Ye SQ, Lavoie T, Usher DC, Zhang LQ (2002) Microarray, sage and their applications to cardiovascular diseases. Cell Research 12(2):105–115

Yip K, Cheung D, Ng M (2004) Harp: A practical projected clustering algorithm. IEEE Transactions on Knowledge and Data Engineering 16(11):1387–1387

Yoo S, Choi J, Lee S, Yoo N, et al (2009) Applications of dna microarray in disease diagnostics. J Microbiol Biotechnol 19(7):635–646

Zhang J, Leung Y (2004) Improved possibilistic c-means clustering algorithms. IEEE Transactions on Fuzzy Systems 12(2):209–217

Zhao L, Zaki M (2005) Microcluster: Efficient deterministic biclustering of microarray data. Intelligent Systems, IEEE 20(6):40–49

Zhu X (2005) Semi-supervised learning literature survey. Tech. Rep. 1530, Computer Sciences, University of Wisconsin-Madison

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: A comparative case study and the strength pareto evolutionary algorithm. IEEE Transactions on Evolutionary Computation 3(4):257–271

Zitzler E, Thiele L, Zitzler E, Zitzler E, Thiele L, Thiele L (1998) An evolutionary algorithm for multiobjective optimization: The strength pareto approach. Citeseer

Part IV

Appendices

Appendix A

Improved Biclustering on Expression Data through Overlapping Control

One of the most relevant differences between clustering and biclustering approaches is the inclusion of possible overlapped elements in biclusters, while clusters do not typically allow one element to be part of two different groups. This characteristic has been studied in chapter 4, where the majority of the biclustering approaches reviewed use a diversity maintenance strategy in order to permit but control the overlapping amount among different solutions.

The most popular way in which overlapping is controlled within the biclustering community is to replace the elements contained in each found bicluster with random numbers. This strategy was initially proposed by Cheng & Church (2000) and has been usually adopted in sequential approaches, where one bicluster is obtained at a time. The original idea behind is the assumption that random values are unlikely to contribute to any future bicluster. Nevertheless, this strategy presents several drawbacks due to this elements masking, in which there is no real impediment for masked values to be included in further solutions.

The study presented in this appendix shows that the original algorithm by Cheng & Church (2000) wrongly estimates the quality of the biclusters after some iterations, due to random values that it introduces. We also present a modified version of the CC algorithm that improves the original one, as proven in the experimentation. The improvement is based on the use of a matrix of weights as the overlapping control mechanisms, instead of using random replacement. Empirical results show that our method is effective in order to improve the heuristic. It is also important to highlight that many interesting biclusters found by using this improved approach would have not

been obtained using the original algorithm. This study has been published in the International Journal of Intelligent Computing and Cybernetics (Pontes et al (2009)).

A.1 Overlapping Control Mechanisms

In this section we present the main drawbacks on using random replacement for overlapping control. Furthermore, we also present a different overlapping control mechanism based on the use of a matrix of weights and which does not make any use of random values.

A.1.1 Random Replacement

Random replacement was originally used by Cheng & Church (2000) in their algorithm, henceforth CC, where a substitution phase is performed any time a bicluster is found. This phase masks in the input matrix those elements contained in the bicluster with random values, aiming to prevent the algorithm from including those elements in future solutions. This strategy succeeds in avoiding the overlapping, however it presents two main drawbacks:

1. Being used in sequential approaches, as biclusters are discovered, more and more elements of the original expression matrix are lost, since they are substituted with random values. It follows that the expression matrix the algorithm is working on contains more and more random values as biclusters are being discovered. As a consequence, the algorithm may return biclusters that are obtained using random values. If a bicluster contains random values its computed MSR (or equivalent evaluation measure) is not real, since it is influenced by the presence of random values. This has a negative influence of the overall search process, since the algorithm cannot compute the real values of MSR for some biclusters. In the case of CC algorithm, these random values will be later replaced by the original ones, which makes the output bicluster of a different quality from the one the algorithm found.

2. Using random replacement some quality biclusters might not be found. For instance, if gene i and condition j are contained in a bicluster $\mathcal{B}$, the element b_{ij} is substituted by a random value in the expression matrix. This may prevent gene i to be included in other biclusters under the same condition j, even if it could have improved the quality of the bicluster, since some of its original expression values have been substituted by random values. In general, it is desirable to avoid overlapping

among biclusters, but not at the cost of loosing possible important interactions among genes.

These considerations show that the replacement strategy adopted by Cheng and Church may, on the one hand, prevent the discovering of interesting biclusters, and, on the other hand, yields the algorithm towards the discovering of biclusters considered to be interesting only because of random values they contain. This clearly illustrates the limitations of the replacement policy adopted by Cheng and Church. These aspects represent our main motivations for working out a new overlapping control mechanism.

A.1.2 Overlapping Control with a Matrix of Weights

In this section we present an overlapping control method based on the use of a weight matrix for sequential biclustering approaches.

Given an expression matrix $\mathcal{M}$, we say that two biclusters $\mathcal{B}_1$ and $\mathcal{B}_2$ are overlapped if there is at least one element $e_{ij} \in \mathcal{M}$ such that $e_{ij} \in \mathcal{B}_1$ and $e_{ij} \in \mathcal{B}_2$.

By controlling the level of overlapping among biclusters, we can decide whether a bicluster may be considered as a significative one, with respect to its overlapping percentage with the previously found biclusters. In our approach, we control the overlapping by means of a matrix of weights $\mathcal{W}$, in a similar way to the approach adopted in Divina & Aguilar-Ruiz (2006). $\mathcal{W}$ has the same dimension of the original expression data matrix, so that each element $\mathcal{W}(e_{ij}) \in \mathcal{W}$ represents a weight associated with $e_{ij} \in \mathcal{M}$. Initially all the elements of $\mathcal{W}$ are equal to zero, i.e., $\mathcal{W}(e_{ij}) = 0, \forall i, j$. Each time a bicluster $\mathcal{B}$ is found by the algorithm, $\mathcal{W}(e_{ij})$ is increased by one if $e_{ij} \in \mathcal{B}$. So, basically, $\mathcal{W}(e_{ij})$ indicates how many biclusters contain the element e_{ij}.

It follows that this matrix can be used to measure the overlapping of a new bicluster. We define the degree of overlapping of a bicluster $\mathcal{B}$ as:

$$P(\mathcal{B}) = \frac{\sum_{e_{ij} \in \mathcal{B}} \mathcal{W}(e_{ij})}{V(\mathcal{B})} \tag{A.1}$$

where $V(\mathcal{B})$ is the volume of the bicluster $\mathcal{B}$. $P(\mathcal{B})$ will be high for a bicluster whose elements are already contained in the previously found biclusters.

Since we aim at avoiding overlapping as much as possible, $P(\mathcal{B})$ can be used, in combination with MSR or any other evaluation measure, in order to reject too much overlapped biclusters. In order to do this, we need to define a criterion for establishing if $P(\mathcal{B})$ is to be considered high. According to equation A.1, $0 \leq P(\mathcal{B}) \leq nb$, where nb is the number of biclusters found so

far at any iteration of the algorithm. Thus, the upper limit of $P(\mathcal{B})$ will be different in each iteration of the algorithm.

In order to use $P(\mathcal{B})$ for rejecting biclusters in different iterations, it is convenient that the range of values $P(\mathcal{B})$ can assume is always the same in all iterations. For this purpose in equation A.2 we define the *overlapping factor* of a bicluster for the iteration nb.

$$P_{nb}(\mathcal{B}) = \frac{\sum_{i,j \in \mathcal{B}} \mathcal{W}(e_{ij})}{V(\mathcal{B}) \times nb} \tag{A.2}$$

Notice that $P_{nb}(\mathcal{B}) \in [0, 1]$, $\forall nb$. In this way, we can use $P_{nb}(\mathcal{B})$ to reject a bicluster $\mathcal{B}$ if $P_{nb}(\mathcal{B})$ is higher than a certain threshold. Moreover, biclusters found in later iterations are allowed to have more elements in common with the former ones. This is because $P_{nb}(\mathcal{B})$ tends to be smaller as nb increases. In other words, the biclusters with high overlapping are penalized more in the first iterations. By setting the overlapping threshold, the user can decide the level of overlapping among the biclusters.

A.2 CC Algorithm

As already mentioned, the original algorithm of Cheng and Church (CC) adopts a sequential covering algorithm in order to return a list of n biclusters from an expression data matrix. In order to assess the quality of a biclusters the algorithm makes use of MSR. This measure aims at evaluating the coherence of the genes and conditions of a bicluster $\mathcal{B}$. If a bicluster has a mean squared residue lower than a given value δ, then we call the bicluster a δ-bicluster. It follows that the smaller the value of MSR, the better the bicluster is considered to be. If a bicluster has a MSR equal to zero, it means that its genes fluctuate in exactly the same way under the subset of experimental conditions, and thus it can be considered a perfect bicluster.

Algorithm 7 shows a scheme of CC algorithm. It takes as input the expression matrix $\mathcal{M}$ and the threshold δ imposed on MSR. δ is used to reject non δ-biclusters. A list L of δ-biclusters is returned as output.

After preprocessing the missing values of $\mathcal{M}$ by replacing them with random numbers (line 1) and initializing the list of bicluster (line 2), the bicluster discovering process is repeated n times (lines 5-11). First, the bicluster $\mathcal{B}$ is initialized to the whole matrix $\mathcal{M}$. Next, the multiple node deletion phase (line 6) produces a δ-bicluster $\mathcal{B}_\delta$. This phase is based on the elimination of those rows or columns whose residue is higher than a certain value, depending on the MSR of the current matrix. Later, the single node deletion phase (line 7) removes the row or column from $\mathcal{B}_\delta$ with the higher residue

Algorithm 7: Cheng and Church's original algorithm (CC).

input : $\mathcal{M}$: expression matrix,
n: number of biclusters,
δ: threshold for MSR
output: L: list of n biclusters

```
1  preprocessMissingValues(M);
2  L ← {};
3  Bicluster B;
4  repeat
5  |  B ← M;
6  |  B_δ ← multipleNodeDeletion(B,δ);
7  |  B'_δ ← simpleNodeDeletion(B_δ,δ);
8  |  B''_δ ← addition(B'_δ,δ);
9  |  L ← L ⊕ B''_δ;
10 |  substitution(B''_δ,M);
11 until n iterations are reached;
12 return L;
```

and returns $\mathcal{B}'_\delta$. Next, the node addition phase (line 8) tries to enlarge the current bicluster $\mathcal{B}'_\delta$. This is done by adding those columns and rows that do not increase the residue of the matrix above the threshold δ. The obtained bicluster $\mathcal{B}''_\delta$ is stored in the list L (line 9). Finally, the substitution phase (line 10) replaces the elements of $\mathcal{M}$ that are contained in $\mathcal{B}''_\delta$ with random numbers. This substitution is applied in order to prevent overlapping among biclusters, since it is very unlikely that elements covered by existing biclusters would contribute to any future bicluster discovery.

Althougth this strategy seems to succeed in avoiding the overlapping, it presents two main drawbacks described in section A.1.1. After having performed a number of experiments, we have found that the percentage of random numbers present in the $\mathcal{M}$ can be very high during the execution of the algorithm. For example, for one of the datasets used in the experiments (Yeast dataset by Cho et al (1998)), we found that after 80 biclusters had been discovered, up to 50% of the elements of $\mathcal{M}$ had been replaced by random values.

Another point that has to be considered is that CC makes use of a threshold δ in order to reject biclusters: biclusters with MSR higher than δ are rejected. However if some elements of the biclusters are random, the MSR of this biclusters might be higher that δ, and thus be rejected. But again the MSR is influenced by the presence of random values. The MSR of the same bicluster with the original elements is different, and could therefore be lower

than δ. Also the opposite case may arise, i.e., a bicluster with estimated MSR lower than δ is accepted, but when the original values are used instead of the random ones, the MSR might increase to a level higher than δ. In this last case the bicluster should have been rejected.

Figure A.1 shows an example of such a situation. The bicluster represented in the figure has a MSR equal to 259.47. If the element shown in bold were a random value and the original value were 153 the MSR of the bicluster would drop to 0 when the MSR is computed with the original value. If a δ equal to, e.g., 100 were used, the bicluster depicted in Figure A.1 would be rejected, even if with the original values it represents a perfect bicluster.

$$B = \begin{pmatrix} 53 & 8 & 65 & 84 \\ 122 & 77 & 134 & \mathbf{60} \\ 55 & 10 & 67 & 86 \\ 73 & 28 & 85 & 104 \\ 140 & 95 & 152 & 171 \end{pmatrix}$$

Figure A.1: Example of a bicluster containing a random element (showed in bold). The original value of the element was 153.

The above considerations clearly show that the replacement strategy adopted by Cheng and Church may, on one hand, prevent the discovering of interesting biclusters, and, on the other hand, yields the algorithm towards the discovering of biclusters considered to be interesting only because of random values they contain. This clearly illustrates the limitations of the replacement policy adopted by Cheng and Church. These considerations represented our main motivations for developing the overlapping control methodology in section A.1.2 and also a modification on the original algorithm of Cheng and Church introducing this new strategy.

A.3 Re-adaptation of Cheng & Church Algorithm

In this section we describe the variations we incorporated to CC. We call the resulting algorithm CC-R. The main variation is represented by the removal of the substitution phase used in the original algorithm (line 10 in Figure 7), and the incorporation of an overlapping control mechanism. Since in CC-R elements contained in already found biclusters are not replaced by random values, a deterministic version of CC would always find the same bicluster. Therefore, our approach includes a different heuristic, consisting of some variations in order to render CC-R non-deterministic.

To this aim, we apply the following variations:

- The substitution phase has been replaced by the overlapping control mechanism. This mechanisms allows to reject biclusters with overlapping factors higher than ω and updates the matrix of weights $\mathcal{W}$.

- Multiple node deletion phase has been redefined in terms of the selection of the rows or columns to be deleted from the bicluster. Removing first those rows or columns that produce more overlapping with previous biclusters speeds up the convergence of the algorithm. The selection of the rows and columns is done using the matrix of weights $\mathcal{W}$.

- The selection mechanism for the columns or rows to be added in the node addition phase has been also redefined. Those columns or rows that are less overlapped with previously found biclusters are selected first, provided that their addition do not increase the matrix residue above δ. As in the previous variation, this selection is also based on $\mathcal{W}$. The redefinition of the node addition phase aims at finding biclusters with a low overlapping degree.

- Finally, the initial bicluster is randomly determined from the original microarray, with the exception of the first iteration where the initial bicluster is the whole matrix, as in CC.

The pseudo-code of CC-R is shown in Algorithm 8.

After preprocessing the missing values of $\mathcal{M}$, the variables L (list of biclusters), nb (counter for the loop or number of biclusters found), $\mathcal{W}$ (matrix of weight) and $\mathcal{B}$ (initial bicluster) are initialized. Notice that in the first iteration, the bicluster $\mathcal{B}$ is initialized to the whole matrix $\mathcal{M}$ (line 5), in order to take into account the whole set of genes and experimental conditions. Next, the while-loop is executed, where the three first phases (lines 7-9) are the re-adapted multiple node deletion phase, simple node deletion phase and re-adapted addition phase, respectively. These steps always produce δ-biclusters, that is $MSR(\mathcal{B}_\delta)$, $MSR(\mathcal{B}'_\delta)$ and $MSR(\mathcal{B}''_\delta)$ are smaller that δ. Notice that single node deletion (line 8) phase is always deterministic, since the selection of the row or column to be removed depends on their residues. Therefore, there is no adaptation of this phase in our approach.

Once $\mathcal{B}''_\delta$ is returned by the re-adapted addition phase, the overlapping control method is performed. If the overlapping factor of the bicluster $P(\mathcal{B}''_\delta)$ does not exceed the threshold ω, then nb is increased, the bicluster is included in the list L and $\mathcal{W}$ is updated (lines 13-15). If $e_{ij} \in \mathcal{M}$ belongs to $\mathcal{B}''_\delta$ then the element $w_{ij} \in \mathcal{W}$ is increased by one. If, on the other hand, the

Algorithm 8: Cheng and Church's re-adapted algorithm (CC-R).

input : $\mathcal{M}$: expression matrix,
n: number of biclusters,
δ, ω: thresholds
output: L: list of n biclusters

```
1  preprocessMissingValues(M);
2  L ← {};
3  nb ← 1;
4  matrix of weights W ← {};
5  bicluster B ← M;
6  repeat
7      B_δ ← multipleNodeDeletion(B,δ,W) [re-adapted];
8      B'_δ ← simpleNodeDeletion(B_δ,δ);
9      B''_δ ← addition(B'_δ,δ,W) [re-adapted];
10     if P(B''_δ)<=ω then
11         nb ← nb + 1;
12         L ← L ⊕ B''_δ;
13         forall the i,j do
14             if e_ij in M ϵ B''_δ then
15                 w_ij ← w_ij + 1;
16     B ← randomSelection(M);
17 until until nb =n;
18 return L;
```

overlapping factor is above ω, the bicluster $\mathcal{B}''_\delta$ is rejected, because it has too many common elements with the biclusters previously found.

Finally, a new bicluster $\mathcal{B}$ is randomly generated from the original dataset $\mathcal{M}$ to be used in the next iteration. The dimension of B is randomly chosen, as well as the specific genes and conditions belonging the bicluster. In the original algorithm, each iteration starts from the whole matrix $\mathcal{M}$, modified from the last iteration by the substitution phase. However, different experiments showed that starting from a random bicluster produced better results.

Figures A.2 and A.3 show the flow charts of the re-adapted multiple node deletion phase and addition phase, respectively. As can be seen in Figure A.2, multiple row/column deletion is only performed if the number of rows/columns is greater than 100. The way in which rows are chosen to be deleted is shown in the right flow chart in Figure A.2 (*deleteMultipleRows*). Columns are deleted in a similar way. Multiple node deletion phase ends up

once the bicluster has a lower value of MSR than the limit or the bicluster does not change after multiple rows and columns deletion.

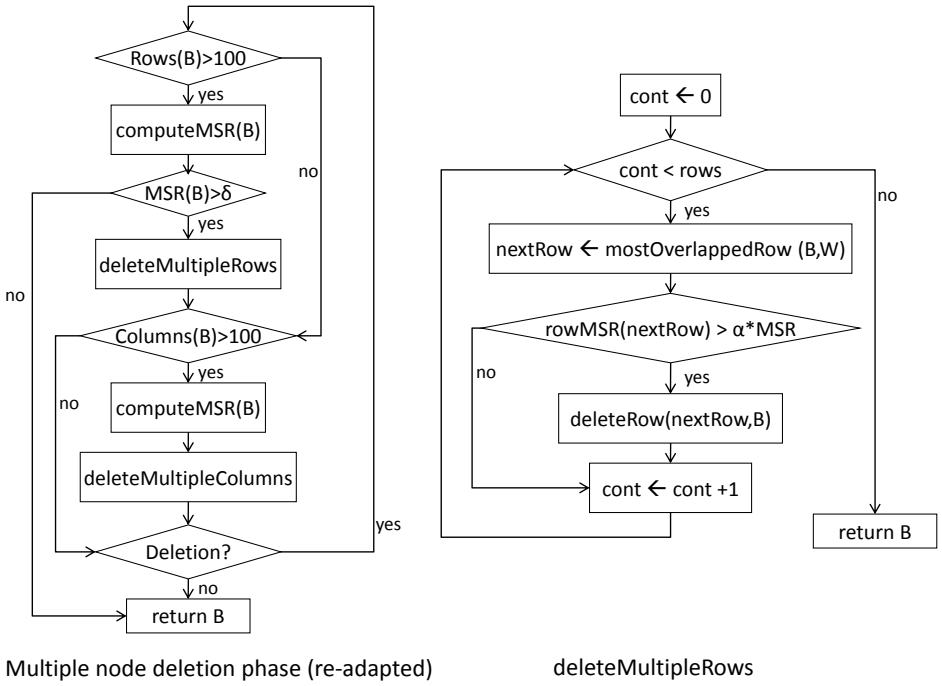


Figure A.2: Flow chart representing the re-adapted multiple node deletion phase.

Figure A.3 depicts the re-adapted addition phase. This phase iteratively add multiple rows, columns and inverse rows until no addition is performed. The way in which multiple rows are added to the bicluster can be seen in the right flow chart of the figure. Multiple columns are added in a similar way. Inverse rows are added in the same way as in the original CC algorithm.

A.4 Experimental Results

In order to test our approach we conducted experiments on three datasets:

1. Yeast *Saccharomyces cerevisiae* cell cycle expression dataset originated from Cho et al (1998). This datasets consists of 2884 genes and 17 conditions.

2. *Human B–cells* expression data originated from Alizadeh et al (2000). The human dataset consists of an expression matrix of 4026 genes and 96 conditions.

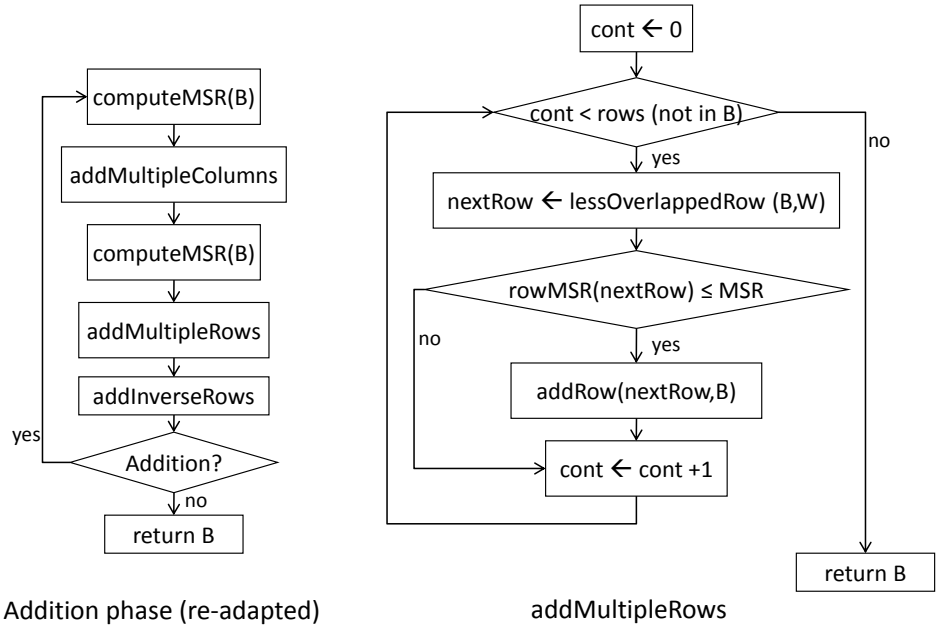


Figure A.3: Flow chart representing the re-adapted addition phase.

3. *Colon Cancer* dataset. This dataset originated from Alon et al (1999), and contains an expression matrix consisting of 2000 genes and 62 conditions.

All these datasets were preprocessed as in Cheng & Church (2000). The most important preprocessing operation regards missing values: missing values are replaced with random values, although it is known the existing risk that these random numbers can affect the discovery of biclusters (Yang et al (2002)). The expectation was that these random values would not form recognizable patterns.

Table A.1 shows the most relevant characteristic for each dataset. The first two columns show the number of genes and conditions, respectively. The third column shows the values of the δ limit that has been used. In the case of the yeast and human datasets they were taken from the original work Cheng & Church (2000), while for the other dataset we have established the value of δ following a procedure suggested in such a reference.

For each dataset, we have obtained 100 biclusters, using both the CC and CC-R. It is important to notice that most of the biclusters found with the last algorithm would have not been obtained using the original CC, since we have eliminated the substitution phase.In fact, this phase masked the values contained in each bicluster by introducing random numbers.

Dataset	#genes	#conds	δ
Yeast	2884	17	300
Human	4026	96	1200
Colon	2000	62	500

Table A.1: Main information for each dataset.

	Yeast	Human	Colon
MSR	124.80(72.86)	857.01(107.99)	389.02(76.99)
MSR(real)	498.46(306.08)	9940.90(8381.73)	2159.61(13343.43)
GeneVarMean	836.36(456.27)	10985.36(8780.13)	5929.48(16066.57)
Overlap	42.94%(36.30)	49.53%(23.62)	9.26%(18.40)
GenesMean	219.47(309.99)	271.52(234.25)	21.89(22.12)
CondsMean	7.25(3.42)	14.70(12.26)	8.81(7.24)

Table A.2: CC average results for each dataset. Standard deviation is given between brackets.

Tables A.2 and A.3 show the average results (and their standard deviations in brackets) obtained on each dataset (in columns) by the two algorithms, CC and CC-R, respectively. The first row gives the average MSR, the second row in Table A.2 shows the mean of the real MSR, i.e., when the MSR is calculated using the original values, and not the random values introduced in the substitution phase. These real MSR values have been calculated for each bicluster once the random values are substituted with the original ones. Table A.3 lacks of this row since there no random values are introduced in the expression matrix by CC-R. The row labelled *GeneVarMean* shows the gene variance. Next row (*Overlap*) represents the average overlapping for each bicluster with all the previous ones. Note that in Table A.2 this value also represents the mean number of random values that have been used in the algorithm. This is because values that are contained in more than a bicluster have been substituted with random values. Finally, rows *GenesMean* and *CondsMean* show the mean of the number of genes and conditions.

Furthermore, in order to statistically validate the results, we applied the Student's t-test. Using a confidence level of 0.5%, we can conclude that all the differences of results shown in Tables A.2 and A.3 are statistically significant.

From these tables, it is evident that the random values introduced in the expression matrix during the substitution phase negatively affect CC. In fact, the MSR computed considering the original values is, on average, higher than the specified δ. This means that many of the biclusters returned by the

	Yeast	Human	Colon
MSR	225.138(24.85)	1109.94(21.09)	435.31(13.84)
GeneVarMean	334.02(84.33)	1432.11(101.06)	742.64(13.99)
Overlap	94.55%(12.21)	91.21%(13.45)	94.81%(12.15)
GenesMean	758.18(212.89)	134.53(17.34)	134.48(18.15)
CondsMean	8.59(2.47)	45.66(7.41)	24.80(4.44)
Overlap2Bics	30.1%(17.58)	25.23%(0.39)	33.94%(0.65)

Table A.3: CC-R average results for each dataset. Standard deviation is given between brackets.

algorithm are not δ-biclusters, which is in contradiction with the specification of the algorithm. This fact is particularly evident for the human and the colon datasets, where the average real MSR is about eight and four time higher, respectively, than the δ used for these datasets.

On the other hand, all the biclusters obtained by CC-R are δ-biclusters, and the average MSR is much lower than the real MSR of the biclusters found by CC. These results alone show the limitations of the substitution phase adopted in CC. This substitution phase is effective for avoiding overlapping among biclusters, as it can be noticed by the overlapping percentages shown in Table A.2. However, this effectiveness is obtained at the cost of possibly producing biclusters that are not δ-biclusters.

As far as the gene variance is concerned, it can be noticed that, in general, CC obtained better results. However this result is influenced by the fact that MSR is much higher for the biclusters discovered by CC. In general biclusters with lower MSR have also a lower gene variance, and this explain the lower average gene variance for the biclusters obtained by CC-R.

Biclusters found by CC-R are characterized by a higher volume, even if the average MSR of the biclusters is lower than the MSR of biclusters found by CC. This is due to the overlapping policy adopted by CC. In fact, the random values introduced causes biclusters found in later iterations of the algorithm to have a very low volume. This is because random values are in general not included in biclusters, since they introduce noise that would cause the genes not be coherent under some conditions.

Table A.3 contains an additional row, named *Overlap2Bics*, which represents the average percentages of overlapped values between every pair of biclusters, for each dataset. Note that these amounts are considerably lower than the overlapping percentage of the whole set of biclusters (row *Overlap*).

In the following, we analyse the obtained results for each dataset individually. These results have been generated using the CC-R approach.

A.4.1 Yeast *Saccharomyces Cerevisiae* Cell Cycle Expression Dataset

The application of the algorithm to this dataset produces big biclusters, compared to the other two datasets. This is due to the intrinsic characteristics of the microarray, since genes contained in it are mostly flat, that is with low row variance. For this reason, the mean number of genes contained in the obtained biclusters is about 750. The number of conditions in the biclusters is about 9 on average, which means that the algorithm works well at discriminating some experimental conditions for each bicluster.

As the algorithm produces big biclusters, the existence of a certain overlapping percentage among them is inevitable. Nevertheless, by analysing the results, we have discovered that the overlapping percentage between two average-size biclusters does not exceed 50%. Naturally, the bigger a bicluster is, the higher the percentage of overlapping.

A special case of this situation is the first obtained bicluster which is specially big, since its overlapping percentage is always 0.

Figure A.4 shows two biclusters found on the yeast dataset. The graph presented, is the most commonly used representation of a bicluster. In such a graph, each line is relative to the expression level that a given gene assume under a particular experimental condition. For each bicluster, there are three pictures in the same column. The first one corresponds to the full obtained bicluster. The second and third ones correspond to the 20% and 10% of the genes in the original bicluster, respectively. These two not-complete biclusters are presented in order to visualize and test the quality of the original ones, since it is difficult to deduce genes shapes in such a big bicluster. These genes have been selected sequentially among the genes in the whole bicluster. The first gene out of each ten sequentially selected genes have been obtained for the 10% case, and the first gene out of five sequentially selected genes have been selected for the 5% case.

As aforementioned, gene expressions in yeast dataset show flatter tendencies than in the other datasets. For this reason, the obtained biclusters contain a big amount of genes, and they can be hardly analysed by their graphical representation. Nevertheless, the representation of a certain percentage of the original biclusters proves that genes show a certain correlation among them.

In this sense, we have obtained some quality biclusters that would have not been found by the original CC, since masking the values of a found bicluster with random ones prevents finding lots of real biclusters.

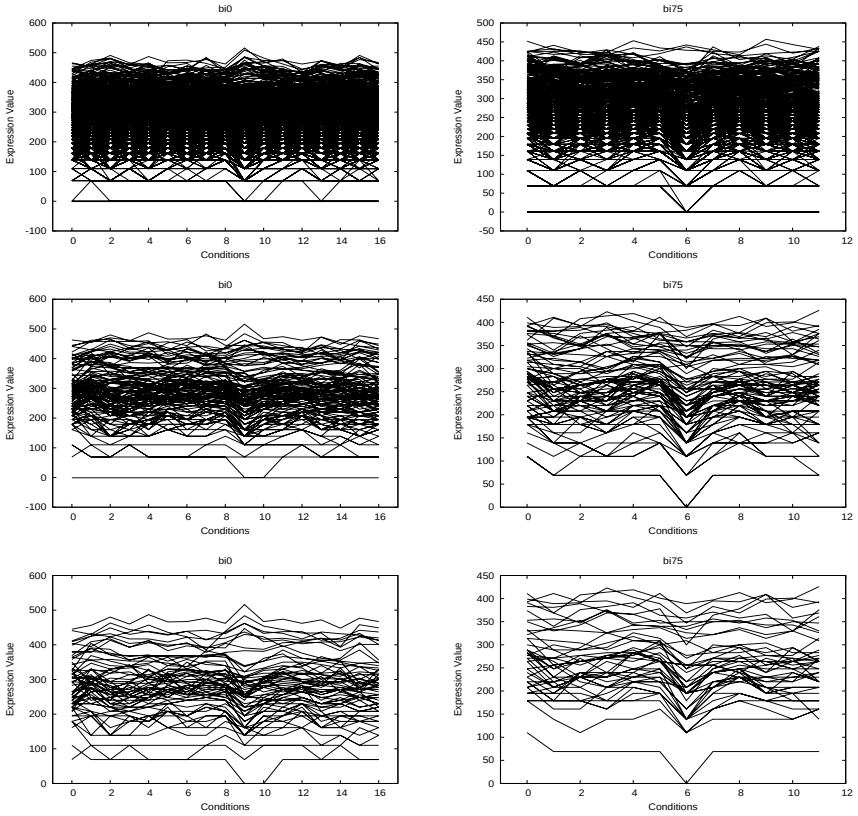


Figure A.4: Examples of biclusters found on the Yeast dataset. Each column shows three pictures of the same bicluster: full bicluster, 20% and 10% of the genes, respectively. X-axis represents each of the experimental conditions in the bicluster, while Y-axis represents the expression level of the genes in each bicluster.

A.4.2 Human B–cells Expression Data

Biclusters found on the Human dataset are significantly smaller than the ones discovered from the Yeast dataset. The mean of the number of genes in the biclusters is about 135. This result is accentuated even more by the fact that the expression matrix relative to Human dataset is bigger than that of the Yeast dataset. The main reason for this situation is the intrinsic characteristics within the datasets. Genes in Human dataset do not present such a flat behavior as in the Yeast dataset. On the contrary, gene expression varies significantly among the different experimental conditions.

The average of the number of conditions for biclusters on Human dataset is about 45, almost the 50% of the total number of conditions. Again, the first bicluster presents the highest volume, since its overlapping percentage is always 0. This bicluster contains 83 genes and 96 conditions, which represents the whole set of experimental conditions. Therefore, it is one of the most overlapped biclusters. Nevertheless, the amount of overlapping between two medium-sized biclusters does not exceed 30%.

Figure A.5 represents four biclusters of the 100 found for this dataset. We can appreciate in these pictures that genes in the same bicluster are strongly correlated. Their expression levels vary in unison under the same subset of conditions. Furthermore, the expression levels of the genes are within the same range of values, for this reason all of them are grouped and it is difficult to differentiate each gene.

A.4.3 *Colon Cancer* dataset

Colon Cancer dataset consists of 2000 genes and 62 conditions. The mean number of genes in the biclusters for this dataset is 135, while the mean number of conditions is 25. In this case, biclusters contain almost the same number of genes as those obtained from the human dataset. Considering the volume of both datasets, biclusters from Colon dataset are expected to be more overlapped among them.

In fact, this dataset produces biclusters with a mean of 40% of common values between two given biclusters.

Figure A.6 shows four biclusters obtained for Colon dataset. They resemble the ones found on the Human dataset, since we can clearly see that each bicluster contains correlated genes. All genes in a certain bicluster follow the same trend, and they are also within a close range of expression values.

In this case, the first obtained bicluster (shown in Figure A.6 with label *bi*0) does not correspond with the highest number of genes, but with the highest number of experimental conditions. It is made up of 55 conditions,

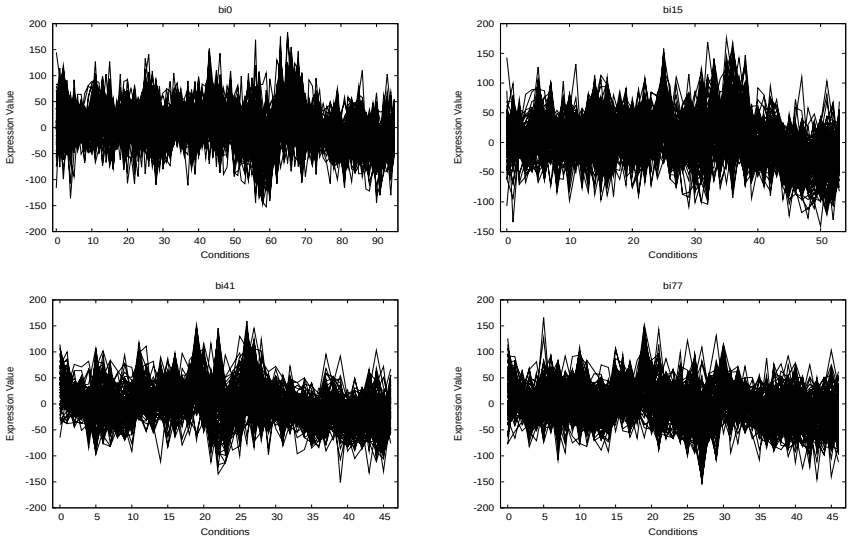


Figure A.5: Examples of biclusters found on the Human dataset. X-axis represents each of the experimental conditions in the bicluster, while Y-axis represents the expression level of the genes in each bicluster.

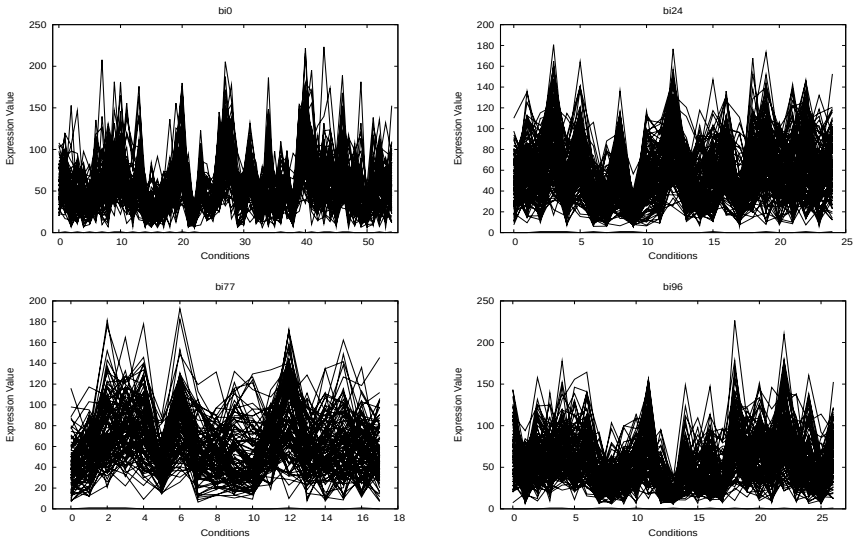


Figure A.6: Examples of biclusters found on the Colon dataset. X-axis represents each of the experimental conditions in the bicluster, while Y-axis represents the expression level of the genes in each bicluster.

which represents the double of the mean numbers of conditions of the whole set of biclusters.

A.4.4 Comparison

Even if the aim of this work was to assess the validity of the mechanism for controlling the overlapping among bicluster, we nevertheless include a comparison with another state of the art biclustering algorithm by Divina & Aguilar-Ruiz (2006), called SEBI. SEBI is an algorithm based on Evolutionary Computation that shown excellent performance at finding patterns in gene expression data. Furthermore, SEBI adopts a strategy for avoiding overlapping similar to the one presented in section A.1.2 and used in this work.

Table A.4 shows the average results (and their standard deviations in brackets) obtained on each dataset by the algorithm SEBI for what concerns the average MSR and the average dimension (genes and conditions) of the biclusters found. Thus, we can compare this table against Table A.3 in order to test performance of our approach.

	Yeast	Human	Colon
MSR	205.18(4.49)	1028.84(29.19)	492.46(6.23)
GenesMean	13.61(10.38)	14.07(5.39)	9.86(4.51)
CondsMean	15.25(1.37)	43.57(6.20)	40.91(8.00)

Table A.4: SEBI average results for each dataset. Standard deviation is given between brackets.

Regarding MSR, CC-R shows very similar results against SEBI and all of them are lower than the δ used for these datasets. However, if we compare these values with those shown in Table A.2 for CC, we can notice that the average real MSR is much higher.

On the other hand, we can see that CC-R is capable of finding biclusters characterized by a higher number of genes than the ones found by SEBI. This is a very important aspect, since the aim of these biclustering method is to find δ–biclusters with maximum size.

A.5 Conclusions

In this appendix we have shown some variations that can be applied to the CC algorithm in order to overcome its shortcomings. The original algorithm is very effective at discovering biclusters, however, after some iterations it

starts to work with more and more random values in the expression matrix, due to the substitution phase used. This causes the algorithm to estimate wrongly the MSR of the biclusters. We have presented an alternative method for avoiding as much as possible overlapping among biclusters.

Our work is based on a matrix of weights, that is used to estimate the overlapping of a bicluster with already found ones. We have defined an overlapping factor which is used in order to reject biclusters if their overlapping is above a certain threshold. In this way the algorithm is always working with the original expression data, and so the biclusters it discovers contain only original data. Since no random values are introduced in the expression matrix, we have included other modifications to the algorithm in order to render it non deterministic.

Results show that many biclusters found by CC have a MSR that is higher than δ, due to the random values they contain. This is an important shortcoming of CC, since this may yield the algorithm to discovering biclusters that are not δ biclusters. It is also important to notice that many biclusters found by CC-R would have not been obtained using the original CC. This is due to the fact that CC does not work with the original expression matrix. This causes that many biclusters are masked by random values.