Abstract of "Informed Structural Priors for Bayesian Networks: Applications in Molecular Biology Using Heterogeneous Data Sources" by Sonia M. Leach, Ph.D., Brown University, May 2006.

The main goal of this thesis is to investigate ways in which a collection of separate prior information sources, of varying scale and reliability, can be integrated in a principled manner to reveal a more complete understanding of a problem domain. We consider a case study from the biological domain: inferring a network of gene interactions. Biology has a rich and diverse body of knowledge already available upon which to leverage the task of learning such a model; the challenge is in developing meaningful ways to encode such prior knowledge which take into account the heterogeneity of the individual prior data sources. We consider a host of 'experts' which specify explicit or implicit interactions between genes, such as physical interaction between the genes, shared biological function or even co-occurrence of literature references. Though each individual source may be a poor indicator for gene interaction when used alone, we demonstrate techniques for combining the experts to formulate a consensus belief or likelihood that any two genes are related. The resulting probability distributions of gene relationships are then used in two distinct ways. The first is in providing structural priors for learning Bayesian Networks. We show how using a prior distribution over interactions between genes can significantly increase the speed and quality of search for high scoring Bayesian Networks when learning from gene expression data. Our studies make use of simulated data from a model of ICU ventilator management (ALARM), a benchmark for Bayesian Network learning, as well as real-world biological data from the Yeast genome. The second application uses the consensus priors over gene relationships to create visualization tools for large scale datasets. We provide results for examples in both Yeast and Mouse which demonstrate how a collection of weakly suggested or sometimes unreliable relationships can be combined to create a powerful and useful working model of interactions, allowing biologists to better understand an overwhelming amount of information.

Abstract of "Informed Structural Priors for Bayesian Networks: Applications in Molecular Biology Using Heterogeneous Data Sources" by Sonia M. Leach, Ph.D., Brown University, May 2006.

The main goal of this thesis is to investigate ways in which a collection of separate prior information sources, of varying scale and reliability, can be integrated in a principled manner to reveal a more complete understanding of a problem domain. We consider a case study from the biological domain: inferring a network of gene interactions. Biology has a rich and diverse body of knowledge already available upon which to leverage the task of learning such a model; the challenge is in developing meaningful ways to encode such prior knowledge which take into account the heterogeneity of the individual prior data sources. We consider a host of 'experts' which specify explicit or implicit interactions between genes, such as physical interaction between the genes, shared biological function or even co-occurrence of literature references. Though each individual source may be a poor indicator for gene interaction when used alone, we demonstrate techniques for combining the experts to formulate a consensus belief or likelihood that any two genes are related. The resulting probability distributions of gene relationships are then used in two distinct ways. The first is in providing structural priors for learning Bayesian Networks. We show how using a prior distribution over interactions between genes can significantly increase the speed and quality of search for high scoring Bayesian Networks when learning from gene expression data. Our studies make use of simulated data from a model of ICU ventilator management (ALARM), a benchmark for Bayesian Network learning, as well as real-world biological data from the Yeast genome. The second application uses the consensus priors over gene relationships to create visualization tools for large scale datasets. We provide results for examples in both Yeast and Mouse which demonstrate how a collection of weakly suggested or sometimes unreliable relationships can be combined to create a powerful and useful working model of interactions, allowing biologists to better understand an overwhelming amount of information.

Informed Structural Priors for Bayesian Networks: Applications in Molecular Biology Using Heterogeneous Data Sources

by
Sonia M. Leach
B. S. E., Bucknell University, 1994
Sc. M., Brown University, 1996

A dissertation submitted in partial fulfillment of the
requirements for the Degree of Doctor of Philosophy
in the Department of Computer Science at Brown University

Providence, Rhode Island
May 2006

This dissertation by Sonia M. Leach is accepted in its present form by
the Department of Computer Science as satisfying the dissertation requirement
for the degree of Doctor of Philosophy.

Date _____ _____

Thomas Dean, Director

Recommended to the Graduate Council

Date _____ _____

David Laidlaw, Reader

Date _____ _____

Lawrence Hunter, Reader
(University of Colorado)

Approved by the Graduate Council

Date _____ _____

Sheila Bonde
Dean of the Graduate School and Research

Vita and Preface

During my quest to define 'what I want to be when I grow up', I was seduced by all three temptresses of life: beauty, money, and science. And the greatest of these is science. At age six, my ambition was to become a hairdresser and the carnage of abandoned dolls left in the dusty corners of my friends' and cousins' houses meant no shortage of willing recipients of my attention. By age 16, owning a drivers license and a temperamental 1981 VW Rabbit brought a greater appreciation for the source of gas money and the like so my ambition escalated as a result of a comment made by my best friend at the time, Michael Moreau. His aunt was a vice president of some sort at IBM and her quoted salary of $235K a year lured me to dreams of being 'the boss.' My adolescent mind translated this into a career in Business Administration so when time came to research colleges, I gathered every pamphlet with that illustrious specialty listed. Fate turned my hand upon receipt of the Society of Women Engineers Award for Excellence in Math and Science, an honor bestowed upon me both my junior and senior year in high school.

The second accolade I considered a mistake since awarding of the first my junior year lead me to believe it was an award for Juniors. Puzzlement over the duplicate honor caused me to reconsider the meaning of the award that suddenly seemed to define my teachers' opinion of me. My first thought, embarrassingly in hindsight, was that I had no idea what an engineer was so I turned to my faithful friend, the World Book Encyclopedia. The phrasing still resonates with me since the description implied that engineering was a very difficult, almost impossible, career requiring someone with superhuman math and science skills, creativity and motivation. I was born in and grew up in Vermont, where stoicism and pragmatism are the two most valued character traits so I took as a personal afront the suggestion that engineering might be above my abilities. I decided at that moment that I was going to be an Engineer. My recently discovered prowess with the beloved RadioShack TRS-80 decided my discipline of Computer Engineering and I happily renewed my search for colleges using this criterion.

Brown was my first choice but typical of Vermont, a blizzard kept me from getting my application in before the deadline. Well, there may have been other factors but this was my reasoning. Instead, I was accepted at Bucknell University and sometimes serendipity is a wiser master of our fates. As a small, primarily undergraduate university, Bucknell allowed very close contact with the faculty and access to incredible opportunities. Starting with the summer of my junior year, I began research in logic programming with Professor James Lu and that experience changed my life. I began to

know that scientific knowledge is not the static text of books, but evolves through ongoing research by individuals who contribute to journals, conferences and workshops. In short, I was exposed to life as an academic. Professor Lu mentored that research as well as my honors thesis and by the time I graduated, I was co-author on two papers. As graduation neared, I was again faced with the question of 'what I want to be when I grow up' and it was Professor Lu who provided the answer. He insisted that graduate school would be a perfect place for me. And this time, all factors dictated that Brown was indeed the right choice.

Within the Brown Computer Science department, my interest quickly gravitated towards Artificial Intelligence, due in part to the magnetic personality of Professor Leslie Pack Kaelbling. She and her colleague Professor Thomas Dean, who was on sabbatical the year I began the PhD program, introduced me to the use of probability and statistics to solve practical problems. So began my education in probabilistic graphical models. The marriage between mathematical theory and real-world problems proved irresistible to my pragmatic nature in a way that my earlier largely theoretical work in logic programming did not. I helped Tom organize a workshop on Learning Dynamical Systems that was designed to bring together researchers from diverse fields in an effort to identify commonalities among their work. I attended a summer school on Bayesian networks in Erice, Sicily. At both these events, I was introduced to the key investigators in the field, finally able to put faces to the names I had read about in the textbooks and research articles, leaving me in stunned reverence and awe amidst such scientific celebrities.

During the Dynamical Systems workshop, a fellow graduate student James Kurien secured an internship at NASA Ames with Dr. Brian Williams. Eleven months later, Jim was still at NASA and told me about summer internship opportunities there. I joined the group in the summer of 1997 as they were working on the Deep Space 1 project, creating software to fly the spacecraft remotely. Combined with the Pathfinder project whose data was being analyzed by a group in the room below where I worked, it was a very exciting time to be at Ames. Leslie and Tom were concerned that they had lost another one to Brian's group when after four months, I was still there. I eventually returned to Brown, renewed with a fascination for science and math but discouraged because I had not yet found my place. I had reached that point in every grad student's life when you begin to wonder why you are doing a PhD at all. That is when I discovered Computational Biology.

I was home for Thanksgiving break, utterly depressed about my lack of direction in my PhD career. I was in my fourth year of the program, a time when I should have some plan for finishing my degree. My mother, in an attempt to be understanding, said that I shouldn't be so sad, I just had to write a paper, right? I realized that it must be hard to understand the import of a PhD thesis for someone who wasn't embroiled in academia. Somehow that thought made me even sadder because it implied that my PhD lacked not only direction but purpose. During a conversation about these musing with my friend Zewei Chen that evening, he asked if I could do anything in the world without pressure of failing to meet any expectations, what would it be. I answered that I could study languages, or Egyptology, or Biology – the latter choice motivated by the fact that he and I had been consulting a biology textbook earlier that week to find out if the knee joints in horses' legs

were reversed relative to those of other mammals.

Coincidentally, among the mail accumulated for me at home was that month's issue of Conduit, the newsletter for the Brown CS department. The feature article was about Professor Franco Preparata and his efforts to start an undergraduate discipline in Computational Biology. That issue also featured an editorial by Jonathan Monsarrat who was a graduate student at Brown when I started but had left the program to start a 3-D gaming company. Jon Mon, as he was affectionately called around the department, was known for his energetic personality and limitless ambition. One always wondered what he would do next and in his editorial was an update of his activities which included the sale of the company and a new job working with a computational biology firm. He cited that biology was ripe with interesting AI problems. I went to sleep that fateful Wednesday night thinking about what I had read regarding Professor Preparata's vision for computational biology and the exciting life turns of Jon Mon. At 1:00 am, I awoke with a start; it suddenly occurred to me in the dead of the night that I too could do biology. That I, like Jon Mon, could contribute my knowledge and be useful in solving the wealth of problems in biology. From that moment on, Science – the ultimate temptress – had me firmly in her clutches. I was so eager to begin researching the possibilities that I resented the holiday that required me to stay at home for the long weekend instead of allowing me to immediately immerse myself in the field.

The next few months saw a flurry of activity as I read everything I could about the exciting and emerging discipline of Computational Biology and tried to define my place within it. I was ready to begin my entire PhD career anew if necessary. Tom, who was now my primary advisor, showed amazing patience and compassion at that time as I struggled to define how I could contribute. I wrote to people in the field, asking advice about appropriate conferences and journals, attended my first RECOMB held that year in New York. I searched for independent sources of funding since my previous grant award was no longer appropriate. I received a NASA Space Grant for the following year, to share my love for math and science by lecturing to elementary and secondary school children on space-related topics. I felt the fastest route to gaining the necessary background was to pursue another summer internship and Dr. Larry Hunter at the National Library of Medicine, part of the National Institutes of Health, agreed to sponsor me. The project involved the search for transcriptional regulators by studying patterns of co-expression of genes using gene expression data, a new biological assay at the time. I continued to work with Larry through the year I was funded by NASA, and eventually discovered that I had exactly the background in probabilistic graphical models that was required for the project. I had finally found my place.

My collaboration with Larry continued in the following years, eventually resulting in my relocation first to Maryland to be at the NIH campus, and eventually to Colorado as Larry accepted a faculty position in the Pharmacology department at the University of Colorado School of Medicine. Our move was welcomed by the biologists on the campus as we began statistical analysis of the large volume of data generated here. In 2000 when I first moved to Colorado, the University was the largest single-lab consumer of gene expression technology from the Affymetrix Corporation, which

meant that I now had exclusive access to an incredible volume of data and nearly limitless opportunities for collaboration with biologists. After working with these researchers and identifying their requirements for analysis, I became proficient in Biostatistics but was dismayed at my own inability to provide them with more powerful tools to facilitate the generation of meaningful explanations for their data. It was precisely this frustration at the lack of integration and automation of hypotheses that motivated the work of this thesis, particularly the work presented in the Visualization chapter. The research presented here represents a first pass at tackling the integration problem in order to compliment the biologists' own knowledge. In the future, I wish to continue understanding and improving upon the available resources, creating new tools through interactive discussions with the very people who can most benefit from my knowledge and experience in bringing computational approaches to practical problems.

As I finish the thesis and look back at how all the pieces eventually came together, I have to admit that I finally agree with my mother. I was too concerned about doing the 'greatest' thing that I did not have the right sense of appreciation for all my contributions along the way. This document represents only a small part of the knowledge I have gained during my PhD and the research that I have accomplished in support of this work. The scope of my study, like all PhD theses, has been so much larger than is captured in these pages; this is, as Mom said, just another paper.

Acknowledgements

During the course of my PhD experience, I have had the distinct pleasure of being mentored by Professors Leslie Pack Kaelbling, Thomas Dean, Brian Williams and Larry Hunter. A special thanks to my committee members Professor Thomas Hofmann and Professor David Laidlaw and my undergraduate advisors Professors James Lu and Jerud Mead. I would like to thank my fellow colleagues at NASA Ames, the National Library of Medicine, the National Cancer Institute, and the University of Colorado. Collaborations at all these institutions have been rewarding due to the extreme pleasure of working with individuals there, most notably James Kurien, Bill Millar, John Bresina, Ioannis Tsamardinos, Pedrito Maynard-Reid II, Lisa McShane, Ron Taylor, Frank Accurso, Jamie Wooldridge, Razvan Lapadat, Sanjiv Bhave, Peggy Neville and Michael Rudolph. The Visualization chapter could not have been accomplished without Michael's painstaking efforts. Thanks to Jon Monsarrat and Professor Franco Preparata for inspiring me to study Computational Biology. Thanks to my officemates along the way, Michael Littman, Jose Castaños, Song Zhang, Purnima Mungalachetty, Brigid Bucci, Tzu Phang, and Anis Karimpour–Fard.

Thanks go to Mair Churchill for providing reliability assessments for the prior information sources, to Helen Johnson and Michael Bada for their invaluable help with my thesis defense presentation, to Kevin Cohen for his assistance with references in Natural Language Processing. Also thanks to Laura Brown of Vanderbilt for the helpful discussion of parameter priors. Extreme thanks go to Kevin Murphy who I first met in 1996 and had a delightful meeting of the minds about implementation of graphical models. Kevin's publically available Bayesian Network Toolbox for Matlab was critical in providing a guide and support code for many of the Bayesian network algorithms needed for this thesis. The AI community is indebted to him for this incredible resource.

Personal thanks go to my friends Jon and Dan Horne, Sarah and Steve Corning, Jeff Odom, Julie Ritner, Zewei Chen and Luis Ortiz who came from all over the country for support at the thesis defense. Thanks to Olga Karpenko and Tomer Moscovich for lending me space on my many visits back to RI. I would like to thank Hagit Shatkay for her excellent example during my time at Brown, and Tony Cassandra for his help with the yeast expression data as well as providing comic relief with his healthy attitude about grad school. Special thanks to Tara Gilligan, Maricel Kann, Galina Shubina, Vaso Chatzi and Remco Chang for their continued support over the years. I extend thanks again to Michael Rudolph for putting up with me in the final stretches of the thesis. Lastly, I would like to thank my family, who although they don't always understand me, love me anyway.

Dedication

This thesis is dedicated to my friend Timothy Bovat who continually taunted me with the knowledge that he would have three degrees before I finished even one. Sadly, he died in February 2000 before he was able to witness the equality in degree count. This thesis is for you, Maggot. F.F.N.F.B.F.M.F.

Contents

List of Tables xiv

List of Figures xv

1 Introduction 1

2 Molecular Biology and Biological Data Sources 8

2.1 Gene Interaction Networks 8

2.2 Interaction Information Sources 10

2.2.1 Explicit Interaction Sources 12

2.2.2 Implicit Interaction Sources 14

2.2.3 Previous Uses of Interaction Sources and Interaction Networks 16

2.3 Regulatory Networks 17

2.4 Microarray Expression Data For Learning Regulatory Networks 18

2.4.1 Previous Uses of Expression Data 21

2.5 Regulatory Networks as Bayesian Networks 22

2.5.1 Gene Interaction Networks as Structural Priors 23

3 Bayesian Networks 25

3.1 Definition of Bayesian Networks 25

3.2 Learning Bayesian Networks 26

3.2.1 Learning Structure: Search and Score 27

3.2.2 Parameter Priors: the BD, BDe, and BDeu metrics 29

3.2.3 Learning Parameters: ML and MAP estimates and the EM algorithm 31

3.2.4 Learning Parameters: Noisy-OR 32

3.3 Learning Bayesian Networks: Updated Framework 34

4 Structural Priors: Previous Approaches 35

4.1 Types of Structural Priors 35

4.1.1 Uninformed Priors 35

4.1.2 Semi-informed Priors 36

4.1.3 Informed Priors . . . 37
4.2 Informed Structural Priors: Previous Approaches . . . 37
4.2.1 Variable Parameter-centric Approach . . . 37
4.2.2 Structure-centric Approach . . . 38
4.2.3 Data-centric Approach . . . 38
4.2.4 Edge Parameter-centric Approach . . . 39
4.3 Limitations of Previous Approaches for Our Application . . . 40

5 Structural Priors: Our Approach **41**
5.1 Method 1: LinOP . . . 41
5.2 Method 2: Noisy-OR . . . 46
5.3 Method 3: BAYES . . . 47
5.4 Method 4: PRM . . . 51

6 Computing Consensus Likelihood Functions **59**
6.1 Consensus Likelihood for ALARM . . . 59
6.1.1 ALARM Experts . . . 60
6.1.2 ALARM Reliability Assignments . . . 61
6.1.3 Computing ALARM Consensus Likelihoods . . . 62
6.2 Consensus Likelihood in Yeast . . . 64
6.2.1 Yeast Data and Experts . . . 64
6.2.2 Yeast Reliability Assignments . . . 68
6.2.3 Computing Yeast Consensus Likelihoods . . . 71
6.3 Consensus Likelihood in Mouse . . . 84
6.3.1 Mouse Data and Experts . . . 84
6.3.2 Mouse Reliability Assignments . . . 86
6.3.3 Computing Mouse Consensus Likelihoods . . . 87

7 Learning Gene Regulatory Networks **94**
7.1 Experimental Design . . . 94
7.1.1 Learning Algorithm Details . . . 95
7.1.2 Individual Evaluation Criteria for Structural Priors . . . 97
7.1.3 Relative Evaluation Criteria for Structural Priors . . . 99
7.2 ALARM Bayesian Network Learning . . . 104
7.2.1 ALARM Results Using Individual Evaluation Criteria . . . 104
7.2.2 ALARM Results Using Relative Evaluation Criteria . . . 111
7.3 Yeast Bayesian Network Learning . . . 115
7.3.1 Gene Expression Dataset for Learning in Yeast . . . 115
7.3.2 Yeast Results Using Individual Evaluation Criteria . . . 119
7.3.3 Yeast Results Using Relative Evaluation Criteria . . . 126

7.3.4 Conclusions from Yeast Results . . . 127

8 Visualizing Gene Interaction Networks 137
8.1 Previous Approaches to Network Visualization . . . 143
8.2 Datasets . . . 146
8.2.1 Well-Studied Mouse Example (MG122) . . . 146
8.2.2 De Novo Mouse Example (MG449) . . . 156

9 Overall Conclusions and Future Work 159

A Discretization 162
A.1 Characterization of Discretization Methods . . . 163
A.1.1 Top-down versus Bottom-up Iteration . . . 165
A.2 Discretization Methods . . . 167
A.2.1 Method 1: Equal Frequency Interval (EqFreqInt) . . . 167
A.2.2 Method 2: Equal Interval Width (EqIntWid) . . . 168
A.2.3 Method 3: Standard Deviation (Std) . . . 168
A.2.4 Method 4: Fold Change (FC) . . . 168
A.2.5 Method 5: Nearest Neighbor (NN) . . . 169
A.2.6 Method 6: Monothetic Contrast Criterion (MCC) . . . 169
A.2.7 Methods 7 and 8: Total Mutual Information (TMIDLC) . . . 169
A.2.8 Methods 9 and 10: Relative Unsupervised Discretization (RUDE) . . . 170
A.2.9 Method 11: Multivariate Discretization (MVD) . . . 173
A.2.10 Methods 12-16: Individual correlation of Discrete to Real (IDR) . . . 176
A.2.11 Methods 17-21: Spearman Individual correlation of Discrete to Real (SIDR) . 187
A.2.12 Methods 22-26: Pairwise correlation of Discrete to Real (PDR) . . . 187
A.2.13 Methods 27-31: Spearman Pairwise correlation of Discrete to Real (SPDR) . 195
A.3 Experimental Design . . . 195
A.3.1 Performance Evaluation Measures . . . 196
A.3.2 Datasets . . . 206
A.3.3 Experimental Parameters . . . 233
A.4 Results . . . 233
A.5 Conclusions . . . 238

B Complete Discretization Results (N=3) 240

C Complete Discretization Results (N=5) 255

D Semantic Similarity using the Gene Ontology **270**
D.1 Semantic Similarity 271
D.2 Question 1: GO Similarity Distribution for Interactions 272
D.3 Question 2: Interactions for GO Similarity Ranges 275
D.4 Conclusions 279

E Individual Results for ALARM Bayesian Network Learning **281**

F Individual Results for IDEKER50 YEAST Bayesian Network Learning **287**

G Visualization of MG122 MOUSE Differential Expression Timeseries **336**

Bibliography **349**

List of Tables

6.1 Distribution of number of edges per expert for ALARM 61
6.2 Distribution of interactions per Explicit Data Source for YEAST 65
6.3 Distribution of interactions per Explicit Expert for YEAST 66
6.4 Distribution of coverage per Implicit Expert for YEAST 66
6.5 Expert Reliability Assignments for YEAST 70
6.6 Expert Reliability Assignments for YEAST (continued) 71
6.7 Data Sources and Distributions for Explicit and Implict Experts for MOUSE 85
6.8 Expert Reliability Assignments $\mathcal{R}_e$ for MG122 MOUSE 86
6.9 Expert Reliability Assignments $\mathcal{R}_e$ for MG449 MOUSE 87

A.1 Comparison of approximations to IDRFull (N=3) by (average) percentage of variables discretized exactly as by IDRFull. 238
A.2 Comparison of approximations to PDRFull (N=3) by (average) percentage of variables discretized exactly as by PDRFull. 239
A.3 Comparison of approximations to SIDRFull (N=3) by (average) percentage of variables discretized exactly as by SIDRFull. 239
A.4 Comparison of approximations to SPDRFull (N=3) by (average) percentage of variables discretized exactly as by SPDRFull. 239

List of Figures

1.1 Example Gene Interaction Network from Ideker *et al.* [ITR+01] 2

2.1 Representation of Types of Experts 11
2.2 Gene Expression Microarrays 19
2.3 Expression data for Cell Cycle Phase, taken from Spellman *et al.* [SSZ+98] 20

3.1 Example Bayesian Network 26
3.2 General Bayesian Network Learning Algorithm 27
3.3 Noisy-OR Structural Decomposition 33
3.4 Updated Bayesian Network Learning Framework 34

5.1 Complete Bayesian Network Learning Framework 42
5.2 Calculating LinOP as matrix addition 45
5.3 MAGIC: Predicting Functional Relationship by a Bayesian Network, from Troyanskaya *et al.* [TDO+03] 47
5.4 BAYES: Predicting Gene Relationship by a Bayesian Network 48
5.5 (a) PRM schema for WebKB domain; (b) Fragment of unrolled network for WebKB model (taken from Getoor *et al.* [GSTK01]) 52
5.6 PRM schema for Gene Semantic Relationships 54
5.7 PRM schema for Gene Semantic Relationships (with Noisy-OR) 56
5.8 Unrolled Model for Gene Semantic Relationships 57

6.1 Graph Structure of the ALARM network [BSCC89] 60
6.2 Adjacency Matrices for ALARM experts 62
6.3 Consensus Likelihood Matrices for ALARM 63
6.4 Excerpt of the Gene Ontology (taken from [WABD04]) 67
6.5 Comparing coverage and **CONS** reliability in YEAST 69
6.6 YEAST IDEKER Gene Sets 72
6.7 LinOP and NoisyOR Consensus Likelihood Matrices for YEAST 74
6.8 Example of correspondence of consensus likelihood to functional categories in IDEKER331 YEAST (using MAIR NoisyOR). 75

6.9 BAYES Model for Computing Consensus Likelihood in YEAST 76
6.10 BAYES Consensus Likelihood Matrices for YEAST 77
6.11 PRM schema for YEAST 79
6.12 PRM Consensus Likelihood Matrices for YEAST 80
6.13 Distribution of Probability for PRM matrices on IDEKER331 YEAST dataset . . . 81
6.14 MAGIC and STRING Consensus Likelihood Matrices for YEAST 83
6.15 BAYES Model for Computing Consensus Likelihood in MOUSE 88
6.16 Example of correspondence of consensus likelihood to functional categories in MG122 MOUSE dataset (using BAYES3R) 89
6.17 All Consensus Likelihood Matrices for MG122 MOUSE dataset 91
6.18 All Consensus Likelihood Matrices for MG449 MOUSE dataset 92
6.19 Example of correspondence of consensus likelihood to functional categories in MG449 MOUSE dataset (using BAYES3R) 93

7.1 Complete Bayesian Network Learning Framework 95
7.2 Example Illustration of Individual Evaluation Criteria 98
7.3 Example Illustration of Relative Evaluation Criteria 102
7.4 Example Illustration of Relative Evaluation Criteria, continued 103
7.5 Example of Individual Criterion, using ALARM 0.0.2 NoisyOR 105
7.6 Using 5 versus 100 models for ALARM 106
7.7 ALARM 2.1.0 LinOP versus NoisyOR 107
7.8 ALARM Criterion 1 (Score Components for Starting and Ending Models) for All Methods 108
7.9 ALARM Criterion 2 (Average Log-Likelihood Per Iteration) for All Methods 109
7.10 ALARM Criterion 3 (Distribution of Number of Search Iterations) for All Methods . 110
7.11 ALARM Boxplots of Log-Likelihood of Starting Models 112
7.12 ALARM Average Log-Likelihood for First 10 Iterations 112
7.13 ALARM Significance of Difference between Log-Likelihood Distributions, per Iteration 113
7.14 ALARM Difference in Average Log-Likelihood per Iteration 113
7.15 ALARM Difference in Average Log-Likelihood, Averaged over Iteration 114
7.16 IDEKER50 STD $N' = 1$ Criterion1: Score Components 120
7.17 IDEKER50 STD $N' = 10$ Criterion1: Score Components 120
7.18 IDEKER50 MAG1–IDRR $N' = 1$ Criterion1: Score Components 121
7.19 IDEKER50 MAG1–IDRR $N' = 10$ Criterion1: Score Components 121
7.20 IDEKER50 STD $N' = 1$ Criterion2: Average Log-Likelihood per Iteration 122
7.21 IDEKER50 STD $N' = 10$ Criterion2: Average Log-Likelihood per Iteration 122
7.22 IDEKER50 MAG1–IDRR $N' = 1$ Criterion2: Average Log-Likelihood per Iteration . 123
7.23 IDEKER50 MAG1–IDRR $N' = 10$ Criterion2: Average Log-Likelihood per Iteration 123
7.24 IDEKER50 STD $N' = 1$ Criterion3: Density Plot of Number of Iterations 124
7.25 IDEKER50 STD $N' = 10$ Criterion3: Density Plot of Number of Iterations 124

7.26 IDEKER50 MAG1–IDRR $N' = 1$ Criterion3: Density Plot of Number of Iterations . 125
7.27 IDEKER50 MAG1–IDRR $N' = 10$ Criterion3: Density Plot of Number of Iterations 125
7.28 IDEKER50 Boxplots of Log-Likelihood of Starting Models 129
7.29 IDEKER50 STD Average Log-Likelihood for First 10 Iterations 130
7.30 IDEKER50 MAG1–IDRR Average Log-Likelihood for First 10 Iterations 131
7.31 IDEKER50 STD Significance of Difference between Log-Likelihood Distributions, per Iteration 132
7.32 IDEKER50 MAG1–IDRR Significance of Difference between Log-Likelihood Distributions, per Iteration 133
7.33 IDEKER50 STD Difference in Average Log-Likelihood per Iteration 134
7.34 IDEKER50 MAG1–IDRR Difference in Average Log-Likelihood per Iteration 135
7.35 IDEKER50 Difference in Average Log-Likelihood, Averaged over Iteration 136

8.1 Typical Gene Expression Data Analysis Pipeline 138
8.2 Disadvantages of Typical Analysis Pipeline 140
8.3 Example Gene Interaction Network from Ideker *et al.* [ITR+01] 141
8.4 Example Network Visualization using Ingenuity Pathways Analysis (Ingenuity©Systems) 144
8.5 Example Network Visualization using STRING Database 145
8.6 Differential Metabolic Gene Expression in Mouse Mammary Tissue between Pregnancy Day 17 and Lactation Day 2 (image courtesy of Michael Rudolph) 147
8.7 MG122 Gene Set for MOUSE 149
8.8 Visual Displays of MG122 Gene Set for MOUSE 150
8.9 Visualization Examples using Force-Directed Placement 151
8.10 Visualization of Differential Expression for MG122 MOUSE Dataset 153
8.11 Visualization of Differential Expression Timeseries for MG122 154
8.12 Visualization of Differential Expression Timeseries for MG122 Subgraph 155
8.13 Network for MG449 formed from **CONS** NoisyOR ($Pr(e_{ij}) > 0.60$) with bi-connected components colored by shared GenMAPP Pathways [DSV+02] (shown using Cytoscape with Organic Layout [SMO+03]) 157

A.1 Dependency between two features 165
A.2 Dependency among three features 165
A.3 Illustration of Binarization 166
A.4 Illustration of Discretization Level Coalescence 167
A.5 Structure Projection 174
A.6 Discretization results in non-linear mapping 177
A.7 Correlation of Discrete to Real 177
A.8 Example of the IDR objective function landscape 179
A.9 Examples of the IDR objective function landscape (39 samples) 180
A.10 Examples of the IDR objective function landscape (90 samples) 181

A.11 IDR objective function for 4-bin examples. 182
A.12 IDR objective function for extreme example of local maxima 182
A.13 Trace of Random Restarts for IDRRestart 183
A.14 Example where IDRRestart fails to find maximum 184
A.15 Enlarged view of where IDRRestart fails to find maximum 184
A.16 Example for IDRSplit 186
A.17 Nine examples of the PDR objective function landscape 190
A.18 Three dimensional views of PDR landscapes from Figure A.17. 191
A.19 PDRRestart finds the global maximum 192
A.20 PDRRestart fails to find the global maximum 193
A.21 PDRRestart fails to find the global maximum 194
A.22 Examples of wavelet functions: the Haar (Dau1), the Daubechies 4 (Dau4) and the Coiflet 4 (Coi4). 199
A.23 PCA projections of Two-dimensional Data. 201
A.24 Example of applying PCAD. 202
A.25 Correlation Histograms (CorrHist) 204
A.26 Hierarchical Clustering 206
A.27 Hierarchical Tree Comparisons (TreeComp) 207
A.28 PCA Projection of the CF dataset 209
A.29 PCA Projection of the five CF subsets 210
A.30 PCA Projection of the CFRatio dataset 211
A.31 PCA Projection of the five CFRatio subsets 212
A.32 PCA Projection of the ADENO dataset 214
A.33 PCA Projection of the ADENORatio dataset 215
A.34 PCA Projection of the five ADENO subsets 216
A.35 PCA Projection of the five ADENORatio subsets 217
A.36 PCA Projection of the SPELLMAN dataset 218
A.37 PCA Projection of the five SPELLMAN subsets 219
A.38 PCA Projection of the YVERT464 dataset 220
A.39 PCA Projection of the five YVERT464 subsets 221
A.40 PCA Projection of the YVERT465 dataset 222
A.41 PCA Projection of the five YVERT465 subsets 223
A.42 PCA Projection of the GASCH dataset 224
A.43 PCA Projection of the five GASCH subsets 225
A.44 PCA Projection of the HUGHES dataset 226
A.45 PCA Projection of the five HUGHES subsets 227
A.46 CG1 Bayesian Network 228
A.47 PCA Projection of the CG1 dataset 228
A.48 ALARM Bayesian Network 229

A.49 PCA Projection of the ALARM dataset 229
A.50 PCA Projection of the SEGMENT dataset 230
A.51 PCA Projection of the IONO dataset 231
A.52 PCA Projection of the SONAR dataset 232
A.53 Example of Evaluation Metrics using YVERT464 Dataset 234

B.1 Metrics on the CF Dataset (N=3) 241
B.2 Metrics on the CFRatio Dataset (N=3) 242
B.3 Metrics on the ADENO Dataset (N=3) 243
B.4 Metrics on the ADENORatio Dataset (N=3) 244
B.5 Metrics on the SPELLMAN Dataset (N=3) 245
B.6 Metrics on the YVERT464 Dataset (N=3) 246
B.7 Metrics on the YVERT465 Dataset (N=3) 247
B.8 Metrics on the GASCH Dataset (N=3) 248
B.9 Metrics on the HUGHES Dataset (N=3) 249
B.10 Metrics on the CG1 Dataset (N=3) 250
B.11 Metrics on the ALARM Dataset (N=3) 251
B.12 Metrics on the IONO Dataset (N=3) 252
B.13 Metrics on the SEGMENT Dataset (N=3) 253
B.14 Metrics on the SONAR Dataset (N=3) 254

C.1 Metrics on the CF Dataset (N=5) 256
C.2 Metrics on the CFRatio Dataset (N=5) 257
C.3 Metrics on the ADENO Dataset (N=5) 258
C.4 Metrics on the ADENORatio Dataset (N=5) 259
C.5 Metrics on the SPELLMAN Dataset (N=5) 260
C.6 Metrics on the YVERT464 Dataset (N=5) 261
C.7 Metrics on the YVERT465 Dataset (N=5) 262
C.8 Metrics on the GASCH Dataset (N=5) 263
C.9 Metrics on the HUGHES Dataset (N=5) 264
C.10 Metrics on the CG1 Dataset (N=5) 265
C.11 Metrics on the ALARM Dataset (N=5) 266
C.12 Metrics on the IONO Dataset (N=5) 267
C.13 Metrics on the SEGMENT Dataset (N=5) 268
C.14 Metrics on the SONAR Dataset (N=5) 269

D.1 Excerpt of the Gene Ontology (taken from [WABD04]) 271
D.2 Examples of GO Semantic Similarity Distributions 273
D.3 Examples Demonstrating Degrees of Separation for GO Semantic Similarity Distributions 274

D.4 Summary of Separation of GO Semantic Similarity Distributions per Expert 276
D.5 Average Similarity from Random Samples of Pairs versus Pairs with Evidence from Any Interaction Source 277
D.6 Number of Pieces of Positive Evidence versus GO Similarity Score: Cellular Component 278
D.7 Number of Pieces of Positive Evidence versus GO Similarity Score: Molecular Function 278
D.8 Number of Pieces of Positive Evidence versus GO Similarity Score: Biological Process 279
D.9 GO Similarity versus MIPS Catalogues 280

E.1 ALARM 0.0.2 LinOP 282
E.2 ALARM 0.0.2 NoisyOR 282
E.3 ALARM 0.1.2 LinOP 283
E.4 ALARM 0.1.2 NoisyOR 283
E.5 ALARM 0.2.0 LinOP 284
E.6 ALARM 0.2.0 NoisyOR 284
E.7 ALARM 1.1.1 LinOP 285
E.8 ALARM 1.1.1 NoisyOR 285
E.9 ALARM 2.1.0 LinOP 286
E.10 ALARM 2.1.0 NoisyOR 286

F.1 IDEKER50 STD $N' = 1$ STRING 288
F.2 IDEKER50 STD $N' = 10$ STRING 288
F.3 IDEKER50 MAG1-IDRR $N' = 1$ STRING 289
F.4 IDEKER50 MAG1-IDRR $N' = 10$ STRING 289
F.5 IDEKER50 STD $N' = 1$ MAGIC 290
F.6 IDEKER50 STD $N' = 10$ MAGIC 290
F.7 IDEKER50 MAG1-IDRR $N' = 1$ MAGIC 291
F.8 IDEKER50 MAG1-IDRR $N' = 10$ MAGIC 291
F.9 IDEKER50 STD $N' = 1$ Mair LinOP 292
F.10 IDEKER50 STD $N' = 10$ Mair LinOP 292
F.11 IDEKER50 MAG1-IDRR $N' = 1$ Mair LinOP 293
F.12 IDEKER50 MAG1-IDRR $N' = 10$ Mair LinOP 293
F.13 IDEKER50 STD $N' = 1$ Cons LinOP 294
F.14 IDEKER50 STD $N' = 10$ Cons LinOP 294
F.15 IDEKER50 MAG1-IDRR $N' = 1$ Cons LinOP 295
F.16 IDEKER50 MAG1-IDRR $N' = 10$ Cons LinOP 295
F.17 IDEKER50 STD $N' = 1$ Unif5 LinOP 296
F.18 IDEKER50 STD $N' = 10$ Unif5 LinOP 296
F.19 IDEKER50 MAG1-IDRR $N' = 1$ Unif5 LinOP 297
F.20 IDEKER50 MAG1-IDRR $N' = 10$ Unif5 LinOP 297
F.21 IDEKER50 STD $N' = 1$ Rand1 LinOP 298

F.22 IDEKER50 STD $N' = 10$ Rand1 LinOP 298
F.23 IDEKER50 MAG1-IDRR $N' = 1$ Rand1 LinOP 299
F.24 IDEKER50 MAG1-IDRR $N' = 10$ Rand1 LinOP 299
F.25 IDEKER50 STD $N' = 1$ Rand2 LinOP 300
F.26 IDEKER50 STD $N' = 10$ Rand2 LinOP 300
F.27 IDEKER50 MAG1-IDRR $N' = 1$ Rand2 LinOP 301
F.28 IDEKER50 MAG1-IDRR $N' = 10$ Rand2 LinOP 301
F.29 IDEKER50 STD $N' = 1$ Mair NoisyOR 302
F.30 IDEKER50 STD $N' = 10$ Mair NoisyOR 302
F.31 IDEKER50 MAG1-IDRR $N' = 1$ Mair NoisyOR 303
F.32 IDEKER50 MAG1-IDRR $N' = 10$ Mair NoisyOR 303
F.33 IDEKER50 STD $N' = 1$ Cons NoisyOR 304
F.34 IDEKER50 STD $N' = 10$ Cons NoisyOR 304
F.35 IDEKER50 MAG1-IDRR $N' = 1$ Cons NoisyOR 305
F.36 IDEKER50 MAG1-IDRR $N' = 10$ Cons NoisyOR 305
F.37 IDEKER50 STD $N' = 1$ Unif5 NoisyOR 306
F.38 IDEKER50 STD $N' = 10$ Unif5 NoisyOR 306
F.39 IDEKER50 MAG1-IDRR $N' = 1$ Unif5 NoisyOR 307
F.40 IDEKER50 MAG1-IDRR $N' = 10$ Unif5 NoisyOR 307
F.41 IDEKER50 STD $N' = 1$ Rand1 NoisyOR 308
F.42 IDEKER50 STD $N' = 10$ Rand1 NoisyOR 308
F.43 IDEKER50 MAG1-IDRR $N' = 1$ Rand1 NoisyOR 309
F.44 IDEKER50 MAG1-IDRR $N' = 10$ Rand1 NoisyOR 309
F.45 IDEKER50 STD $N' = 1$ Rand2 NoisyOR 310
F.46 IDEKER50 STD $N' = 10$ Rand2 NoisyOR 310
F.47 IDEKER50 MAG1-IDRR $N' = 1$ Rand2 NoisyOR 311
F.48 IDEKER50 MAG1-IDRR $N' = 10$ Rand2 NoisyOR 311
F.49 IDEKER50 STD $N' = 1$ BAYESUN 312
F.50 IDEKER50 STD $N' = 10$ BAYESUN 312
F.51 IDEKER50 MAG1-IDRR $N' = 1$ BAYESUN 313
F.52 IDEKER50 MAG1-IDRR $N' = 10$ BAYESUN 313
F.53 IDEKER50 STD $N' = 1$ BAYES0N 314
F.54 IDEKER50 STD $N' = 10$ BAYES0N 314
F.55 IDEKER50 MAG1-IDRR $N' = 1$ BAYES0N 315
F.56 IDEKER50 MAG1-IDRR $N' = 10$ BAYES0N 315
F.57 IDEKER50 STD $N' = 1$ BAYES3N 316
F.58 IDEKER50 STD $N' = 10$ BAYES3N 316
F.59 IDEKER50 MAG1-IDRR $N' = 1$ BAYES3N 317
F.60 IDEKER50 MAG1-IDRR $N' = 10$ BAYES3N 317

F.61 IDEKER50 STD $N' = 1$ BAYESUR 318
F.62 IDEKER50 STD $N' = 10$ BAYESUR 318
F.63 IDEKER50 MAG1-IDRR $N' = 1$ BAYESUR 319
F.64 IDEKER50 MAG1-IDRR $N' = 10$ BAYESUR 319
F.65 IDEKER50 STD $N' = 1$ BAYES0R 320
F.66 IDEKER50 STD $N' = 10$ BAYES0R 320
F.67 IDEKER50 MAG1-IDRR $N' = 1$ BAYES0R 321
F.68 IDEKER50 MAG1-IDRR $N' = 10$ BAYES0R 321
F.69 IDEKER50 STD $N' = 1$ BAYES3R 322
F.70 IDEKER50 STD $N' = 10$ BAYES3R 322
F.71 IDEKER50 MAG1-IDRR $N' = 1$ BAYES3R 323
F.72 IDEKER50 MAG1-IDRR $N' = 10$ BAYES3R 323
F.73 IDEKER50 STD $N' = 1$ PRMUN 324
F.74 IDEKER50 STD $N' = 10$ PRMUN 324
F.75 IDEKER50 MAG1-IDRR $N' = 1$ PRMUN 325
F.76 IDEKER50 MAG1-IDRR $N' = 10$ PRMUN 325
F.77 IDEKER50 STD $N' = 1$ PRM0N 326
F.78 IDEKER50 STD $N' = 10$ PRM0N 326
F.79 IDEKER50 MAG1-IDRR $N' = 1$ PRM0N 327
F.80 IDEKER50 MAG1-IDRR $N' = 10$ PRM0N 327
F.81 IDEKER50 STD $N' = 1$ PRM3N 328
F.82 IDEKER50 STD $N' = 10$ PRM3N 328
F.83 IDEKER50 MAG1-IDRR $N' = 1$ PRM3N 329
F.84 IDEKER50 MAG1-IDRR $N' = 10$ PRM3N 329
F.85 IDEKER50 STD $N' = 1$ PRMUR 330
F.86 IDEKER50 STD $N' = 10$ PRMUR 330
F.87 IDEKER50 MAG1-IDRR $N' = 1$ PRMUR 331
F.88 IDEKER50 MAG1-IDRR $N' = 10$ PRMUR 331
F.89 IDEKER50 STD $N' = 1$ PRM0R 332
F.90 IDEKER50 STD $N' = 10$ PRM0R 332
F.91 IDEKER50 MAG1-IDRR $N' = 1$ PRM0R 333
F.92 IDEKER50 MAG1-IDRR $N' = 10$ PRM0R 333
F.93 IDEKER50 STD $N' = 1$ PRM3R 334
F.94 IDEKER50 STD $N' = 10$ PRM3R 334
F.95 IDEKER50 MAG1-IDRR $N' = 1$ PRM3R 335
F.96 IDEKER50 MAG1-IDRR $N' = 10$ PRM3R 335

G.1 Differential Expression Visualization for MG122 MOUSE Gene Set 337
G.2 Differential Expression Timepoint Virgin Mouse for MG122 MOUSE Gene Set . . . 337
G.3 Differential Expression Timepoint Pregnancy Day 1 for MG122 MOUSE Gene Set . 338

G.4 Differential Expression Timepoint Pregnancy Day 3 for MG122 MOUSE Gene Set . 338
G.5 Differential Expression Timepoint Pregnancy Day 7 for MG122 MOUSE Gene Set . 339
G.6 Differential Expression Timepoint Pregnancy Day 12 for MG122 MOUSE Gene Set . 339
G.7 Differential Expression Timepoint Pregnancy Day 17 for MG122 MOUSE Gene Set . 340
G.8 Differential Expression Timepoint Pregnancy Day 19 for MG122 MOUSE Gene Set . 340
G.9 Differential Expression Timepoint Lactation Day 1 for MG122 MOUSE Gene Set . . 341
G.10 Differential Expression Timepoint Lactation Day 2 for MG122 MOUSE Gene Set . . 341
G.11 Differential Expression Timepoint Lactation Day 9 for MG122 MOUSE Gene Set . . 342
G.12 Differential Expression Timepoint Involution Day 2 for MG122 MOUSE Gene Set . 342
G.13 Differential Expression Visualization for MG122 MOUSE (subgraph) 343
G.14 Differential Expression Timepoint Virgin Mouse for MG122 MOUSE (subgraph) . . 343
G.15 Differential Expression Timepoint Pregnancy Day 1 for MG122 MOUSE (subgraph) 344
G.16 Differential Expression Timepoint Pregnancy Day 3 for MG122 MOUSE (subgraph) 344
G.17 Differential Expression Timepoint Pregnancy Day 7 for MG122 MOUSE (subgraph) 345
G.18 Differential Expression Timepoint Pregnancy Day 12 for MG122 MOUSE (subgraph) 345
G.19 Differential Expression Timepoint Pregnancy Day 17 for MG122 MOUSE (subgraph) 346
G.20 Differential Expression Timepoint Pregnancy Day 19 for MG122 MOUSE (subgraph) 346
G.21 Differential Expression Timepoint Lactation Day 1 for MG122 MOUSE (subgraph) . 347
G.22 Differential Expression Timepoint Lactation Day 2 for MG122 MOUSE (subgraph) . 347
G.23 Differential Expression Timepoint Lactation Day 9 for MG122 MOUSE (subgraph) . 348
G.24 Differential Expression Timepoint Involution Day 2 for MG122 MOUSE (subgraph) 348

Chapter 1

Introduction

The main goal of this thesis is to investigate ways in which a collection of separate prior information sources, of varying scale (low- or high-throughput), varying reliability (collected by trusted experts or not) and varying density (knowledge concentrated in disjoint areas of the problem) can be integrated in a principled manner to reveal a more complete understanding of a problem domain. We consider a case study from the biological domain: inferring a network of gene interactions.

Every cell of an organism contains a set of instructions for manufacturing all molecules necessary for the structure and function of the cell. This instruction set is organized into hereditary units of information called *genes*. A gene is a segment of *DNA* found on a *chromosome* in the cell and information contained in each gene directs the cell to produce a particular *protein* specific to that gene. Proteins are created through a series of processes which generate a number of intermediate gene products. The most important product is *messenger RNA* (mRNA), which transfers the gene information from the DNA to areas of the cell responsible for the production of the corresponding protein. Each protein has a specific set of functions, such as providing structural components of the cell and performing of all cellular activities. To carry out cellular functions, proteins can physically organize into complexes or coordinate indirectly through other molecules.

To understand the inner workings of a cell, biologists must therefore understand how all the various gene products interact in order to orchestrate life in the cell. A motivating example of the type of gene interaction network we wish to learn is given in Figure 1.1, taken from Ideker *et al.* [ITR+01]. Genes are represented as nodes in the graph. Information about the individual genes can be superimposed as visual attributes on the nodes. In this example, the size and color of the nodes indicate the relative level of activity measured for each gene, *i.e.,* the *gene expression level*, under a given experimental biological condition versus a control condition. Large white nodes depict decreased activity in response to the stimulus while large black nodes indicated increased activity. Arcs between the nodes capture known relationships between the corresponding genes. Yellow undirected arcs between two genes indicate that the corresponding genes bind physically while blue directed arcs signify that one gene regulates the expression level of another gene. Note that these relationships are actually defined on different gene products: physical binding occurs

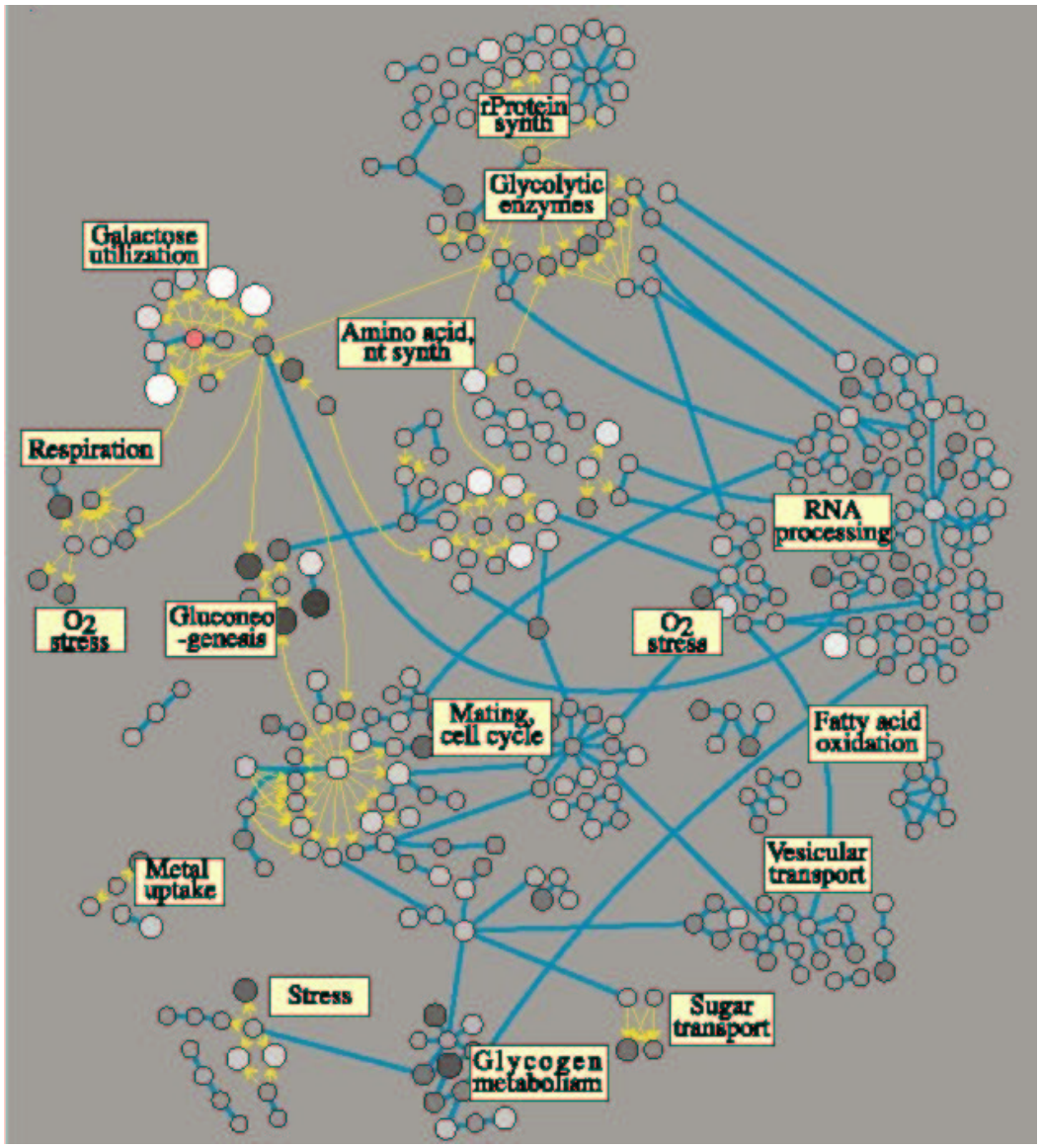


Figure 1.1: Example Gene Interaction Network from Ideker *et al.* [ITR+01]

.

between two proteins and regulation occurs when the protein defined by one gene binds to the DNA encoding the second gene. Though technically a gene is a segment of DNA, for our purposes we do not make a distinction between the various gene products since we are ultimately interested in the network of relationships among any intermediate product associated with the genes. We use the term gene to encompasses any of those intermediates.

The ability to create such a network presents a critical advantage for biologists since the graph allows an enormous amount of information to be viewed simultaneously and within a global context. For example, Figure 1.1 shows changes in gene activity in an organism whose GAL4 gene (shown as a red circle in the figure) has been deleted. Viewing the consequences of the deletion in the context of related genes can help provide an immediate global explanation for the observed data. Inspired by this example, our objective in this thesis is to develop a strategy of incorporating other indicators of potential relationships among genes, not just binding or regulation, to first suggest a potential network based on the prior information about interaction, and to then use the measured expression levels between genes to infer novel relationships and further refine the model. We hope that creating networks as such will provide a tool with which biologists can further explore, and perhaps eventually understand, the complexity of the relationships among the genes they study.

Biology has a rich and diverse body of knowledge already available upon which to leverage the task of learning such a model; the challenge is in developing meaningful ways to encode such prior knowledge which take into account the heterogeneity of the individual prior data sources. In this thesis, we demonstrate a number of strategies for combining prior information to predict whether two genes interact or are related in some other more general semantic context. We show that such a *consensus likelihood of interaction* can then be used in two distinct ways. By itself, the consensus likelihood can provide high confidence working models of interaction, which can be incorporated into visualization tools for the biologist. In addition, the consensus likelihood can be used as prior estimates of interaction when learning relationships from yet another type of data: *gene expression data.*

To understand cellular function, we are ultimately interested in genes at the protein level. Though data about *interaction* between proteins is becoming increasingly available, several factors complicate the ability to monitor *activity* at the level of proteins. Fortunately, the intermediate product mRNA is much more amenable to investigation, providing a surrogate measurement for protein. Since an mRNA must be present in order for the cell to produce the corresponding protein, the respective activity levels are roughly correlated. Measurements of the amount of mRNA produced by a cell is precisely the *gene expression level* mentioned earlier. Since the regulation of gene expression is directly controlled by proteins, a popular topic is the inference of gene regulation networks from the expression data. Note that such a network is an example of the more general gene interaction network we propose to study yet we wish to make a careful distinction here. When we speak of an *interaction* network or a network of gene *relationships*, we refer to one inferred from prior sources suggesting potential interactions or relationships, while when speaking of a *regulatory* network, we mean one inferred from gene expression data.

To infer a regulatory network, a modelling formalism for representing genes and the relationships between activity levels must first be selected. A popular choice is to use Bayesian networks [Pea88]. As in the example above, Bayesian networks represent also gene expression levels as nodes in a graph and arcs between nodes capture relationships between genes (*i.e.,* between their activity levels) although unlike the example, the graph is restricted to be a directed acyclic graph. Bayesian networks also quantify the strength of the relationship by specifying conditional probability distributions of the expression value of each node given the value of its parents in the graph.

Bayesian networks have received increased attention in computational biology as a modelling technique for capturing dependencies between genetic entities [BF01, HGJY01, PREF01, HGJY02, IGM02, TKB+03, JYG+03, TDO+03]. Their probabilistic basis makes them ideal for capturing both the underlying stochasticity of the interactions between genes as well as any noise or uncertainty involved in obtaining measurements of gene expression. Moreover, Bayesian networks are a compact and computable representation which can be learned from data and then queried efficiently. A further attraction for their use is the ability to naturally incorporate prior information during learning, ideal for our biological application where a large body of prior knowledge exists.

Previous efforts for learning Bayesian networks from biological data, however, have concentrated on learning the networks from expression data *de novo* and prior biological knowledge is mostly used for validation or as additional attributes [BF01, PREF01, IGM02, TKB+03]. Though evaluating the ability to learn previously known groups of regulatory interactions between genes is a noble task, the novelty of this thesis is that instead of using known interactions as validation or additional attributes, we investigate the effect of using existing knowledge about interactions as a *true* source of prior bias during learning.

The use of prior information in this particular application is especially desirable since most previous uses of Bayesian networks in domains other than biology have involved learning models over a few variables from a reasonable number of samples. For example, the popular real-world benchmark model of ICU ventilator management (ALARM) has 37 variables and typically uses thousands or tens of thousands of samples during learning [CH92, HGC95]. In our application of gene regulatory networks, there are typically thousands of genes and the cost of obtaining samples is high, resulting in an undesirable ratio of variables to samples. Prior information about which models are more plausible thus becomes critical in this domain, since as noted in [MCM00], without incorporating prior knowledge, the task of learning even a fragment of the large and complex biological network we hope to understand will be almost impossible.

Our first major contribution of the thesis is **to demonstrate how using a prior distribution over *interactions* between genes can significantly increase the speed and quality of search for high scoring Bayesian Networks when learning from *gene expression data*.** Though it may perhaps seem obvious that adding prior information will accelerate learning, it is not clear in this domain whether the types of information that are available for formulating priors are going to be helpful. Some information exists indicating which proteins are known to affect the regulation of which other genes, but our data for learning does not measure proteins: we measure

levels of mRNA where any number of additional factors contribute to the actual levels observed. The intuition is that knowledge about gene interaction might help further explain the data, whether that information is about regulation, physical binding, DNA or protein sequence features, and the like. In this thesis, we formally investigate whether prior knowledge of interaction at all different levels of indirection and gene product type will be informative in uncovering relationships contained in the gene expression data.

In order to provide meaningful priors to the Bayesian network learning algorithm, we need methods to integrate the various prior sources of gene interaction information, namely to formulate the consensus likelihood mentioned earlier. Any combination strategy must remain sensitive to the varying reliability, scope, and density of the information available by our various sources and it is precisely the characteristics of the data available in the biological domain that makes this problem both novel and interesting.

Previous work describing the use of prior information during Bayesian network learning in other domains has primarily focused on providing informative parameter priors, *i.e.,* quantifying the probabilistic dependencies between entities. However, the type of prior most abundant in our biological application is structural, *i.e.,* specification of arcs between entities, and there has been little research on incorporating structural priors in Bayesian networks. Typically, the use of structural priors is restricted to uninformed global constraints such as constraining the maximum number of parents for any node or introducing similar biases toward less connected models. In the few examples of using informed structural priors, the prior information is solicited from a single expert familiar with the entire problem domain. This is not the case in our biological application.

Prior information in our domain comes in the form of many experts and each 'expert' is in fact a database compiled by many individuals who performed experiments to measure the interaction between specific genes. The quality of the expert depends on the particular method of measuring the interaction. These methods vary in terms of reliability since some are performed in vitro and are thus highly prone to false positives and false negatives. Methods vary in terms of scope since some are capable of high-throughput application while others are inherently small scale techniques. Methods also vary in terms of density over the domain since there are some subsets of genes that are more well studied than others. For example, the p53 gene and its interactors are well studied as prime implicants in human cancers. The second major contribution of this thesis is in **developing techniques for combining prior information from a collection of incomplete and noisy data sources**.

Like the example in Figure 1.1 which considered regulation and binding information between genes, we consider a host of 'experts' which specify explicit or implicit interactions between genes, such as physical interaction between the genes, shared biological function or even co-location within a particular cellular compartment. Though each individual source may be a poor predictor for gene interaction when used alone, we demonstrate techniques for combining the individually less informative experts, thereby formulating a more reliable consensus likelihood for predicting whether any two genes are related.

The third major contribution of this thesis is **to demonstrate the use of the consensus likelihood in creating visualization tools for large scale datasets**. As in the example of Figure 1.1, additional information can be superimposed on a graphical structure to create a powerful and useful working model for investigating the dynamic interaction of entire groups of genes as experimental conditions vary. In the example, the genes are arranged spatially by general functional category, such as Galactose utilization or Metal uptake. Also, the size and color of the nodes depicts the magnitude of change in gene expression value between conditions, simultaneously for all nodes.

Together with the functional category assignment, viewing the expression changes allows a global snapshot of the response of the organism to the experimental condition. In this particular example, it becomes evident that deleting GAL4 shuts down the Galactose utilization pathway (large white nodes) and turns on the Gluconeogenesis and Glycogen metabolism pathways (large black nodes). Viewing the expression data in the context of the graph suggests an immediate explanation for the response of the system: the expression data changes are consistent with the knowledge that when faced with the inability to convert galactose to glucose (Galactose utilization), an organism will increase production of glucose from raw products (Gluconeogenesis) or obtain glucose from the storage form, glycogen (Glycogen metabolism). The ability to overlay information on a meaningfully spatially-organized graph facilitates the ability of biologists to view and understand an otherwise overwhelming amount of information. Our goal becomes to create such a graph.

Ideally we would like the graphical structure to be that resulting from Bayesian network learning, since during learning the original graph suggested by the combination of interaction information sources has been refined using the additional source of genome-wide expression data. However, the amount of expression data currently available prohibits learning networks for a large number of genes, a situation that will hopefully improve somewhat in the future. In the meantime, the consensus likelihood itself provides a valuable interim solution since it can specify highly probable graph structures over a much larger set of nodes. Also, by using the consensus likelihood to suggest plausible graphs, we can incorporate the probability of each arc during the layout or visualization process. For example, the length or the width of the arc between two nodes can be made proportional to the probability that two nodes interact, as specified by the consensus likelihood. We provide results using the consensus likelihood which demonstrate how a collection of weakly suggested, noisy, and incomplete relationships can be combined to create a powerful visualization tool for large scale datasets.

The remainder of the thesis is organized as follows. We first present the problem of inferring gene interaction networks in Chapter 2 and introduce the different types of data available in biology for this task. We include a list of the characteristics inherent in the data that make the problem of combining the sources so challenging, thereby motivating our work in this area. We then formally define the theory of Bayesian networks and present an overview of the general learning algorithms in Chapter 3 before detailing previous approaches to formulating structural priors in Chapter 4. Chapter 5 presents our four different approaches for computing a consensus belief over the probability of interaction between any pair of genes. Chapter 6 describes the application of our consensus belief

functions for examples in the mouse and yeast genome, as well as for a simulated application using a benchmark Bayesian network for ICU ventilator management (ALARM). The latter allows us to explore the limitations of our approaches in a well-studied model. Each of the four approaches can then be substituted as the structural prior component of the Bayesian network learning algorithm. Experiments using the consensus likelihood functions as priors are presented in Chapter 7. A further use of the structural prior is given in Chapter 8 where the probability over structures can be used to create a visualization tool for viewing large scale datasets in the context of the gene interaction network. Chapter 9 discusses the overall conclusions of the thesis and outlines future work.

Chapter 2

Molecular Biology and Biological Data Sources

To understand the motivation and intricacies of the challenges addressed in this thesis, it is necessary to first become familiar with the terminology and types of data available in the biological domain. The primary entity of interest is the *gene*, a unit of hereditary information, defined as a segment of *DNA*. The set of all genes of an organism is known as the *genome*. The DNA segment encoding a gene consists of a specific *sequence* of molecules that can be later interpreted by the cell to manufacture other molecules known as *mRNA* and *proteins*, specific to the gene from which they originate. With some abuse of interpretation, we use the term gene in this thesis to refer to any of the various gene products, becoming more specific when necessary.

In this chapter, we describe two distinct types of genetic networks: *interaction* networks and *regulatory* networks. We also detail the data available for learning each type of network. The last section of the chapter outlines how we might use an interaction network to provide prior knowledge when learning a regulatory network. The later chapters, Chapter 3 through Chapter 5, will then describe the specifics of that task.

2.1 Gene Interaction Networks

All activity within a living cell comes about through a series of coordinated processes, controlled and executed by specific gene products. In order to understand cellular mechanisms, it is therefore vital for biologists to understand how genes interact with each other as well as other cell components. Over the past few decades, numerous techniques have developed to investigate certain properties of genes and to interrogate relationships among them. Despite the growing volume of information, biologists are far from a complete understanding of the network of gene relationships, due in part to the sheer amount and diversity of facts, as well as the lack of proper integration of that information.

Findings in biology have been accumulating over the last half-century, slowly at first, then nearly

exponentially in the past decade. As a consequence, most existing knowledge comes in the form of free-text descriptions of genes and their interactions, requiring a enormous amount of effort for a biologist to assimilate the data and realize its implications. Even with the recent efforts by computer scientists to automate the process and build databases which allow the information to be more easily attained, too often the necessary details are scattered across distinct data sources, leaving the biologists again to collate the information.

The primary interest of this thesis is to combine the various sources in order to present the biologists with an integrated view of the global network of interactions, such as the example shown in Figure 1.1. However, a number of characteristics specific to the data available in this domain make such an integration a challenging task. Any attempt to combine prior knowledge in the domain must be sensitive to these characteristics. In particular, due to limitations in the ability to interrogate biological entities, the data vary in terms of *scale, reliability*, and *density* over the domain.

The scale of the data refers to whether a biological assay is performed for a small number of genes (*low-throughput*) or (nearly) genome-wide (*high-throughput*). Reliability is often related to scale since low-throughput methods are small scale precisely because their higher precision makes them more time consuming and costly. On the other hand, high-throughput techniques sometimes sacrifice complete biological accuracy for the sake of more complete coverage of the genome. In particular, most high-throughput studies are performed *in vitro*, where gene activity is observed outside the natural cellular environment. High-throughput techniques are prone to high false positive and false negative rates for this reason. Differences in reliability can also refer to the ability of an assay to predict gene relationships, in terms of whether the data source (however precisely measured) is a direct or indirect indicator of an interaction. For example, a direct interaction might indicate physical binding of two proteins whereas an indirect interaction might indicate that two proteins affect the same set of other proteins, or are involved in the same general biological process.

The different scales of the techniques also affect the density of study of genes over the whole biological domain. The collection of information from low-throughput studies creates a patchwork of dense areas of knowledge which must then be connected with the data from the high-throughput studies. Regardless of scale, the density of study may also suffer because particular experimental techniques can be biased toward collecting certain results. For example, most techniques are biased toward collecting information for large or very abundant proteins since they provide sufficient physical sample for the assay. In other cases, the particular three-dimensional structure a protein might not be amenable to study by most traditional techniques and as a result of the effort involved, the protein is rarely studied. Certain proteins and their interactors might also be more well studied than others simply because they have important agricultural, pharmaceutical or biomedical consequences. For example, diabetes research has motivated the investigation of genes involved in the glucose and insulin related pathways.

Aside from issues of scale, reliability and density, biological data has another, perhaps more important, characteristic: existing knowledge about gene interaction is almost exclusively given as positive assertions. One can not assume that the lack of information about a particular interaction

implies that the genes do not interact. It could simply be that a particular assay did not involve certain genes, as happens by definition with the low-throughput techniques which study only a handful of all possible pairwise interactions. Even if the interaction is investigated by an assay, the failure of the technique to detect an interaction does not mean there is no interaction. It could be that the particular conditions of the assay were not those under which the interaction occurs. Or physical limitations of the experimental technique caused a failure to detect the interaction. The lack of negative assertions has serious consequences for any method that attempts to learn to predict the existence of an interaction. We return to this point later in Chapter 5.

In this thesis, we develop four techniques for combining prior information sources about potential gene interactions. Vital to our approaches is their probabilistic basis, the key component which allows us to account for the unique characteristics of data in this domain. Probability theory can capture the underlying stochasticity of the interactions as well as compensate for incompleteness and noise in the data. To goal is to create a probabilistic consensus belief of the network of interactions as a function of the individual data sources. This consensus network can then be used in many ways, as we demonstrate in Chapter 7 and Chapter 8. In the following section, we detail the types of data sources available for our task.

2.2 Interaction Information Sources

A deluge of information has become available recently for a number of model organisms. The most well studied are *Saccharomyces cerevisiae* (baker's yeast) and *Mus musculus* (house mouse). These organisms are model organisms because yeast is a single-celled organism with a very fast reproductive cycle, making it ideal for repeated experimental manipulation, while the biology of the multi-cellular, mammalian mouse is very close to human biology. Both of these genomes, like humans, are *eukaryotic* in that their DNA resides in a distinct compartment of the cell called a *nucleus*. In contrast, *prokaryotic* organisms are uni-cellular and their DNA floats freely in the cell. Examples include the well-studied bacterial genomes, such as *Escheria coli* and *Haemophilus influenzae*. It is worthwhile to note that even in the most well studied organisms, many of the genes still have unknown function. In this thesis, we mostly consider the yeast and mouse genomes, though the techniques apply in any organism.

All previously known interactions used in this thesis are extracted from public databases or manually extracted from comprehensive listings found in the literature. Each assay or technique for measuring gene relationships becomes an *expert* in our domain, asserting that an interaction is detected. Rarely do these sources indicate with certainty that an interaction does *not* exist. Thus, we collect mostly positive assertions. A representation of the types of experts we consider is given in Figure 2.1.[1] The figure provides a global overview of the relationships among experts. In general, the various experts, such as Y2H or COPHASE, appear at the leaf nodes in the graph. We discuss the meaning of each of the nodes in the next two sections. In anticipation of their use within our

[1] This model will be discussed more fully in Section 5.3.

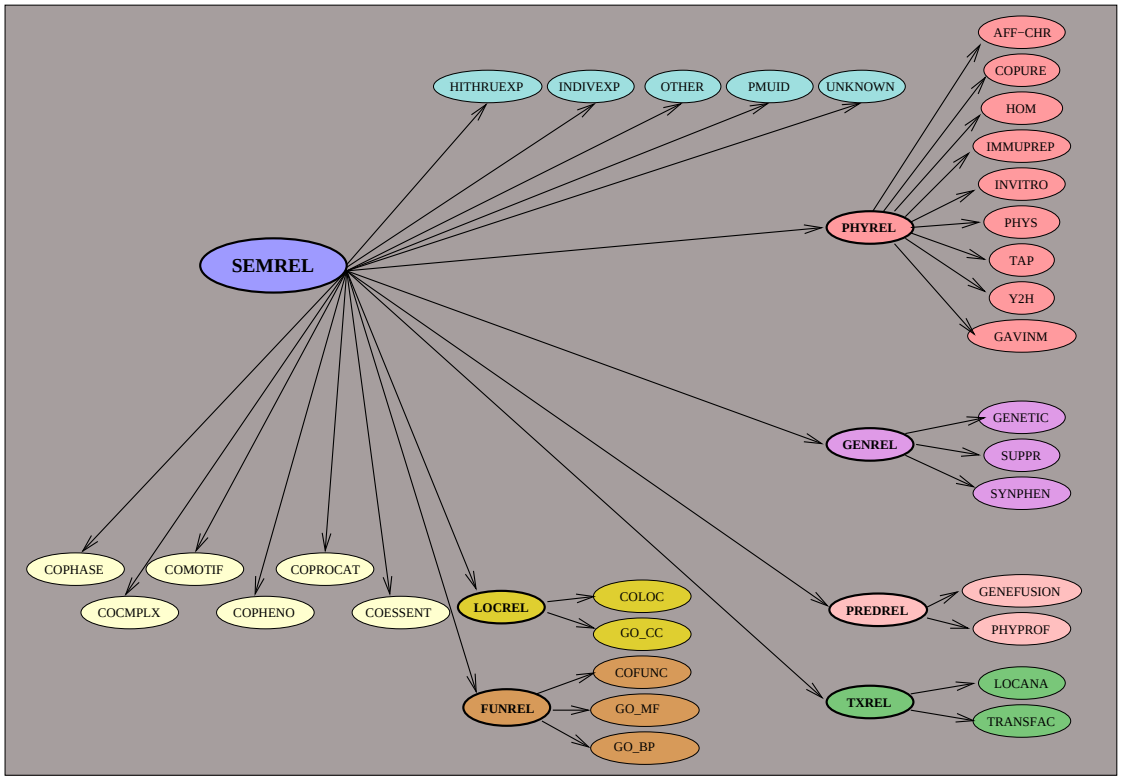


Figure 2.1: Representation of Types of Experts

consensus belief algorithms, we include an abbreviation (node label) after the name of each expert or expert relational type.

In our application, we are interested in capturing any type of relationship between genes, not placing importance of any one source over the other. Thus, we want to capture a generalized notion of relationship, which we term *semantic relationship* (SEMREL), to encapsulate the various types of interaction that can be measured. Two genes are semantically related if there is any evidence arguing that they should be considered together: if they share the same function, if they share the same cellular location, if they are shown to physically bind via an experimental assay, and so on.

The type of information available for indicating interaction between genes varies in many ways. The information can be gathered from biological experiments carefully designed to manipulate specific genes or can be predicted from DNA (or protein) sequence and evolutionary attributes. Also, the notion of interaction may be direct or indirect. For our purposes, it is most useful to distinguish the various sources as *explicit* or *implicit* interaction sources. Explicit interactions are those measured experimentally or predicted computationally for a given pair, while implicit interactions are those derived from individual features common to both genes in the pair. We describe each in detail in the sections below.

2.2.1 Explicit Interaction Sources

Various databases exist for cataloguing interactions explicitly measured among gene products in a number of model organisms. Interactions are typically listed for gene pairs, though occasionally the source contains larger complexes of genes. Public data sources, with potential overlap of information, include:

- **DIP** (Database of Interacting Proteins [XSD+02]), containing experimentally determined interactions between proteins curated both manually by domain expert curators and also automatically using computational approaches.
- **BIND** (Biomolecular Interaction Network Database [BDW+01]), containing full descriptions of interactions (for DNA, RNA or proteins), molecular complexes and pathways from peer-reviewed literature and from direct submissions.
- **BRITE** (Biomolecular Relations in Information Transmission and Expression Database for Biomolecular Relations [KKK97, KGH+06]), containing molecular interactions that form regulatory networks.
- **TRANSFAC** (TRANscription FACtor Database [WCH+00]), containing links between regulatory proteins and their DNA targets.
- **PREDICTOME** [MYC+02], containing predicted functional associations among genes or proteins using both experimental and computational techniques.
- **hp-DPI** (Helicobacter pylori database of protein interactomes [LCC+05]), containing experimentally verified and inferred interactions in the organism *H. pylori*.
- **BioGRID** (Biological General Repository for Interaction Datasets [SBR+06], containing physical and genetic interactions for many organisms.
- **HPRD** (Human Protein Reference Database [PNA+03]), containing hand-curated interactions for human proteins and their homologs in mice.
- **IntAct** (InterAction Database [HML+04], containing curated protein interactions.
- **MINT** (Molecular INTeraction database [ZMQ+02]), containing curated interactions, both pairwise and groupwise.
- **MIPS** (Munich Information Center for Protein Sequences [MFG+02]), containing of number of catalogues of manually curated physical and genetic interactions.
- **STRING**(Search Tool for the Retrieval of Interacting Genes/proteins [vMHJ+03, vMJS+05]), containing predicted functional associations between proteins based on comparative sequence analysis of over 175 genomes.

From these databases, we collect what will be referred to as *explicit* or *experimental* interactions. Our studies will use a subset of these resources, based on the organism of study, the comprehensiveness of the database, and the purpose of the information. The applicable databases will be listed by name in the appropriate sections of Chapter 6.

The databases contain data collected from a variety of techniques directly measuring physical relationships (PHYSREL), genetic relationships (GENREL), transcription factor relationships (TXREL), or computationally predicted relationships (PREDREL). Sometimes the database fails to list the technique (UNKNOWN) or includes a miscellaneous category (OTHER). Often the PubMed [2] literature reference is given (PMUID) or the scale of the assay is indicated (HITHRUEXP or INDIVEXP).

When indicating *physical interaction* (*i.e.,* physical binding between two proteins), sometimes the database will simply state that a interaction was measured without specifying the technique. In this case, we designate the expert by PHYS. In other cases, to measure physical interaction a technique can either be a low-throughput assay [for example, co-purification (COPURE) , affinity chromatography (AFFCHR) , immunoprecipitation (IMMUPREP) , invitro studies (INVITRO)], or a high-throughput method [for example, yeast two-hybrid (Y2H), tandem affinity precipitation with spoke representation (TAP) or matrix representation (GAVINM and HOM) [GBK$^+$02, HGH$^+$02]]. Of the latter, the tandem affinity precipitation method is used to purify an entire protein complex using a single tagged protein (*i.e.,* the bait). Since the exact configuration of physical interactions within this complex cannot be measured, there are two alternative methods for representing the data as pairs. A *spoke* representation conservatively pairs the bait with each other purified protein while the *matrix* representation generously pairs all proteins with all others in the same complex. There are many other assays for physical interaction which are not listed here. The reliability of the expert type depends on the scale (low or high-throughput) as well as level of indirection (spoke or matrix).

In contrast to physical binding studies which measure direct interactions, *genetic* relationships measure indirect influences of one gene on another. For example, observing the over-expression or suppression (SUPPR) of the level of activity for one gene by manipulating another may involve actions from intermediate genes. Also, simultaneously manipulating a pair of genes may induce a synthetic phenotype (SYNPHEN). For example *synthetic lethality* studies discover genetic relationships by uncovering pairs of genes where mutating both genes results in an inviable organism but mutation of either gene individually does not. Clearly the genes are related since they must both be present for survival but there is no other indication quantifying the degree of indirectness, or reliability, of the suggested relationship. The term GENETIC is used to indicate any other type of unspecified genetic interaction.

Transcriptional relationships occur between genes when the protein of one gene causes the cell to produce mRNA of another gene, a process known as *transcription.* The protein is referred to as the *transcription factor* which binds to a specific DNA sequence upstream from the target gene DNA sequence. Since the cell has many elaborate control mechanisms, presence of the upstream protein binding sequence does not guarantee that the gene is in fact a target of the regulatory protein. This

[2] `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?DB=pubmed`

affects the reliability of experts capturing transcriptional relationships. Putative upstream regions can be identified by examining whether a protein can physically bind to a location upstream of a gene (LOCANA) [LRR+02]. The known transcriptional relationships are listed in the TRANSFAC database (TRANSFAC).

Computationally predicted relationships are those where DNA sequence traits are examined genome-wide to suggest potential interactions, typically comparing the genomes from several different organisms. For example, phylogenetic profiles (PHYPROF) [MPT+99] indicate the simultaneous presence or absence of a group of genes across a host of genomes. Genes lost or gained as a group through time by evolution suggests they are biologically related. Gene fusion (GENEFUSION) [MPT+99] identifies groups of genes which appear individually in some genomes but occur as a single fused gene in others, arguing the intuition that evolutionary pressures cause the genome to compact related genes. Similarly, within a single organism, a group of related genes often appear in close proximity on the DNA sequence, sometimes with preserved gene order (GENENEIGH). As with putative transcriptional relationships, these computationally predicted relationships are very informative but often suffer from high false positive rates.

2.2.2 Implicit Interaction Sources

Aside from experimental or computational indications of interaction, relationships between genes can be inferred from individual properties of the genes. Such *derived* interactions are typically less reliable than their experimentally-measured counterparts, yet still provide valuable information. Though each implicit interaction source may not be a perfectly accurate indicator of real interaction, the collection of implicit sources can offer compelling evidence that genes are related, in the absence of (and in addition to) more direct information.

Below we list a number of individual gene attributes from which interaction may be derived. The lower left of Figure 2.1 contains examples for yeast, typically beginning with the prefix 'CO'. Next to the description of the attribute, we provide the appropriate abbreviations of the *derived* or *implicit* experts used in our models.

- **Cell Cycle Phase** (COPHASE) Eukaryotic cells undergo a cycle of growth, replication and division. At each phase of the cell cycle, a distinct suite of genes becomes active to perform the necessary cell functions. Yeast, in particular, have well defined phase-specific groups. Over 200 of the 6500 yeast genes have known cell cycle phase, while over 800 have an experimentally predicted cell phase [SSZ+98].

- **Phenotype** (COPHENO) Phenotypes are observable characteristics of an organism produced by the organism's genetic makeup interacting with the environment. Examples include conditional phenotypes like heat or drug sensitivity, or defect phenotypes such as stress response defects. Phenotypes may be interrogated by obstructing the activity for an individual gene and observing the cell's response. In yeast, over 1400 genes have a known phenotype according to the MIPS Phenotype catalogue [MFG+02].

- **Essentiality** (COESSENT) Essentiality refers to whether the deletion of a gene results in a *viable* organism or is a *lethal* deletion. In yeast, virtually all genes are classified as viable or lethal according to the MIPS Disruption catalogue [MFG+02].

- **Protein Category** Proteins may belong to families (or categories) based on their function. For example, we mentioned that certain proteins are *transcription factors* while others are responsible for degrading proteins (proteases). In yeast, the MIPS Protein Category catalogue [MFG+02] lists protein families for nearly 1100 proteins (COPROCAT).

- **Protein Complexes** In order to carry out a specific function, proteins often organize into complexes. In yeast, the complexes are listed in the MIPS Complex catalogue [MFG+02] (COCMPLX) for over 2700 genes. Note that the information in this catalogue is not necessarily independent of the other sources. In particular, it includes some of the information found in the TAP, GAVINM and HOM experts described above.

- **Sequence Features** Regions of the DNA or protein sequence may contain patterns, or *motifs*, which confer important properties, for example affecting the binding properties or three-dimensional structure. In the case of yeast, these are given by the MIPS Motif catalogue [MFG+02], denoted by the expert COMOTIF. For mouse, there are two sources. The PFAM resource [BCD+04] (COPFAM) contains both a curated database and a generated database of conserved protein domains and families. PROSITE [HSL+04] (COPROSITE) contains mostly patterns found by its author as well as some published in the literature, many of which correlate strongly with the protein domains of PFAM.

- **Cellular Location** Typically proteins cannot physically interact or functionally interact unless they are in the same cellular compartment. Thus, location provides a weak indicator of potential interaction, even aiding in flagging potential false positives of other methods by identifying suspect interactions indicated between genes known to inhabit different cellular compartments. In yeast, the MIPS Localization catalogue [MFG+02] (COLOC) includes cellular locations for over 5100 genes. For most other organisms, locations are assigned using the *cellular component* branch of the *Gene Ontology* [ABB+00] (GO_CC).

- **Function or Pathway** Proteins coordinate with one another to achieve certain biological objectives in the cell. The objectives can be described as a specific *function* or organized into a sequence of events, *i.e.,* a *pathway*. For example, a function might be cell cycle and DNA processing and a pathway might be the galactose pathway which describes the process of conversion of the sugar galactose into glucose. The terminology used by any particular information source might not correspond directly with another but the general meaning is to describe the (hierarchy of) activity of proteins. In yeast, the MIPS Functional Catalogue [MFG+02] (COFUNC) lists the function of nearly 5000 genes which loosely corresponds to the terms assigned using the *molecular function* branch of the Gene Ontology (GO_MF). In terms of pathways, the KEGG [KGH+06] and GenMAPP [DSV+02] databases contain hand-drawn metabolic and

signalling pathways for many organisms. We denote these using COKPATH and COGMAPP, respectively. The *biological process* branch of the Gene Ontology describes a series of events or molecular functions (GO_BP). The GO definition of a biological process is not strictly equivalent to pathways, though certain terms do describe pathways.

Note that in Figure 2.1, the sources for describing cellular location are grouped under the node LOCREL while the sources generally describing function or pathway are listed under the FUNREL node. These intermediate nodes capture the potential correlation between the various data sources beneath.

2.2.3 Previous Uses of Interaction Sources and Interaction Networks

Many researchers have examined the networks created from the interaction sources. Some of their work concentrates on revealing properties of individual sources, often comparing them, while a few others combine the resources. However, most of the combinations use only a few sources and even then, there is little attention paid to quantifying the relative reliability of the sources. The novel contribution of this thesis is the probabilistic integration of numerous data sources which account for scale, reliability and density of the information. And our purpose is unlike previous applications since we wish to infer whether any type of relationship exists among genes rather than predict one type of information.

The studies by von Mering *et al.* [vMKS$^+$02], Deane *et al.* [DSXE02], Bader and Hogue [BH02] and Huynen *et al.* [HSvMB03] assess the overall quality, bias and overlap of the high-throughput sources, such as Y2H and TAP, compared to each other and to the collection of low-throughput sources, such as SYNLETH and IMMUPREP listed in the DIP and BIND databases. Note that all of these sources examine explicit interactions only. The studies report poor overlap of sources, a bias toward certain function or toward abundant proteins and a bias for proteins in specific cellular locations. Moreover, they found that over 50% of the high-throughput interactions are spurious, when compared to known protein complexes or known interactions. In regard to the ability to link functionally similar proteins, the techniques of gene fusion and gene-order conservation were found to have high reliability (72% and 80% respectively) [YDD01, YMD02, vMHJ$^+$03]. Though all the authors did not provide qualitative estimates of the reliability of the individual sources, all of these studies found the highest accuracy for those interactions indicated by more than one method. This argues strongly for a principled integration, not only to provide better coverage, but to increase accuracy.

To assess the quality of individual interactions, the studies by Saito *et al.* [SSH03], Bader [Bad03], and Bader *et al.* [BCRC04] examine topological properties of the networks implied by the sources. Identifying bridges, hubs, articulation points, path lengths and indegrees of nodes, they found that nodes with indegree one are likely spurious interactions, that proteins with lethal deletions (see COESSENT in Section 2.2.2) have more partners, and often proteins with path lengths of two physical interactions are connected by a single genetic interaction. Each of these authors develops an index for determining the reliability of a protein interaction based on these observations. Though their

techniques provide an alternative method for assessing reliability, they do not make use of the implicit experts, as we wish to do.

Often one source (or a combination of sources) is used to predict information from another source. The most popular examples of this combine the different sources to predict function [MPT+99, VFMV03, LK03, SL03, TDO+03, DTSC04, NJA+05], to predict protein complexes or interaction [BH03, JYG+03, KvMB03, JEMF05, SIK+05], and to predict essentiality [JOB03, PWJ04]. Several approaches incorporate a number of heterogeneous data sources [MPT+99, JYG+03, TDO+03, NJA+05, SIK+05]. Some of these techniques, like ours, even have a probabilistic basis. However, we are not interested in predicting a single type of relationship; we wish to predict the existence of any type of relationship (through SEMREL) which can not be easily accomplished using these methods.

In general, the scope of the previous studies does not capture the goal of this thesis. Namely, we wish to generalize the notion of relationship such that the probabilistic combination of all sources can be used in later tasks. In the Introduction and Chapter 8, we discuss and later demonstrate how inducing a network from the high-probability interactions can provide biologists with a global overview of the biological system. Note that this task necessarily argues for the more comprehensive definition of SEMREL since the view presented to the biologist should be as complete as possible. If only physical or functional interactions were displayed, many important relationships would not appear. Also, our choice to include implicit sources has great merit since these sources can provide information in the absence of any other more direct indication of interaction, as mentioned before. Most previous research does not incorporate implicit information, unless as a measure of validation and even then they do not use as many sources as we employ. In the next section and Chapter 7, we describe another task which can make use of the probabilistic combination of all sources: inferring regulatory networks.

2.3 Regulatory Networks

For proteins to perform functions in the cell, their activity is regulated by yet other proteins. In particular, certain proteins cause the production, or *expression* of mRNA for other genes, either directly because those proteins are transcription factors, or indirectly by cell-signaling through intermediate proteins. The expressed mRNA then signals production of a protein specific to the message carried in the mRNA. The completed protein can then perform its function, which may include the regulation of other genes, causing a cascade of regulation events. Biologists wish to uncover this network of coordinated regulation since patterns of gene expression influence many fundamental biological phenomena such as cell growth, morphology, cell differentiation, the response to external stimuli and environmental conditions, as well as disease [Str89].

One popular approach to inferring regulatory relationships is to manipulate a certain protein in the cell and measure the subsequent mRNA expression level of potential downstream targets. Until recently, biologists were limited in their ability to interrogate a large number of such relationships. Beginning in 1996, technological advances in the form of *microarrays* have allowed the monitoring

of tens of thousands of genes simultaneously, revolutionizing the way biologists do science. In the following sections, we examine this type of data and its uses, including for inference of regulatory networks.

2.4 Microarray Expression Data For Learning Regulatory Networks

Gene expression describes the process by which a gene is *transcribed* from the DNA to create mRNA which is then *translated* by the cell to produce a protein. Since most cellular functions are performed by proteins, biologists are interested in the complement of proteins expressed in a cell at any given time. Unfortunately, several factors complicate the ability to measure protein levels, including the difficulty in expressing particular proteins in sufficient quantities for assay and difficulty in purifying proteins. Also identifying proteins in a sample can be a challenge since alternative 3-dimensional configurations and modifications can cause proteins to be indistinguishable from one another using current techniques.

An alternative to measuring protein expression is to measure the corresponding mRNA expression levels. Certain properties of an mRNA molecule make it extremely amenable to experimental interrogation. Although presence of mRNA does not immediately guarantee the later presence of the protein, the respective activity levels are roughly correlated. We use the term *gene expression data* to refer to measurements of mRNA levels.

Early techniques for quantifying mRNA expression were high-cost, low-throughput techniques typically limited to measuring between 5 and 20 genes. These methods include quantitative RT-PCR [RTS95], SAGE [VZVK95], and Northern blots [Kru94]. Since roughly 1996, biologists have been able to study mRNA expression levels for tens of thousands of genes simultaneously using *microarrays*. This technology affixes markers, or *probes*, specific to a sequence of mRNA in a regular pattern across a glass slide or tray. Typically there are between 12,000 and 30,000 markers per slide, hence the *micro* designation. The identity of a particular mRNA is given by a two-dimensional coordinate, hence the *array* designation.

To measure the mRNA levels in a particular biological sample, the mRNA is extracted and labelled with a fluorescent tag. A solution containing the labelled sample is then poured over the microarray. If a particular mRNA is present in the sample, it will bind (*hybridize*) to its specific marker on the array and then its tag will be detected using a laser. Due to the regular arrangement of the probes on the slide, the signal from a particular bound mRNA can readily be identified with image processing software. The amount of mRNA can be quantified by the strength of the fluorescent signal.

Two main technologies are used for gene expression microarrays: oligonucleotide chips [LDB+96] which measure absolute levels of mRNA in a single sample, and two-color cDNA microarrays [SSH+96] which measure relative levels of mRNA between two samples. An example use of a two-color cDNA microarray is illustrated in Figure 2.2. A cDNA microarray is typically used to determine which

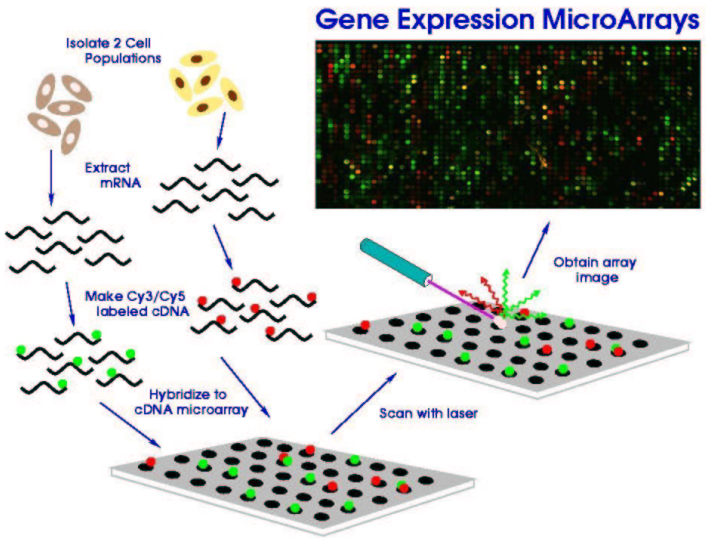


Figure 2.2: Gene Expression Microarrays

genes are differentially expressed in one cell type or another, such as a cancer cell versus a normal cell. Here, let the brown cells on the left represent normal cells and the yellow cells on the right represent cancer cells. The mRNA from each cell type is transformed into complementary DNA (cDNA) which is then fluorescently labelled and hybridized to the array. Two fluors are used: Cy3 which by convention is interpreted as *green* signals for mRNA present in the normal cells (*i.e.*, the control) and Cy5 which is interpreted as *red* signals for mRNA present in the cancer cells (*i.e.*, the experiment). The image in the top right is then the result of scanning the array, such that a bright red spot indicates overabundance of a particular mRNA in the cancer cell versus the normal cell (over-expression), and a bright green spot indicates deficiency of expression of the mRNA in the cancer cell versus normal (under-expression). A yellow spot means the mRNA was equally present in both cell types. Over-expression or under-expression of a gene might suggest a defect in function that results in the disease.

Data from single samples can be concatenated to obtain an expression dataset. The dataset can be *static* expression data, collected from a number of independently perturbed samples, or *dynamic* expression data, collected over time from a system in flux. A popular representation of a dataset, called a *heatmap*, is shown in Figure 2.3, taken from Spellman *et al.* [SSZ+98]. This dataset measures the mRNA expression of over 800 cell-cycle dependent yeast genes (see COPHASE description in Section 2.2.2). Using four different techniques to synchronize the cell-cycle of the sample (Alpha, cdc15, cdc28 and Elu), samples are taken in roughly 10 minute increments over one to two complete cell cycles, for a combined total of 72 samples. Green signals indicate under-expression of the gene versus time zero and red signals indicate over-expression versus time zero. Black signals indicate no relative change in expression from time zero, or by convention, missing data. Each column in the figure represents a microarray sample, ordered by cell cycle phase (M/G1, G1, S, G2, M), and each

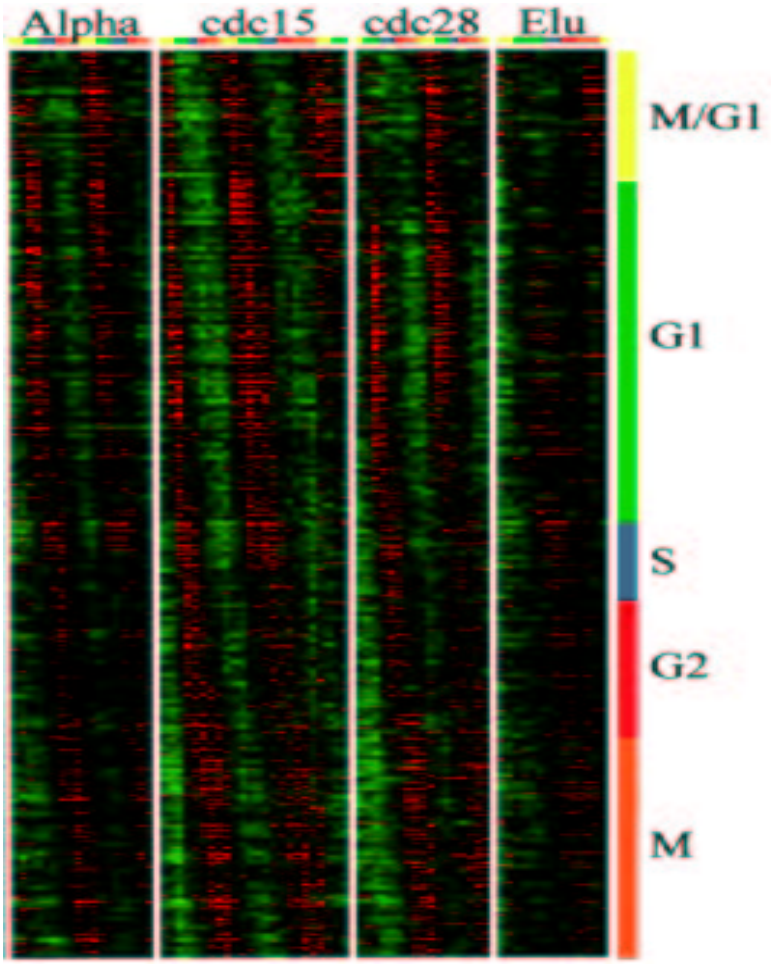


Figure 2.3: Expression data for Cell Cycle Phase, taken from Spellman *et al.* [SSZ+98]

row represents one gene, organized by the phase in which expression for that gene peaks (maximum red signal).

The example of Figure 2.3 suggests how gene expression data might be used to infer regulatory networks. Cell-cycle dependent expression is regulated by key transcription factors, where the expression of a transcription factor in one cell phase causes a cascade of expression of its targets in a later phase. Note that the genes have been ordered along the vertical axis in the figure so that this periodic behavior of expression is immediately apparent as alternating vertical bands of green and red. The study of temporal dependence of gene expression can suggest or verify transcription factor-target relationships. Another use of gene expression data to interrogate causal relationships gathers samples where one gene has been disrupted and its effects on the remaining genes is measured. This approach was illustrated in Figure 1.1 for the disruption of GAL4 and its effect on the genes of the galactose and related pathways [ITR+01]. The next section discusses other uses of expression data.

2.4.1 Previous Uses of Expression Data

Examples from a rich body of literature dedicated to the use of gene expression data include inferring the function of unknown genes [CCW+98, SSZ+98, HMJ+00], creating gene deletion phenotype profiles [AED+00], creating temporal patterns in response to environmental stimuli [DIB97, CDE+98, WFM+98, IER+99], mining for putative regulatory elements [CCW+98, HBKS00, HB00, VBJ+00], mutation site typing [RSL+00], pathway refinement [ZKZL00, TS01], reducing the need for sequencing entire cDNA libraries [HSL+99] and classifying tumor types [ABN+99, GST+99].

Our primary interest in expression data is for the purpose of inferring regulatory networks. Any attempt to learn a regulatory network is limited to learning only those relationships presented in the mRNA expression data. Regulation which occurs at the level of proteins (through post-translational modifications) may not be revealed directly through the expression data, though we may observe indirect effects.

A number of different approaches have used expression data to estimate gene regulatory networks. Early methods used clustering techniques to find groups of genes with correlated gene expression profiles [DIB97, SSZ+98, CDE+98, WFM+98, CCW+98, IER+99, HMJ+00], operating under the premise that targets of a particular transcription factor would have similar expression patterns. The discussion above of the cell-cycle example in Figure 2.3 alluded to such use. DNA sequences upstream of all genes in a given cluster are then searched for common putative regulatory elements.

Other approaches attempt to mathematically characterize relationships in the expression data through the use of Boolean networks [LFS98, AMK99, MTO+01, SDKZ02], qualitative models [TT98, AMK00, SLP02], linear models [DWFS99, vSWR00], differential equations [CHC99, dHIK+03, ZGSD03], Probabilistic Relational Models [STG+01, BSK05, SS05], and Bayesian networks [BF01, HGJY01, PREF01, HGJY02, IGM02, TKB+03]. We examine the use of Bayesian Networks more fully in the next section.

2.5 Regulatory Networks as Bayesian Networks

A popular representation of regulatory networks is as *Bayesian networks* [Pea88]. Upcoming chapters fully describe this formalism and our adaptations; we provide motivation and a brief outline of that approach here.

Bayesian networks are probabilistic graphical models, where genes are represented as nodes in a graph while arcs depict dependencies between them. The dependencies are qualified by a probability distribution. Ideally, the arcs would have a causal interpretation, thereby capturing the notion of regulation. Bayesian networks have great appeal for use in representing regulatory networks since their probabilistic nature captures noise in the expression data while the graphical component easily conveys the dependencies between the genes, making them readily interpretable by biologists. Moreover, the theory of Bayesian networks provides a natural means for incorporating prior domain information. Prior information is incorporated as a *structural* prior influencing the presence or absence of edges in the graph, or as a *parameter* prior influencing the quantification of the dependencies indicated by those edges.

Several groups have used some form of Bayesian networks for learning regulatory relationships. Pe'er *et. al* [PREF01] find subnetworks of interacting genes by learning an ensemble of models and extracting statistically confident subgraphs occurring in many of the models. Their models are constructed by discretizing the gene expression data, thereby assuming discrete variables. Imoto *et al.* [IGM02] use the continuous expression levels and model the dependence between a node and its parent using nonparametric regression. For a target gene, they find potential regulators using a Bayesian scoring metric and construct a larger network by concatenating the local results. In both of these approaches, prior information about genes known to regulate other genes was used post hoc to validate the subnetworks found.

Barash and Friedman [BF01] and Tamada *et al.* [TKB+03] use the clustering approach to identify potential co-regulation and attempt to improve cluster quality by including additional information during clustering. They estimate the joint distribution of expression data and transcription factor binding site data (see discussion of TXREL in Section 2.2.1). Hanisch *et al.* [HZZL02] learn a joint distribution over expression data and metabolic networks. Note that the transcription factor data or metabolic network data provide additional attributes over which to learn rather than any prior bias.

Using prior information to bias Bayesian network learning has been attempted by Hartemink and colleagues who used Bayesian networks to model a small well-known network for galactose metabolism [HGJY01] and the pheromone response pathway [HGJY02]. The former paper incorporates information about specific gene pairs, annotating the corresponding edge in the graph by indicating whether one gene was known to positively or negatively influence the expression level of the other. The annotation affects what relationships could be learned when quantifying the dependencies among variables (*i.e.,* a parameter prior). In the latter research, location data (as in our LOCANA expert [LRR+02]) was used to (strictly) allow or disallow particular edges in the network

(*i.e.,* a structural prior). Le *et al.* [LBU04] similarly investigated the utility of including strong structural priors when learning from simulated data in a small, realistically created simulation model of hepatic glucose homeostasis. They found that the data requirements for learning the target model dropped as a larger percentage of edges were asserted or disallowed in the graphs learned. In this thesis, we would like to demonstrate that claim using real data, as discussed in the next section.

2.5.1 Gene Interaction Networks as Structural Priors

Learning regulatory networks presents an intriguing application since it not only challenges the state of the art of Bayesian network learning algorithms, but simultaneously offers a potential solution. The primary limitation in learning regulatory networks from expression data is the (small) number of samples versus the (large) number of genes. The magnitude of this disparity has not been seen in previous applications of Bayesian networks, and is a very real constraint since it is questionable whether there will ever be sufficient availability of samples in biology for completely de novo network reconstruction given the high cost or rare nature of the samples being studied.

The lack of samples has two important consequences. The most obvious effect is on the power of any statistic measuring the quality of a relationship between two genes. Insufficient power translates to an inability to identify quality models during search. For example, even learning a Bayesian network for N *binary* nodes requires $O(N^2 \log N^2 log N^{k+1})$ samples, where k denotes the maximum indegree of a node [Hof93]. Currently, the amount of data for learning is well below such a requirement. To circumvent this difficulty, most of the approaches listed above lower the complexity of the learning task by either using simulated data, simplifying the form of relationship learned, or learning relationships for only a subset of genes.

Where the first consequence involves the *quality* of the search, the second consequence involves the *speed* of the search. Learning a Bayesian network is an NP-complete problem [Chi96], therefore typical learning algorithms resort to greedy searches. With limited data, the functions being optimized during learning suffer greatly from local maxima, causing any greedy search to waste time exploring suboptimal solutions. Ideally, we wish to concentrate the search on the maximal peaks of the optimization landscape. The same techniques listed above can also be used to improve search speed by limiting the size of the search space in order to reach high quality solutions more quickly. To address the fact that numerous good solutions of near equivalence exist with small datasets, many previous approaches pool results from many searches (final models) and extract high-confidence subnetworks or *features* [PREF01, HGJY02]. This is a well-known technique called *Bayesian model averaging* [MGR95].

Even though the challenge provided by our biological application is rather unique in terms of complexity, biology also offers assistance for the task in the form of additional, almost independent, sources of information about genes. The difficulty lies in finding the appropriate use for that information to aid our objective of learning regulatory networks from expression data. Our goal is to show that by incorporating additional information as priors, we are able to improve both the speed and quality of the search.

To do so, this thesis explores an alternative for Bayesian network learning which directly addresses both issues by using prior information about gene relationships. The basic idea is to provide the learning algorithm with our best estimate of the true model and use the available expression data to update that belief. Thus model learning becomes model refinement. In previous sections we alluded to how interaction information sources might be combined to create a probabilistic consensus belief of interaction, that is, a quantified estimate of which genes might influence which others. Our Bayesian network learning scheme would use this consensus belief during learning, both to seed the initial model of the search, as well as guide the learning during search.

Starting at a better model addresses the issues of quality and speed since we begin exploration of a higher peak of the objective function landscape, which will potentially result in higher local maxima at the end of the search. Searches are shorter since we begin closer to better solutions, avoiding needless exploration of suboptimal alternatives. Prior information can be used not only at the start of learning but during the process of learning. Using our consensus belief during learning improves the speed to find better solutions by focusing the direction of the search toward more promising alternatives.

We propose to use the consensus belief as a *structural* prior during Bayesian network learning. Though information about the polarity of influence between genes exists, as used by Hartemink *et al.* [HGJY01] for parameter priors, the majority of information available in a computable form asserts (unqualified) relationships between genes and is thus better suited for creating structural priors in this domain. Though it seems intuitive that jumpstarting the learning by providing prior information will improve the search, it is not an immediate conclusion in this domain.

Regulation in our context describes the influence of a protein to initiate transcription of another gene, *i.e.,* to produce mRNA. For this reason, expression data is particularly well suited to the task of inferring regulatory networks since transcriptional regulation involves mRNA, the very molecule that is being measured and used for learning. However, as catalogued in previous sections, the type of information available for asserting relationships is varied, both in terms of the gene product being measured (DNA, mRNA or protein) and in terms of the nature of the relationship (explicit versus implicit, physical versus genetic, etc). It remains to be validated experimentally whether interaction evidence can be successful prior information when learning from gene expression data.

In the next three chapters, we formally introduce Bayesian networks, discuss previous work in incorporating structural prior information, and present our approaches to using gene interaction information, respectively. Throughout the discussion, we highlight issues specific for learning in this domain and how we address those issues within our methods.

Chapter 3

Bayesian Networks

In the Introduction and Section 2.5.1, we stated our claim that using informed structural priors can help Bayesian network learning algorithms find higher quality models and speed the time to find those models during search. Toward that end, we first formally describe Bayesian networks and their learning algorithms and then discuss previous uses of prior information during learning (continued in Chapter 4). Our presentation follows closely that of Cooper and Herskovits [CH92] and Heckerman *et al.* [HGC95]. Additional material can be found in the book by Pearl [Pea88].

3.1 Definition of Bayesian Networks

Bayesian networks (also known as belief networks) are a type of probabilistic graphical model. They provide a compact representation of the joint probability distribution over a set of random variables $\mathbf{X} = X_1, \ldots, X_N$. The set $\mathbf{X}$ can be (any combination of) continuous or discrete variables. A Bayesian network $B =< S, \theta >$ consists of a *structure* component S which is a *directed acyclic graph* (DAG) whose vertices correspond to the random variables, and a *parameter* component θ which consists of a collection of conditional probability distributions specifying the probability of a variable given its immediate parents in S. These two components are sufficient to represent any distribution over $\mathbf{X}$ [Pea88].

One of the most important features of a Bayesian network is the ability to explicitly represent conditional independence and dependence among variables. The structure of the graph S captures the conditional independencies of the joint distribution, in particular the *Markov independencies* which state that each variable X_i is independent of its non-descendants given its parents in S.[1] A distribution which satisfies this condition can be decomposed into the *product form*:

$$Pr(X_1, \ldots, X_N) = \prod_{i=1}^{N} Pr(X_i \mid \pi_i)$$

[1] Other types of independencies can also be readily identified by visual inspection of the network structure. See [Pea88, Chi96] for details about v-structures and d-separation.

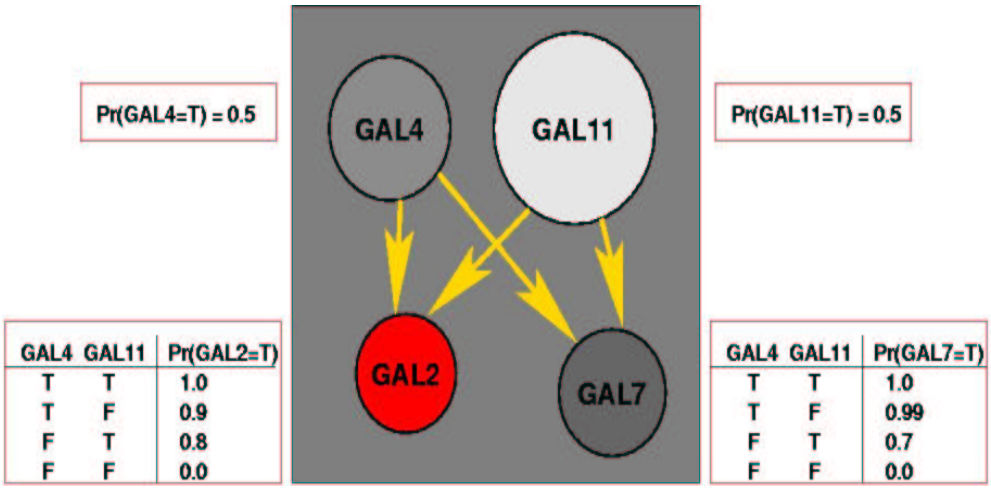


Figure 3.1: Example Bayesian Network

where π_i is the set of parents of X_i in S. The probability of any instantiation of all variables can then be computed as the product of only N probabilities. Thus, a Bayesian network provides a compact and efficient representation of the complete joint probability distribution by factoring the joint into smaller, local conditional probability distributions subject to the Markov independence property. The compact representation also allows Bayesian networks to be easily queried; marginal probabilities are found by exploiting the local factorization, allowing the prediction of values of one subset of variables given value assignments to any other subset.

An example network is shown in Figure 3.1 for four binary variables: GAL4, GAL11, GAL2, and GAL7. The conditional probability tables quantify the relationships between the variables. For example, the conditional probability distribution of GAL7 essentially encodes a *noisy-OR* function of its parents where the probability of GAL7=true increases if either parent is true.[2] Since the probabilities sum to one, the probability that any variable is false is just one minus the probability that the variable is true. Using a Bayesian network, we see the compactness of the representation even for this simple example; we can represent the joint distribution over these four variables by specifying only 10 parameters instead of enumerating all $2^4 = 16$ possible values.

3.2 Learning Bayesian Networks

The task of learning a Bayesian network is to estimate an unknown distribution $Pr(\mathbf{X})$ by a network $B =< S, \theta >$ given a dataset $D = \mathbf{X}_i, \ldots, \mathbf{X}_M$ of M independent samples from $Pr(\mathbf{X})$. Depending upon the task, the goal might be to use the data to estimate both the parameters and the structure of the model or to only estimate the parameters given a structure. A general depiction of the task is given in Figure 3.2 where a dataset of N variables and M samples is presented to a learning

[2] Noisy-OR will be more formally described in Section 3.2.4.

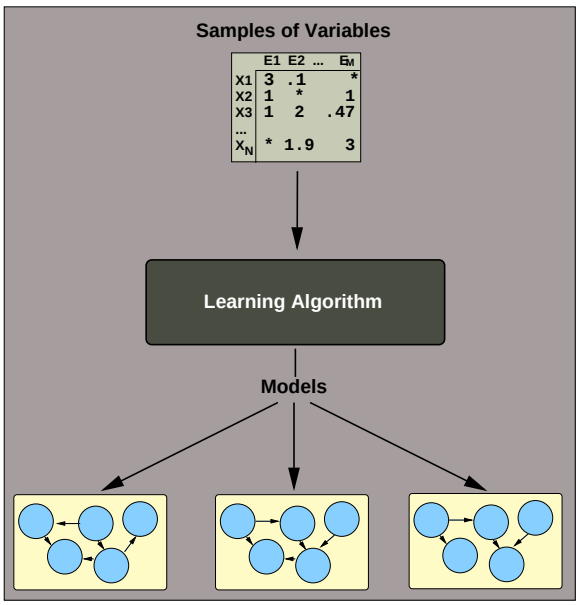


Figure 3.2: General Bayesian Network Learning Algorithm

algorithm which outputs potential models. In the following sections we will be refining this basic picture by describing each aspect in more detail.

Since the topic of this thesis concerns learning the structure, we first provide an overview for structure learning in the next section before detailing how to compute the probabilities given a structure in Section 3.2.3. In Section 3.2.4, we also consider the special case of learning parameters for a node whose the conditional probability distribution is represented as a *noisy-OR* function. The noisy-OR appears repeatedly in our research.

3.2.1 Learning Structure: Search and Score

A common structure learning strategy is the *search and score* approach which searches over the space of all possible DAGs while optimizing a statistically motivated scoring function [Hec95]. A candidate network is scored by how well it can encode the (in)dependencies apparent in the data. A popular scoring metric is the *Bayesian score* which compares candidate graphs S by their posterior probability $Pr(S \mid D)$ given the data D:

$$Pr(S \mid D) \propto Pr(S)Pr(D \mid S). \tag{3.1}$$

The term $Pr(S)$ denotes the *prior distribution over structures* and the term $Pr(D \mid S)$ denotes the *marginal likelihood of the data* given the structure. The latter term involves an integration over all possible parameterizations compatible with the given structure:

$$Pr(D \mid S) = \int_{\theta} Pr(D \mid S, \theta)Pr(\theta \mid S)\ d\theta. \tag{3.2}$$

Typically, the formulas are transformed into their logarithmic equivalents which involve simple summations rather than products. Note that learning a Bayesian network in this context technically does not learn a point estimate for the parameters, rather the procedure searches the space of structures to find a graph that, on average, allows a parameterization which captures well the dependencies apparent in the data. A point estimate of the parameters can be estimated, as described later in Section 3.2.3, once a high-scoring structure is found.

Computing the Bayesian score of any graph S thus requires specifying a set of *structural priors* $Pr(S)$ and a set of *parameter priors* $Pr(\theta|S)$. These factors encode existing knowledge about the domain to bias the learning toward models with particular graphical structures or toward particular functions in the conditional probability distributions. In fact, one of the major attractions of Bayesian network learning is this ability to naturally include prior information during learning. Priors become particularly important in biased or limited data applications such as ours since they provide estimates for quantities not observed, thereby ensuring that the learned models generalize.

Surprisingly, previous work on providing priors has centered almost exclusively on specifying conditions for the *parameter* priors under which Eq (3.2) has a closed form solution [CH92, HGC95]. Very little focus has been paid to incorporating informative *structural* priors and for this reason, this thesis represents a significant contribution by developing methods for incorporating complex structural prior information sources. Further discussion of these points is delayed to Section 3.2.2 where we elaborate on conditions for the parameter priors which yield an analytical solution to Eq (3.2) and to Chapter 4 where we review previous work on structural priors before presenting our approach to formulating informative structural priors in Chapter 5.

Given the scoring function with appropriately specified parameter and structural priors, the search component of the search and score technique then proceeds to explore the space of structures to maximize the Bayesian score. This problem is known to be NP-hard [Chi96] since the space of DAGs is super-exponential in the number of nodes [CH92]. Thus, typical applications resort to using a greedy hill-climbing procedure which iterates to achieves a local maximum. For this procedure, the search is initialized with some starting graph. At each subsequent iteration, all possible one-arc modifications to the current graph (adding, deleting, or reversing an arc) are scored and the maximum among them is chosen for the next iteration. The process continues until no further one-arc change results in a higher scoring graph with respect to the Bayesian score.

Typically, to avoid local maxima the search is performed from many initial models, each iterated to convergence, and either the best scoring final model is chosen as the final output or several models are averaged. See [MGR95] for further discussion of Bayesian model averaging. Also, there are other alternative to the search procedure, such as simulated annealing or best-first search, though Chickering [Chi96] has shown that for a fixed amount of computation time, greedy search with random restarts produces better models than does either of the others. For this reason, we will use greedy hill-climbing in the experiments of Chapter 7, paying particular attention to the set of initial and final graphs from multiple search runs.

3.2.2 Parameter Priors: the BD, BDe, and BDeu metrics

The calculation of the marginal likelihood component of the Bayesian score is computationally intensive. However, a closed form solution exists if the following assumptions hold [CH92, SDLC93, DL93, Bun91, HGC95]:

1. **(Multinomial Sample)** The database D is a multinomial sample, *i.e.,* the variables $\mathbf{X}$ are discrete. Appendix A examines a large host of discretization algorithms for the case where (some of) the variables are continuous-valued.

2. **(Complete Data)** The database is complete, *i.e.,* there are no missing values. In Section 7.3.1, we will discuss one method for imputing missing values which is useful in cases where the data is not missing at random.

3. **(Data Independence)** The cases in D are independent given a Bayesian network.

4. **(Parameter Independence)** The parameters associated with each variable given the network structure are independent.

5. **(Parameter Modularity)** The parameters associated with each variable depend only on the set of parents of the variable, *i.e.,* if a node has the same parents in two distinct networks, then the parameters associated with the node are identical in both networks.

6. **(Dirichlet Priors)** The prior distribution of the parameters is a *Dirichlet distribution* [DeG70]. If a discrete node X_i has r_i possible values and the parents π_i are instantiated to their j-th configuration, let θ_{ij} be the vector of probabilities for $Pr(X_i|\pi_i = j)$. Then there exist numbers (virtual counts) $N'_{ijk} > 0$ such that:

$$Pr(\theta_{ij} \mid S) = \frac{\Gamma(\sum_{k=1}^{r_i} N'_{ijk})}{\prod_{k=1}^{r_i} \Gamma(N'_{ijk})} \prod_{k=1}^{r_i} \theta_{ijk}^{N'_{ijk}-1}, \tag{3.3}$$

   where the *Gamma* function is the generalized factorial function satisfying $\Gamma(x+1) = x\Gamma(x)$.

In order to compute Eq (3.2), we also need to compute the term for the likelihood of the data given the structure and the parameters. For this, we must compute sufficient statistics from the database. If q_i is the number of possible parent instantiations for the parents π_i of variable X_i, then let N_{ijk} count the number of datacases in which $X_i = k$ when π_i assume their j-th configuration. Note that the virtual counts N'_{ijk} of the Dirichlet in Assumption 6 can be seen as pseudo counts for 'imaginary' datacases, similar to the actual database sufficient statistics.

Given the above assumptions and the sufficient statistics, the Bayesian score can then be rewritten as:

$$\begin{aligned} Pr(S \mid D) &\propto Pr(S)Pr(D \mid S) \\ &= Pr(S) \int_{\theta} Pr(D \mid S, \theta) Pr(\theta \mid S)\, d\theta \\ &= Pr(S) \prod_{i=1}^{N} \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ij})}{\Gamma(N_{ij} + N'_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + N'_{ijk})}{\Gamma(N'_{ijk})}, \end{aligned} \tag{3.4}$$

where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ and $N'_{ij} = \sum_{k=1}^{r_i} N'_{ijk}$. This metric is known as the *BD metric (Bayesian Metric with Dirichlet priors)* [CH92]. Note that the score decomposes into terms for each variable, a property that is crucial for the one-arc operators used in the greedy hill-climbing search procedure. Local changes to the graph can be easily evaluated by recomputing only those terms in the product involving nodes affected by the change.

There are two key disadvantages to the BD derivation: 1) a domain expert must specify the virtual counts N'_{ijk} for all values ijk; and 2) networks that are *likelihood equivalent* may not receive the same score. Likelihood equivalent networks are those which encode the same assertions about conditional independence. In this case, the data should not be used to distinguish between the networks, as can result when using the BD metric [HGC95].

To overcome the first disadvantage of specifying a large number of values for N'_{ijk}, several special cases of priors have been considered. The first two are *non-informative*, or *uninformed*, priors. Setting all $N'_{ijk} = 1$ is known as the *uniform Dirichlet prior* [CH91, CH92] which is equivalent to estimation without prior information [BY98]. A variant is the *Jeffreys' prior* where all $N'_{ijk} = \frac{1}{2}$ that, unlike the uniform Dirichlet prior, is invariant to the choice of parameterization of the model [Jef61, BY98]. A third alternative, not based on the Dirichlet distribution, is the *entropic prior* [Bra99], which asserts that parameters which do not reduce uncertainty are improbable. This prior minimizes the entropy of the model by calculating $Pr(\theta)$ as:

$$Pr(\theta) = \prod_i \theta_i^{\theta_i}.$$

This prior is non-informative in that it does not favor one parameter set over another provided they specify equally uncertain models. And like Jeffreys' prior, it is parameterization invariant. All three of these examples simplify the specification of priors over the parameters but none simultaneously satisfies the property of likelihood equivalence.

To alleviate both problems of the BD metric, a variation called the *BDe metric (Bayesian metric with Dirichlet priors and likelihood equivalence)* was proposed which assumes likelihood equivalence. Enforcing this assumption has the important consequence of imposing constraints on the Dirichlet parameters N'_{ijk} [HGC95]. The constraints allow the BDe metric to only require the domain expert to specify an *equivalent sample size* N' and a *prior Bayesian network* B^p from which all N'_{ijk} can be calculated as:

$$N'_{ijk} = N' \cdot Pr(X_i = k, \pi_i = j \mid B^p).$$

Note that the set of parents π_i are those in the candidate network, not necessarily those in B^p. In fact, the details of B^p are irrelevant so long as the queries for the relevant quantities over node families can be computed. In the original derivation of Heckerman *et al.* [HGC95], the formula actually conditions on a *complete* network compatible with B^p.

A further variation of the metric was introduced by Buntine [Bun91] and termed the *BDeu metric (Bayesian metric with Dirichlet priors, likelihood equivalence and uniform joint)* by Heckerman *et al.* [HGC95]. This metric assumes that every instance of the joint space conditioned on a complete prior network is equally likely. This distribution can be achieved by assigning the virtual counts

N'_{ijk} as follows:

$$N'_{ijk} = \frac{N'}{q_i r_i},$$

where again r_i is the number of discrete values for X_i, q_i is the number of possible instantiations for the parents π_i of X_i, and N' is the equivalent sample size.

The BDeu metric is often used in practice because it only requires the domain expert to specify a single parameter N'. The value of N' for both the BDe and the BDeu metric reflects the confidence that the domain expert has in the correctness of the parameter prior. Experiments confirm that learning is very sensitive to the choice of N', especially for the BDe metric. Heckerman *et al.* [HGC95] found that for a database of 10000 samples on 37 variables, performance using BDeu degraded for values of $N' > 16$. However, for databases with between 100 and 10000 samples on the same model, the uninformed BDeu metric performed better than using the informed BDe over a large range of values of N'. Our sample size of around 2000 falls within this range. Since the focus of the thesis is on investigating the effect of structural priors rather than parameter priors, we use the non-informative BDeu metric in the experiments of Chapter 7.

3.2.3 Learning Parameters: ML and MAP estimates and the EM algorithm

Structure learning, as presented in Section 3.2.1, searches for a structure by averaging over all possible parameterizations compatible with a structure, thereby not requiring one particular setting of the parameters. However, a point estimate for the parameters of a given structure can also be estimated from the data. Using the assumptions of the previous section, we can consider a *maximum likelihood (ML)* estimate or a *maximum a posteriori (MAP)* estimate. As the size of the data grows, the ML estimate approaches the MAP estimate [Hec95].

Each parameter θ_{ijk} in the model denotes the probability that $X_i = k$ given the j-th instantiation of its parents. The maximum likelihood (ML) estimate assumes each parametrization is a priori equally likely and uses the data to determine a value. The ML estimate is thus simply a count of the number of times a particular configuration of a node and its parents occurs in the database:

$$\theta^{\mathrm{ML}}_{ijk} = \frac{N_{ijk}}{N_{ij}}, \tag{3.5}$$

where as before, $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

The ML estimate is sensitive to biased or small databases, which is an issue in our application. To alleviate this problem, the maximum a posteriori (MAP) estimate considers a prior over the parameter values and uses the database to update the initial beliefs. For the case of the Dirichlet priors seen in the previous section, the MAP estimate is simply the count of occurrences for a configuration, augmented with the virtual counts provided by the parameters of the Dirichlet:

$$\theta^{\mathrm{MAP}}_{ijk} = \frac{N_{ijk} + N'_{ijk}}{N_{ij} + N'_{ij}}. \tag{3.6}$$

The ML and MAP estimates given above assume that the sufficient statistics are computed on a *complete* database. When values are missing, the parameters are no longer independent and can no longer be estimated by simple counts. In this case, we can approximate using the *Expectation Maximization (EM)* algorithm [DLR77] which replaces the actual counts by expected counts. The algorithm begins with an initial guess for the parameters and iterates between two steps, expectation (the E-step) and maximization (the M-step). In the E-step, expected values for the unobserved variables are computed given the observed data and the current parameter estimates. In the M-step, the parameters are updated by the same formulas as Eq(3.5) and Eq(3.6), using the expected counts as if they were the real counts.

Let $\mathbf{X}_l$ denote each (possibly incomplete) datacase, then the expected value of each missing variable $X_i \in \mathbf{X}_l$ can be computed based on the assignment of the other variables by querying the network. To perform this query, we must use an inference procedure. Since exact inference in complex Bayesian networks is intractable, we use the popular approximation known as the *loopy belief propagation* algorithm [Pea88], which has been shown to yield reasonable results on a large class of problems [MWJ99]. We refer the reader to these sources for further information.

We can compute the expected sufficient statistics by summing over all M independent datacases and current parameter estimates θ':

$$E[N_{ijk}] = \sum_{m=1}^{M} Pr(X_i = k, \pi_i = j \mid \mathbf{X}_l, \theta'),$$

where the expectation is taken over $Pr(X \mid D, \theta)$. These estimates are then used in place of N_{ijk} in the ML and MAP estimates:

$$\theta_{ijk}^{\mathrm{ML}} = \frac{E[N_{ijk}]}{\sum_{k'=1}^{r_i} E[N_{ijk'}]}$$

and

$$\theta_{ijk}^{\mathrm{MAP}} = \frac{E[N_{ijk}] + N'_{ijk}}{\sum_{k'=1}^{r_i} E[N_{ijk'}] + N'_{ij}}.$$

The EM algorithm is described here as a way to estimate *parameters* in the presence of missing data. We would like to briefly mention that it is also used by the *Structural EM (SEM)* algorithm [Fri98] to learn *structure* from incomplete data. The basic idea behind SEM is to augment the search and score algorithm with an inner loop that estimates parameters by EM to calculate sufficient statistics to use in the scoring metric. We refer the reader to the original paper for further details.

3.2.4 Learning Parameters: Noisy-OR

Often based on prior knowledge of the relationships among variables or to reduce the complexity of inference in Bayesian networks, the type of function allowed for the conditional probability distribution specified at each node is restricted to one with fewer parameters than the full multinomial. Perhaps the most popular simplification is the *noisy-OR* node, where the value of a (boolean) node is essentially the logical OR of its (boolean) parents except there is some probability that the effect

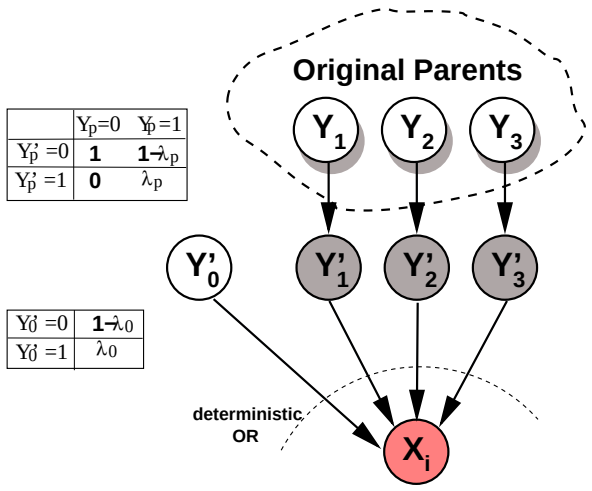


Figure 3.3: Noisy-OR Structural Decomposition

of each parent is inhibited. Inhibition of a parent means that the value of the node may be false even when the parent is true. The number of parameters for a noisy-OR nodes is thus linear in the number of parents. Since some of the techniques we develop for formulating structural priors use the noisy-OR, we briefly describe this node and techniques for learning its parameters.

For a binary node X_i with P parents, let q_p be the probability that parent p is inhibited. Also, let q_0 the inhibition probability of an additional (virtual) node, known as a *leak* node, which acts like a parent whose value is always 1 (true). The leak node accounts for all the unmodelled cases which might cause X to be set to 1 regardless of the values of its parents.

The probability of a noisy-OR node X_i given its parents π_i and the leak node L can then be computed as:

$$Pr(X_i = 0 \mid \pi_i, L) = \prod_{\{p|\pi_{ip}=1\}} q_p$$

and

$$Pr(X_i = 1 \mid \pi_i, L) = 1 - \prod_{\{p|\pi_{ip}=1\}} q_p,$$

where π_{ip} is the p-th parent of node i. Notice that in the case where all (actual) parents are false, the probability of X_i depends only on the inhibition probability of the leak node.

The parameters of the noisy-OR function can easily be learned using EM by introducing a set of unobserved variables whose parents are the original parents of X_i. The value of X_i is then changed to become a *deterministic-OR* of the values of the unobserved variables. The process is illustrated in Figure 3.3 where a node X_i originally had parents $Y_1, \ldots, Y_P$ and Y_0 corresponds to the leak node. Let $\lambda_p = 1 - q_p$ where q_p is as defined above. The shaded nodes Y'_p denote the set of unobserved variables introduced to facilitate learning the corresponding inhibition parameters q_p.

In the table for the unobserved nodes, note that the parameters for $Y'_p = 0$ and $Y'_p = 1$ when $Y_p = 0$ are fixed during EM to 1 and 0, respectively. If using MAP estimates with Dirichlet

parameters, the remaining table values can be adjusted by the appropriate virtual counts. With M datacases and the current parameter estimates θ', the M-step is modified for these nodes as follows (ML estimation):

$$\lambda_p = \frac{\sum_{m=1}^{M} Pr(Y_p' = 1, Y_p = 1 \mid \mathbf{X}_l, \theta')}{\sum_{m=1}^{M} Pr(Y_p = 1 \mid \mathbf{X}_l, \theta')}$$

and

$$\lambda_0 = \frac{\sum_{m=1}^{M} Pr(Y_0' = 1 \mid \mathbf{X}_l, \theta')}{M}.$$

We use the noisy-OR formulation in two of our derivations for structural priors, described in Section 5.3 and Section 5.4.

3.3 Learning Bayesian Networks: Updated Framework

In Figure 3.4, we see an updated version of the general learning framework which incorporates the requirements and assumptions described thus far. The data is complete and discrete. The learning algorithm is the greedy hill-climbing procedure which performs local search using one-arc changes to the current graph, scoring each graph using the variant of the Bayesian scoring metric known as BDeu. The learning procedure uses random restarts to output a set of locally maximal models. The only remaining component is to incorporate domain knowledge about the relationships among variables through the term $Pr(S)$, the prior over structures. We elaborate on the topic of structural priors in the next two chapters.

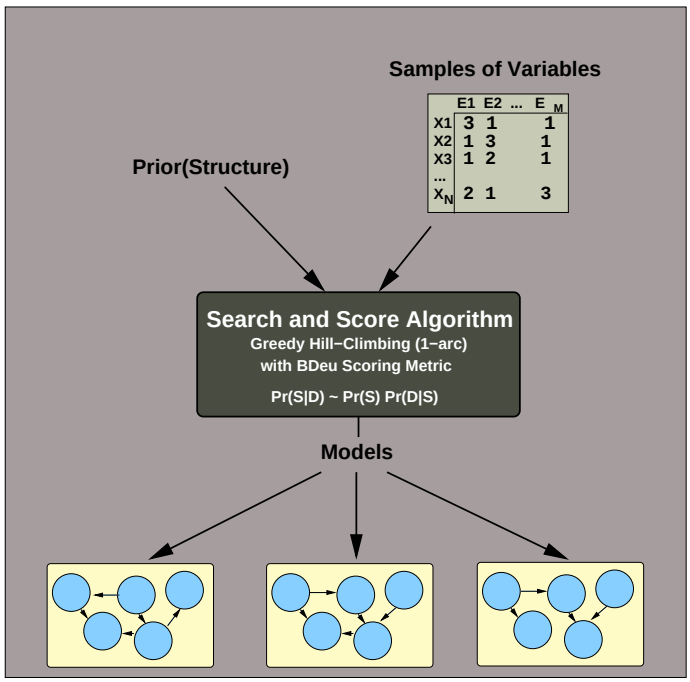


Figure 3.4: Updated Bayesian Network Learning Framework

Chapter 4

Structural Priors: Previous Approaches

Computing the Bayesian score of Eq(3.1) requires specifying both a prior over parameters and a prior over structure. However, as mentioned earlier, the prior over parameters has received greater attention since unlike the structural prior, its influence on the model score grows as the size of the dataset grows [FK03]. Most previous applications of Bayesian network learning have indeed been in domains where there are sufficient samples for the number of variables in the model. In these cases, the choice of structural prior has not had a significant contribution. However, in domains like ours where there are many variables with few samples available, the impact of the structural prior deserves reexamination. In this chapter, we review existing techniques for formulating structural priors before describing how these approaches are not immediately appropriate for our application.

4.1 Types of Structural Priors

Based on the type of information the domain expert provides, the limited number of approaches to structural priors can be grouped into two basic categories: *uninformed* or *informed* priors. We introduce a third category, *semi-informed priors*, for those cases which do not fall neatly into either of the others.

4.1.1 Uninformed Priors

Non-informative, or uninformed, structural priors are those where constraints are applied to limit the space of possible graphs and those constraints are specified without regard to the identity of the particular nodes or edges they affect. The simplest uninformed prior, often chosen for pragmatic reasons, is to assume a uniform prior over the space of graphs. In this case, the Bayesian scoring metric reduces to maximizing the marginal likelihood of the data and for data-sufficient applications, this is a reasonable choice.

Another popular uninformed prior is to limit the number of parents allowed for a node, *i.e.,* specify a *max fan-in* to restrict the size of the families. A uniform prior is then assumed over the remaining DAGs. This restriction is justified by the fact that there are few applications with sufficient data available to robustly estimate the parameters for large families. Moreover, large families tend to have lower scores given almost any scoring metric [FK03].

In some applications, it might be the case that some nodes must be allowed more parents than others in order to accurately capture the corresponding relationships. In this situation, applying the same max fan-in constraint to all nodes is not the best answer since it would be governed by the largest family necessary. An alternative uninformed prior is to impose a global constraint on the graph, where the number of parents per node is allowed to vary as appropriate while the total number of edges in the graph must remain within a specified range. A further alternative is to combine a max fan-in with a global range constraint, as is done in our experiments of Chapter 7.

Motivated by the observation that larger families have lower scores, another option is to use an uninformed prior which penalizes for the complexity of the graph. One example is based on the *Minimum Description Length (MDL)* encoding of the graph [FG96]:

$$Pr(S) \propto 2^{DL(S)}.$$

The description length $DL(S)$ of the graph S is given as:

$$DL(S) = \sum_{i=1}^{N} \left(\log n + \log \left(\begin{array}{c} n \\ k_i \end{array} \right) \right)$$

where k_i is the number of parents for node X_i. This formula assigns a uniform probability over each of the $\left(\begin{array}{c} n \\ k_i \end{array} \right)$ possible families of k_i parents. Consequently, larger families automatically receive lower probabilities. With the MDL structural prior, each graph is penalized by the complexity of its encoding, implementing a bias toward smaller structures. Note that this is an uninformed prior since the identity of the node is considered only in that its indegree is known.

4.1.2 Semi-informed Priors

A second category of structural priors, the semi-informed priors, are those where the domain expert provides explicit information about nodes but not a full characterization of the relationships between all nodes. One example is in enforcing or prohibiting particular edges between nodes when a certain relationship is known to exist or known to be impossible between the corresponding variables. Perhaps the most prominent example of this category is when the domain expert can provide an ordering on variables, perhaps through knowledge of a time precedence. A Bayesian network can always be found for some (total) ordering, by restricting the set of parents for any node to those nodes appearing earlier in the ordering. This greatly reduces the search space of graphs during learning. Note that while the ordering provided by the domain expert specifies what edges are prohibited, it does not include a preference over the remaining edges that respect the ordering. Thus,

the ordering does not explicitly require that particular relationships must absolutely exist among a set of variables.

4.1.3 Informed Priors

To explicitly capture the probability that a certain family exists, we need the third category, the informative structural priors. All of the existing approaches to informed priors make assumptions which allow them to avoid explicitly enumerating the prior over a super-exponential number of structures. The most common requirement is one of *structural modularity* where the prior is a product over the distribution $f(\pi_i)$ of potential parent sets for each variable X_i:

$$Pr(S) = \prod_{i=1}^{N} f(\pi_i).$$

The product captures the assumption that the choice of parents for the variables is independent [FK03]. Further independence assumptions might be included in specific formulations of the structural prior. We consider several examples in the next section.

4.2 Informed Structural Priors: Previous Approaches

Current literature offers few examples of informed structural priors. To the best of our knowledge, the alternatives described in this section represent the entire collection of methods available for forming priors over structures. In the following, we adopt the convention that calligraphic uppercase letters denote information provided by a (human) domain expert.

4.2.1 Variable Parameter-centric Approach

The first approach for informative structural priors is given by Buntine [Bun91] who employs two simplifying assumptions. The first assumption is that the domain expert provides an ordering on the variables, as discussed above. The second assumption requires that the probabilities of edges are mutually independent, *i.e.,* the probability that an arc exists in the graph is independent of the probability of any other arc. Combined with the first assumption, this requires the domain expert to only assess $N(N-1)/2$ probabilities for a network of N variables.

Given a total ordering $\mathcal{O}$ over variables and the set of probabilities $\mathcal{E} = Pr(e_{ij} \mid \mathcal{O})$ that there exists an edge e_{ij} from node X_i to a node X_j later in the ordering (*i.e.,* X_i is a parent of X_j), the prior probability of any structure can be computed as:

$$Pr(S \mid \mathcal{O}, \mathcal{E}) = \prod_{j=1}^{N} \left(\prod_{\{i \mid i<j, e_{ij} \in S\}} Pr(e_{ij} \mid \mathcal{O}) \right) \cdot \left(\prod_{\{i \mid i<j, e_{ij} \notin S\}} (1 - Pr(e_{ij} \mid \mathcal{O})) \right). \tag{4.1}$$

The first term includes all the parents X_i of a node X_j while the second term considers the nodes earlier in the ordering not chosen as parents of X_j. The ordering ensures that this probability

distribution is well defined over the space of permissible DAGs. Given that the information solicited from the domain expert involves an ordering and probabilities specified over *variables*, we refer to this approach as the *variable parameter-centric approach.*

Buntine also suggests that this model could be extended to include correlations over parents, an approach adopted by Riggelsen [Rig02]. The prior probabilities are given over subsets of edges or *chunks* of the graph such that the set of edges contained within each chunk is disjoint from the edge set of any other chunk. The union of the chunks defines the set of DAGs permissible during search. The prior over any structure can be computed by an appropriate generalization of Eq(4.1) and the Buntine approach can be seen as a special case where all chunks are singleton edge sets.

4.2.2 Structure-centric Approach

In contrast to the Buntine approach which requires the domain expert to assess parameters, in some domains it might be easier or more natural for the domain expert to consider more qualitative information such as which variables might influence which others. As such, Heckerman *et al.* [HGC95] adopt what we refer to as a *structure-centric approach* which requires the domain expert to specify a prior network structure.

Candidate structures during search are scored based on how many edges differ from the prior network structure. The strength of the penalty is supplied by the domain expert. Let k denote a constant edge penalty such that $0 < k \leq 1$ and let d be the symmetric difference between edge sets defined by the candidate network S and the prior network $\mathcal{P}$. Then the prior over network structures is calculated as:

$$Pr(S \mid \mathcal{P}) \propto k^d. \tag{4.2}$$

Note that this formula, like the Buntine approach, assumes mutual independence of edges and that the penalty is applied without regard to the identity of the edge. Enforcing edge-specific penalties reduces to the Buntine formula of Eq(4.1).

4.2.3 Data-centric Approach

Madigan *et al.* [MGR95] remark that rather than ask a domain expert to express knowledge about unobservable quantities like parameters or structure, it is easier for domain experts to think in terms of observable quantities, *i.e.,* value assignments to the variables. For this reason, we refer to their approach as the *data-centric approach.*

Their idea is to start with a uniform prior $Pr(S)$ over structures. Imaginary datacases are then created by the domain expert who is repeatedly presented partial (random) assignments to the set of variables and is prompted to estimate the values for the remainder given that partial assignment. To avoid having the domain expert fall into a pattern of answering, the procedure for partial assignment alternates between first randomly choosing a variable (and then randomly setting its value) and then choosing another variable at random for the domain expert to determine

its value. A complete datacase is obtained by continuing the process until all variables are assigned values.

The imaginary database $\mathcal{I}$ is then used to update the prior distribution, as in Eq(3.1):

$$Pr(S \mid \mathcal{I}) \propto Pr(S)Pr(\mathcal{I} \mid S).$$

Finally, this updated prior is used as the prior distribution during learning when scoring a structure against the real data:

$$Pr(S \mid D, \mathcal{I}) \propto Pr(S \mid \mathcal{I})Pr(D \mid S). \tag{4.3}$$

Note that unlike the two previous approaches, this method for computing structural priors does not make the assumption of mutual independence of edges.

4.2.4 Edge Parameter-centric Approach

Murphy [Mur01] introduces an alternative structural prior which concentrates on soliciting probabilities about *edge orientation* from the domain expert, thus we refer to this approach as the *edge parameter-centric approach*. The idea is essentially the same as the variable parameter-centric approach of Buntine, without the variable order assumption. However the edge-centric formulation nonetheless provides a useful construct for thinking about the probabilities provided by the domain expert, namely the concept of a *probabilistic adjacency matrix*.

Assuming mutual independence of edge probabilities, let $\mathcal{W}$ be a matrix which captures the probability distribution of edges as follows: let $\mathcal{W}_{ij}$ be the prior probability that there is an edge from X_i to X_j, let $\mathcal{W}_{ji}$ similarly denote the prior for an edge from X_j to X_i, and let $1 - (\mathcal{W}_{ij} + \mathcal{W}_{ji})$ be the prior that no edge exists between the nodes. When the probabilities are set to the extremes of 1 and 0, $\mathcal{W}$ becomes an adjacency matrix for a graph (not necessarily a DAG). Thus, $\mathcal{W}$ in general can be thought of as a probabilistic adjacency matrix.

Let e_{ijk} denote an edge between X_i and X_j in one of the three orientations:

$$k \in \{X_i \rightarrow X_j, \quad X_i \leftarrow X_j, \quad X_i \not\leftrightarrow X_j\},$$

where $\not\leftrightarrow$ means no edge. For convenience, we use the notation $\mathcal{W}_{ijk}$ to denote the corresponding prior probabilities computed using $\mathcal{W}$. The formula of Murphy then calculates the weight $w(G)$ of any graph G as the sum over the prior on each edge and its orientation:

$$w(G) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{3} \mathcal{W}_{ijk}. \tag{4.4}$$

Since this formula technically defines a prior over any graph G, the prior over DAGs is defined as the normalized weight

$$Pr(S \mid \mathcal{W}) \ \propto \ w(S),$$

where non-DAGs are given zero probability.

Note that, unlike the other examples of structural priors, this formula uses a sum over quantities. An alternative formulation can be derived which more closely resembles computing a joint probability over independent events (*i.e.,* exhibits structural modularity):

$$Pr(S \mid \mathcal{W}) = \prod_{e_{ijk} \in S} \mathcal{W}_{ijk}. \tag{4.5}$$

The correspondence with the variable parameter-centric approach of Section 4.2.1 thus becomes obvious, yet as we will see in Chapter 5 and Chapter 7, thinking of the information provided by experts instead as an edge-centric matrix can be a very useful pedagogical tool.

4.3 Limitations of Previous Approaches for Our Application

We argued earlier that simple application of Bayesian network learning is not appropriate in this domain since there are too many variables with too few samples, and difficulty in obtaining or assaying biological samples dictates that the availability of samples might never be adequate. We then proposed the use of structural priors to alleviate the sparse data problem. However, the existing approaches for structural priors are neither adequate to our problem. One immediately apparent issue is that the large number of genes and complexity of the interactions makes it difficult for a human expert to provide an ordering of the variables (as in Buntine's variable parameter-centric approach) or to complete partial data (as in Madigan and colleagues' data-centric approach).

The second important issue involves the notion of expert. In biology we do not have a single 'expert' with reliable knowledge over the whole biological domain, as needed for the data-centric approach or the structure-centric approach of Heckerman and his colleagues. Our expert information is in fact a collection of potentially incomplete and noisy biological assays where the reliability of the expert depends on the assay technique. To gain coverage over the whole domain, we must combine the data from each of these sources and any combination technique must consider the relative reliability of each source.

Also, our prior information sources provide almost exclusively *positive* assertions of dependence, which does not allow immediate assessment of the probability of a non-edge as in Murphy's edge parameter-centric approach. In biology, the lack of a positive assertion does not imply a negative assertion. In cases where the assay attempted to measure an interaction but failed to find one may mean that the interaction was studied under the wrong context or fell under the sensitivity level of the assay. A positive assertion may also be missing simply because interaction between a pair was not studied by that assay, as happens necessarily in the low-throughput methods.

In the next chapter, we introduce four methods for combining prior information sources asserting relationships between genes. We borrow concepts from the existing techniques described above yet also address the additional issues presented by our application. To highlight the difference between prior information sources in our domain versus in previous applications, we use the term *expert* to denote individual types of prior sources available in our domain, reserving the term *domain expert* to refer to a human expert.

Chapter 5

Structural Priors: Our Approach

This chapter describes four distinct methods we have developed for creating a consensus likelihood over the existence of an interaction between gene pairs. The consensus likelihood can then be used to induce a prior over structures during Bayesian network learning, to be explored in Chapter 7, or to induce graphs for use in visualization tools, as will be demonstrated in Chapter 8.

With respect to Bayesian network learning, Figure 5.1 depicts the full learning framework where structural prior information is supplied to a combination algorithm. Input to that algorithm is the set of implicit or explicit interactions specified by each of the prior information sources listed in Section 2.2. Additional input might include reliability estimates for each expert as well as individual attributes about the genes, such as cellular location or cell cycle phase. The output of each combination method is a probabilistic adjacency matrix (described in Section 4.2.4) which is used to formulate a prior $Pr(S)$ over network structures. The prior over structures is then fed into the search procedure, along with the expression data samples, to produce a series of high-scoring models.

In the following sections, we introduce four possibilities which can be substituted for the combination method component, presented in order of increasing complexity. Each of these includes a discussion of estimating reliability of the various experts. By reliability, we refer to the probability that a relationship exists if an expert observes it, commonly referred to as the *precision* of the technique. A related concept is *recall* which measures the probability that a method detects a relationship if it exists. In our domain, recall is inappropriate since as discussed in Section 4.3, our experts may only interrogate a subset of relationships and can only supply positive assertions. Thus, we can only judge the quality of an expert on the relationships it observes, not on the relationships it fails to observe.

5.1 Method 1: LinOP

The statistics community contributes a body of literature about combining joint probability distributions [CW99, GZ86]. A *consensus joint probability function* Pr^* combining the beliefs of E experts is any function f of the joint probability distributions Pr_e for each expert e:

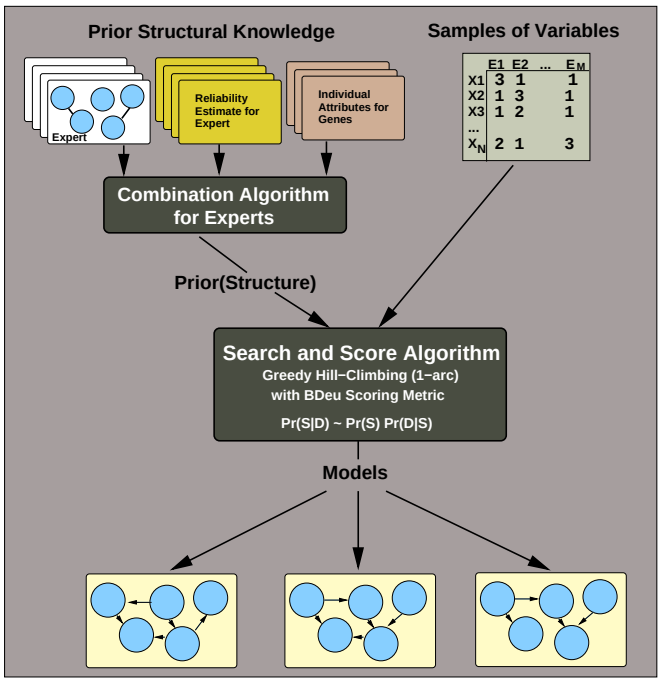


Figure 5.1: Complete Bayesian Network Learning Framework

$$Pr^* \equiv f(Pr_1, \ldots, Pr_E).$$

Typically the function f is designed to exhibit certain desirable properties, such as unanimity (if all agents agree on the probability of all states, the consensus should agree), non-dictatorship (the consensus function does not reduce to the beliefs of a single agent), external marginalization (pooling the opinions first and then computing marginals is equivalent to marginalizing in each individual expert and pooling the result after), as well as other independence properties. A popular consensus function is the *Linear Opinion Pool* or LinOP which satisfies both unanimity and external marginalization:

$$Pr^*(X_1, \ldots, X_N) \equiv \sum_{e=1}^{E} \alpha_e Pr_e(X_1, \ldots, X_N) \tag{5.1}$$

where the α_e are *expert weights* that are non-negative numbers that sum to one.

Since Bayesian networks encode joint probability distributions, a natural extension is to examine the aggregation of beliefs encoded as networks. Pennock and Wellman [PW99] find that even if all experts agree on the same structure, no combination function can preserve the independence assumptions implicit in the structure. On the other hand, Maynard and Chajewka [MC01] argue that requiring the preservation of independence is too restrictive when applied in a learning context. Their aggregation scheme for Bayesian networks uses LinOP and offers a semantic interpretation for the weights α_i as the proportion of the total data seen by each individual expert. As such, they argue

that it is unreasonable to assume each expert will have learned any independencies not justified in the limited data to which the expert had access. In our domain of learning gene interactions, the issue of preserving independence is further complicated by the fact that our experts typically can only observe *dependencies*, thereby offering no judgement about independence.

Though the Maynard and Chajewka approach is motivated by factors in line with our application, their results are not immediately applicable to our problem. They make two critical assumptions about the datacases seen by the expert: that the cases are complete and disjoint from those seen by other experts. Neither assumption is true in our domain. Each of our sources does not necessarily observe every pair of genes and all experts are measuring exactly the same entities. This is unlike the medical domain considered by Maynard and Chajewka where the datacases were measured on different patients.

However, we can make adaptations to LinOP which address two of the key issues in our domain: that our experts measure only a subset of the interactions between genes and that they provide almost exclusively positive assertions. For the first issue of incomplete coverage, we introduce a distinguished 'virtual' expert who supplies a distribution Pr_0 over all pairwise interactions. Like the leak node of the noisy-OR node defined in Section 3.2.4, this expert provides a default belief when no other expert has information. We found that setting the probability of a non-edge to 0.9 (or 0.05 for either directed arc orientation) results in reasonable distributions over interaction, *i.e.,* connectivity of graphs during Bayesian learning. We discuss this point further in Chapter 6.

For the second issue of only positive assertions, note that the lack of a positive assertion does not imply the existence of a negative assertion of interaction. This means that experts with no information about an interaction should not be considered in the sum of Eq(5.1); the expert weight α_e for an expert e should be set to zero for those interactions not asserted by the expert. However, the α_e then no longer sum to 1 as required to define a valid probability distribution. The solution is to generalize the semantics of the expert weights. Let $\mathcal{R}_e$ be a non-zero value that expresses the reliability of expert e. The $\mathcal{R}_e$ will be used in place of the α_e in our adapted version of LinOP.

Recall that the experts are expressing probabilities over pairwise interactions between genes, which we will then use to induce a probability distribution over Bayesian network structures. Following the notation used in Section 4.2.4, we use the probabilistic adjacency matrix representation to encode the probability of edge orientation supplied by each expert as follows. Let e_{ijk} denote an edge between X_i and X_j in one of the three orientations:

$$k \in \{X_i \rightarrow X_j, \;\; X_i \leftarrow X_j, \;\; X_i \not\leftrightarrow X_j\},$$

where $\not\leftrightarrow$ means no edge. Let $\mathcal{W}^e$ be a matrix provided by expert e such that $\mathcal{W}^e_{ij} = Pr(e_{ij1})$, $\mathcal{W}^e_{ji} = Pr(e_{ij2})$ and $(1 - (\mathcal{W}^e_{ij} + \mathcal{W}^e_{ji})) = Pr(e_{ij3})$. For convenience, we use the notation $\mathcal{W}^e_{ijk}$ to denote the corresponding prior probabilities computed using $\mathcal{W}^e$.

Let Ψ_{ij} denote the set of experts making positive assertions about an edge between X_i and X_j. Our adaptation of LinOP for computing the *LinOP consensus likelihood* over edges can then be defined over the set $\mathcal{R}$ of reliabilities and the set $\mathcal{W}$ of probabilistic adjacency matrices provided by

all experts (including $\mathcal{W}^0$ and $\mathcal{R}^0$ provided by the default expert):

$$Pr^*(e_{ijk} \mid \mathcal{R}, \mathcal{W}) \propto \sum_{\{e \in \Psi_{ij}\}} \mathcal{R}_e \mathcal{W}^e_{ijk}.$$

Note that Ψ_{ij} always contains the default expert for every edge (including $k = 3$ for a non-edge). Also, since the $\mathcal{R}$ no longer sum to one, $Pr^*(e_{ijk}|\mathcal{R}, \mathcal{W})$ must be normalized using the values contributing to the sum; this is explained more fully below.

Thinking of the information provided by an expert as an adjacency matrix is a very powerful concept, as alluded to in Section 4.2.4. In our application, each expert asserts a (symmetric) pairwise relationship with full certainty: the assay either found or failed to find an interaction. The only uncertainty comes from the reliability of the assay. Thus for our collection of experts, let $\mathcal{W}^e$ be a (symmetric) deterministic adjacency matrix given by expert e such that $\mathcal{W}^e_{ij} = \mathcal{W}^e_{ji} = 1$ if expert e asserts an interaction between gene X_i and X_j, and 0 otherwise. Note that the default expert has the adjacency matrix of a completely connected graph (excluding self-arcs). The computation of the LinOP consensus likelihood then reduces to simple matrix operations. Let W be the matrix sum calculated as:

$$W = \sum_{e=0}^{E} \frac{\mathcal{R}_e \mathcal{W}^e}{2}$$

The division by 2 accounts for the symmetry of an edge probability. Figure 5.2 illustrates an example calculation of W. Note that this no longer requires distinguishing experts by Ψ_{ij} since experts without information contribute a zero to the sum. This also eliminates the requirement that the $\mathcal{R}_e$ sum to 1, as was the case with the α_e of the original LinOP operator. However, the consequence is that W is merely a matrix sum and does not define a proper probability distribution. To create the probabilistic adjacency representation for the LinOP consensus belief $\mathcal{W}^{\text{LinOP}}$, the sum must be normalized:

$$\mathcal{W}^{\text{LinOP}} = \frac{W}{(1 - \mathcal{R}_0) + W + \texttt{transpose}(W)}, \tag{5.2}$$

where transpose$(\cdot)$ is the familiar matrix transpose function and the division is done componentwise. The diagonal of $\mathcal{W}^{\text{LinOP}}$ is set to zero to disallow self-arcs. Also note that this formula assumes only positive assertions from the experts since the probabilities for a non-edge that appear in the formula is the term $(1-\mathcal{R}_0)$ for the default expert. The denominator would have to be adjusted appropriately if negative assertions were allowed.

The probability of a particular edge orientation can then be extracted from the consensus $\mathcal{W}^{\text{LinOP}}$ in the same manner as described above for extracting from $\mathcal{W}^e$ provided by individual expert e, where $\mathcal{W}^{\text{LinOP}}_{ij} = Pr(e_{ij1})$ and so on. Also, we continue to use the shorthand $\mathcal{W}^{\text{LinOP}}_{ijk}$ to denote the corresponding prior probabilities computed using $\mathcal{W}^{\text{LinOP}}$.

Given the above derivations, the *LinOP consensus likelihood prior over structures*, or LinOP prior, can be calculated as:

$$Pr(S \mid \mathcal{R}, \mathcal{W}^{\text{LinOP}}) = \prod_{e_{ijk} \in S} \mathcal{W}^{\text{LinOP}}_{ijk}. \tag{5.3}$$

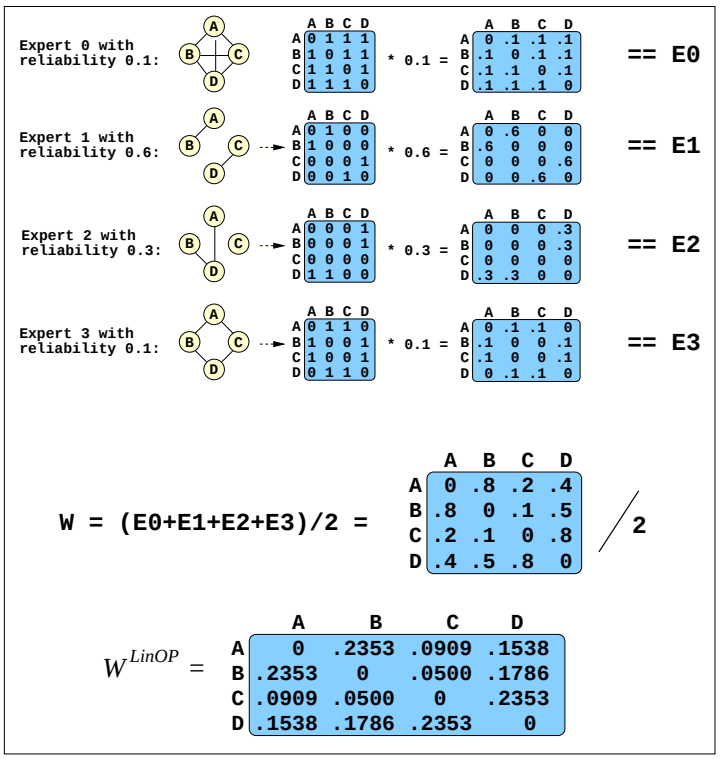


Figure 5.2: Calculating LinOP as matrix addition

Thus far, we have not indicated how to assess the reliabilities $\mathcal{R}$ for each expert; this will be covered later in Section 6.1.2 and Section 6.2.2.

We conclude this section by elaborating on a further connection of our LinOP consensus likelihood with concepts introduced in earlier sections. We can view the LinOP consensus as a method for computing the *maximum a posteriori* (MAP) estimates (discussed in Section 3.2.3) for the probability of an edge, given an imaginary database contributed by our experts, akin to the data-centric approach to priors presented in Section 4.2.3. The idea is to create a virtual database of samples, where each expert e contributes a number of samples for the random variable e_{ijk} proportional to $\mathcal{R}_e$, if the expert measured e_{ijk}. For example, to collect a database of approximately $[100 * \sum_e \mathcal{R}_e]$ samples, expert e would contribute $\texttt{floor}(100 * \mathcal{R}_e)$ samples of each edge it observed. Computing the MAP estimate of the parameter $Pr(e_{ijk})$ then uses the database to compute the sufficient statistics N_{ijk}, *i.e.,* the count for an edge between X_i and X_j with orientation k. Note that for experts with strictly positive assertions, the counts for the no-edge orientation $N_{ij3} = 0$ for all i and j. The virtual counts N'_{ijk} used in the MAP estimate are provided by the default expert who similarly contributes a number of samples proportional to $\mathcal{R}_0$ for all three edge orientations.

5.2 Method 2: Noisy-OR

The LinOP consensus likelihood presented in the previous section was an additive function of the beliefs of the experts. Another option is offered by the work of von Mering *et al.* [vMJS+05] and Nabieva *et al.* [NJA+05] who consider that the probability of an interaction between two genes is multiplicative. The probability of a pair is computed as the noisy-OR of the unreliability of the experts asserting information about that pair. Recall that the noisy-OR function was discussed in Section 3.2.4.

If $\mathcal{R}_e$ is the reliability of expert e, the probability of interaction (or edge) between X_i and X_j is computed as:

$$Pr(e_{ij} \mid \mathcal{R}) = 1 - \prod_{\{e \in \Psi_{ij}\}} (1 - \mathcal{R}_e),$$

where Ψ_{ij} denotes the set of experts making positive assertions about edge e_{ij} and $\mathcal{R}$ denotes the set of reliability assignments, as before. This formula implicitly assumes the experts offer only positive assertions of an edge and do not specify the orientation, in other words that the edge is bi-directional so the probabilities are symmetric across orientation. Note that the reliability $\mathcal{R}_e$ is treated as a probability here and is the same value for all edges witnessed by the expert. Since the $(1-\mathcal{R}_e)$ is the probability of no edge between a pair of genes, the product in the formula computes the consensus probability across experts of no edge. The probability of an edge is one minus that value.

When no expert has witnessed an interaction between two genes, there are no terms over which to apply the product. Therefore, as the formula is given, the probability of an edge with no information is 1. This situation is undesirable since it implies highly connected graphs when the collection of prior sources is sparse. Our solution is to again introduce the 'virtual' expert with reliability $\mathcal{R}_0$ to provide a default probability for all edges. Thus, the product will always include this expert.

We introduce our formulation of a *NoisyOR consensus likelihood* by applying the concept of the probabilistic adjacency matrix representation, as we did for the LinOP consensus likelihood. The probabilistic adjacency matrix $\mathcal{W}^{\text{NoisyOR}}$ representation of the NoisyOR consensus likelihood is computed as:

$$\mathcal{W}^{\text{NoisyOR}} = 1 - \prod_{e=0}^{E} (1 - \mathcal{R}_e \frac{\mathcal{W}^e}{2}), \tag{5.4}$$

where the product is taken componentwise. Note that the adjacency matrix is divided by two to enforce symmetric probabilities for each arc orientation.

The *NoisyOR consensus likelihood prior over structures*, or NoisyOR prior, can then be defined as:

$$Pr(S \mid \mathcal{R}, \mathcal{W}_{\text{NoisyOR}}) = \prod_{e_{ijk} \in S} \mathcal{W}_{ijk}^{\text{NoisyOR}}. \tag{5.5}$$

The NoisyOR prior can use the same strategies as used in LinOP for assessing reliabilities of the experts, as will be discussed in Chapter 6.

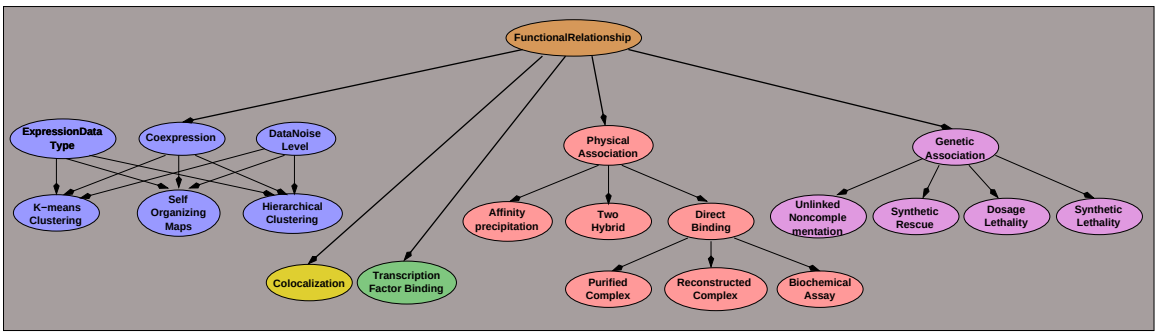


Figure 5.3: MAGIC: Predicting Functional Relationship by a Bayesian Network, from Troyanskaya *et al.* [TDO+03]

5.3 Method 3: BAYES

For both the LinOP and NoisyOR priors, no information about the experts was known beyond the reliability assessments $\mathcal{R}$. There may be cases where the type of expert or the relationship among experts matters. For instance, one may want to distinguish experts that measure an interaction explicitly, like protein binding assays, versus those from whom interactions are merely implied, such as shared sequence motifs. Also, it might be important to separate sources measuring physical interaction from those measuring genetic interactions. To deal with the increased complexity of adding such information, we move from simple functions on adjacency matrices to a representation that can capture both the identities and the relationships among the expert sources: a Bayesian network.

This third approach was motivated by the work of Troyanskaya *et al.* [TDO+03] who used a Bayesian network to integrate multiple sources of gene interaction for the purpose of predicting gene function in yeast. The structure of their network, called MAGIC (Multisource Association of Genes by Integration of Clusters), is shown in Figure 5.3. All variables are binary and set to 1 if the value of the variable is true for a given pair of genes. Note that explicit sources like Yeast Two-Hybrid and Synthetic Lethality are combined with implicit sources like Coexpression and Colocalization. The parameters of the network were solicited directly from yeast genetics experts. For any pair of genes, the corresponding pairwise information was presented at the leaves of the network and the posterior probability of a functional relationship between the pair was calculated at the root node, labelled FunctionalRelationship. Troyanskaya and her colleagues found that predictions from combining the sources more accurately correlated with the Gene Ontology Biological Process groupings than did predictions using gene expression alone.

Recall that in our application, we want to capture a generalized notion of relationship, which we term *semantic relationship*. This term is used to encapsulate the various types of interaction that can be measured, not placing importance of any one source over the other. Figure 5.4 shows the structure of our adaptation of the MAGIC network for the yeast genome, incorporating similar sources to those found in MAGIC. This model was shown previously in Section 2.2 when describing

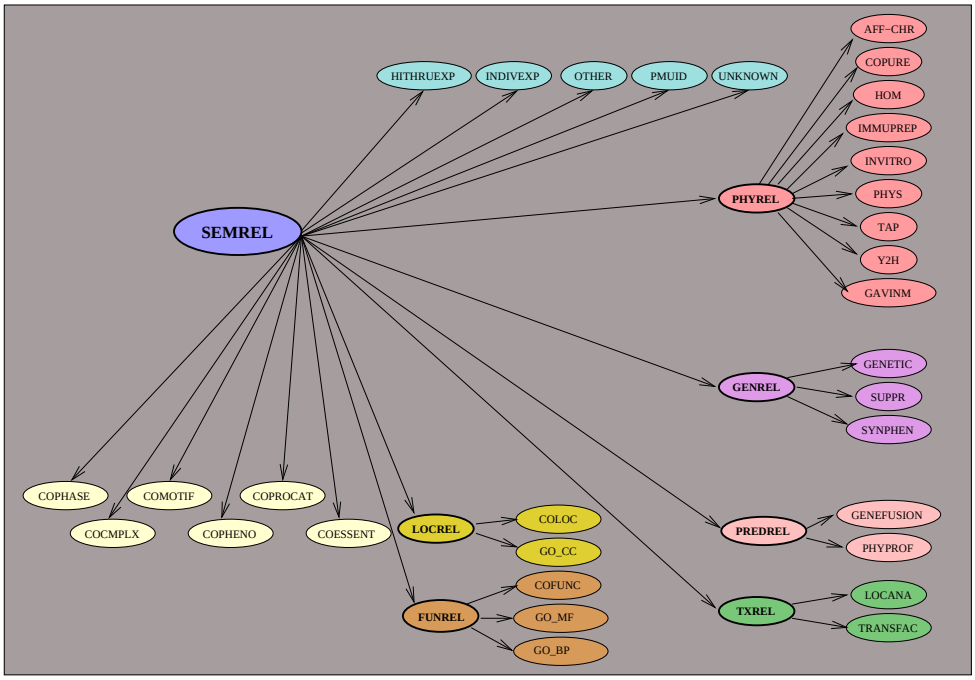


Figure 5.4: BAYES: Predicting Gene Relationship by a Bayesian Network

the interaction sources corresponding to the variables. The model is used to compute a posterior over the probability of a semantic relationship between genes. Since that posterior distribution will later be used as a prior over structures when learning Bayesian Networks from gene expression data, we do not include expression data as a prior source, unlike in Coexpression variable in MAGIC.

Parameters of the MAGIC model were provided by domain experts. Since our model requires the assessment of probabilities over the generalized semantic relationship, a somewhat subjective task, we attempt to learn the parameters from data. In particular, we wish to learn the probability that a semantic relationship between two genes exists, given information about interaction provided by our experts. Since the experts are represented as leaf nodes in the model of Figure 5.4, each datacase in our learning dataset is an assignment to the binary leaf variables for a single pair of genes (interactions are considered symmetric). Note that the interior variables PHYSREL (physical relationship), TXREL (transcriptional relationship), GENREL (genetic relationship), LOCREL (location relationship), FUNCREL (functional relationship), and PREDREL (predicted relationship) describe the type of interaction and are not observed. Moreover, the variable SEMREL (semantic relationship) is also never observed in this setting.

At first glance, learning the parameters of this model given the (incomplete) data appears to be a straightforward task using the techniques discussed in Section 3.2.3. However, the nature of the data available in our domain causes several complications. One issue is the fact that our collection of experts supplying values for the leaves of the model contain almost exclusively positive assertions. A second issue is that the dataset, even for positive assertions, is very sparse. And finally, the fact

that the value of SEMREL is never observed, when coupled with the earlier two issues, means that the learning task is extremely prone to local maxima.

If our data contained observations for all nodes except SEMREL, the learning problem for a model with the structure of Figure 5.4 resembles that of the unsupervised naive Bayes classifier [CH97, WGR+02] where typically EM or a large-sample approximation is used to estimate the parameters of the model. However, our data has a number of hidden variables and even for the leaf nodes, the data is sparse. Moreover, the data contains mostly positive examples.

Suppose our data had some examples of SEMREL set to 1 (true), to signify that some gene pairs are known to be semantically related, but the remainder are left unlabelled. There is a large body of research for classification in a naive Bayes model given only positive and unlabelled examples [DGT02, LLYL02, LL03]. The research is not immediately applicable since these approaches, as earlier, mostly assume the leaf nodes are complete or that the percentage of positive examples is known. In our case, we do not have such an estimate of the number of positive examples, nor can we be completely certain that our positive examples are labelled correctly. However, one previous approach does address the problem similarly to our solution [LLYL02]. Their method initializes the learning by setting the root variable to 0 (false) for a subset of the unlabelled examples that are highly likely to be considered negative. During EM, the positive examples remain positive but the labels of the unlabelled examples are allowed to change. Our approach is similar in that we identify cases that are likely to be negative and initialize EM with values that reflect our belief that gene relationship is relatively rare.

To address the issues of a bias for positive assertions, we augment the data with *negative* assertions where it is reasonable to assume that no interaction exists. For the implicit experts, such as co-location (COLOC) or co-GO Molecular Function category (GO_MF), the corresponding variable for a pair of genes is set to 1 (true) if both genes were assigned a category by the expert and that category was the same for both genes. Since we found that the implicit experts were typically assessed over almost all of the genome (see Section 6.2), we can create negative assertions of interaction by setting the variable to 0 (false) if both genes had assigned categories by an expert but the categories differ. If only one gene has been assigned a category, the variable remains unset. For the explicit experts like Yeast Two-Hybrid (Y2H) and Synthetic Lethality (SYNLETH), the corresponding variable is assigned 1 (true), if an interaction was found by that assay, and left unset otherwise. Note that for the explicit experts, the absence of a measured interaction does not imply no interaction, so there are no assignments of 0 (false) to the explicit experts, only true or missing values. Such a policy for creating negative assertions is a reasonable approximation to the truth.

Even given the dataset augmented with negative assertions from the experts, the sparsity remaining in the data makes the EM algorithm for learning the parameters extremely sensitive to the values used to initialize the parameters. Our solution is to initialize the values in the conditional probability tables to reasonable values, biasing toward non-interaction. For the root node, we initialize the probability that SEMREL is true to 0.1. For the interior nodes, we initialize the probability of a true value when the parent is false to 0.05, to cover noise in the expert data. When the parent

is true, the probability of a true value for the child is initialized to 0.4.

Another alternative is to learn the probabilities when the hidden variable SEMREL has also been set to reasonable values given the data provided by the experts. We set the value of SEMREL for a pair of genes to 1 (true) if more than three experts have made positive assertions of potential interaction between the genes. Requiring at least four experts helps reduce the number of false positives. If more than one but fewer than four experts observe an interaction, SEMREL is left unset. The value of SEMREL is set to 0 (false) if there is at least one expert asserting negative information (as described above) and no other expert asserting a positive interaction. In contrast to the threshold of three for setting SEMREL=1, we allow SEMREL=0 even when only one expert asserts negative information since it must also be the case that there are no positive assertions. We observe that since many of the implicit sources cover the whole genome, the lack of *any* positive assertions for the interaction is favorable evidence toward no interaction. In Chapter 7, we show results for learning for both datasets where SEMREL has been set or not set in this manner.

Given a database over all pairs created by setting the leaves and potentially the root SEMREL according to the set of experts, we learn the parameters of the model in Figure 5.4. We can use the learned model to predict, for any pair, the probability of a semantic relationship. From this prediction we can formulate a prior over structures to be used in the original task of learning Bayesian networks from gene expression data. Unfortunately, we found that the EM algorithm learns a distribution for SEMREL that highly favors interaction a priori (*i.e.,* the marginal probability when no evidence is supplied at the leaves). When later adding (biased and sparse) interaction evidence about a specific pair, this translates to making most pairs also have a high *posterior* probability of interaction. Such a situation means all pairs are extremely likely, regardless of identity or evidence. We attempt to circumvent this problem by instead calculating the probability that a pair interacts as the *relative* change from the prior probability to the posterior probability. This has the effect of using the evidence to normalize the value of the probability. If adding the evidence for a pair does not change the probability of SEMREL from the prior to the posterior, then the probability of interaction for that pair is uniform at 0.5, reflecting the intuition that the consensus likelihood is uninformed for that edge. However, when the evidence for a pair of genes increases the probability of SEMREL, the probability of an interaction (edge) between that pair will also increase. Likewise, if the evidence decreases the probability of SEMREL, the corresponding edge probability decreases.

Let $Pr(\mathsf{SEMREL} = 1)$ be the prior probability that a given pair interacts as inferred from the model learned by the EM algorithm. Let $Pr(\mathsf{SEMREL} = 1|d_{ij})$ be the posterior probability of interaction given the datacase d_{ij} representing the pair X_i and X_j. Then the probability of an edge e_{ij} between X_i and X_j can be computed as

$$Pr(e_{ij}) = \frac{Pr(\mathsf{SEMREL} = 1|d_{ij})}{Pr(\mathsf{SEMREL} = 1)}. \tag{5.6}$$

Assuming symmetry of orientation, the *BAYES consensus likelihood* can be expressed as a probabilistic adjacency matrix $\mathcal{W}^{\text{BAYES}}$ such that:

$$\mathcal{W}^{\text{BAYES}}_{ij} = \mathcal{W}^{\text{BAYES}}_{ji} = \frac{Pr(e_{ij})}{2}.$$

In order to ensure non-zero probabilities, a default value $\mathcal{R}_0$ is added to $Pr(e_{ij})$ and renormalized before creating $\mathcal{W}^{\text{BAYES}}$. The *BAYES consensus likelihood prior over structures*, or BAYES prior, can be defined as:

$$Pr(S|\mathcal{W}^{\text{BAYES}}) = \prod_{e_{ijk} \in S} \mathcal{W}^{\text{BAYES}}_{ijk}. \tag{5.7}$$

Note that unlike the LinOP and NoisyOR priors, the relative reliabilities of the sources can be learned via the corresponding parameters in the model. Reliability is captured as precision in our setting and we can compute the probability that a relationship exists if an expert observes it using inference in the learned BAYES network. In particular, we can calculate $Pr(\mathsf{SEMREL} = 1|X_e = 1)$ for each expert e with the corresponding variable X_e in the model. However, since the learned distribution for SEMREL highly favors interaction even when no evidence is supplied at the leaves, we employ the ratio technique once again and compute reliability of expert e as:

$$\mathcal{R}_e = \frac{Pr(\mathsf{SEMREL} = 1|X_e = 1)}{Pr(\mathsf{SEMREL} = 1)}. \tag{5.8}$$

5.4 Method 4: PRM

In the BAYES prior, a gene participates in $O(N)$ pairs, contributing $O(N)$ samples to the database. Note that the database thus created from the experts considers each pair as a independent sample even though the samples are clearly correlated. Assuming independence has several consequences since not only are interactions learned assuming independent samples, they are also predicted independently. Important information may be revealed by instead studying the possible dependencies between pairs. For example, a particular gene may tend to interact with genes of a certain function, so information about interaction with one of its partners may be informative of potential interaction to other partners. Thus, a better strategy for learning gene relationships would be to choose a representation that allows us to consider interactions for all pairs simultaneously. A similar observation was made by Jaimovich *et al.* [JEMF05] who used an undirected probabilistic graphical model which combined interaction sources, like our explicit experts, with cellular location in order to predict protein-protein interactions. Their modelling framework is essentially the undirected equivalent of the method we present next, though we use more types of expert information and consider the model for a different purpose.

In order to capture the correlations between related gene pairs and leverage the additional information implied by those relations, we consider a combination function based on the *Probabilistic Relational Model* (PRM) [FGKP99]. A PRM can be viewed as a type of template Bayesian network where the structure and parameters of the model are defined on the level of classes and their attributes. The template is then repeated for every instantiation of the classes and links between class types capture the relations inherent in the data.

A PRM is best shown by an example: consider the PRM of Figure 5.5 taken from Getoor *et al.* [GSTK01] for a Web domain. The objective is to predict the category of a webpage based on certain anchor words and links to other webpages. Figure 5.5(a) depicts the class-level specification

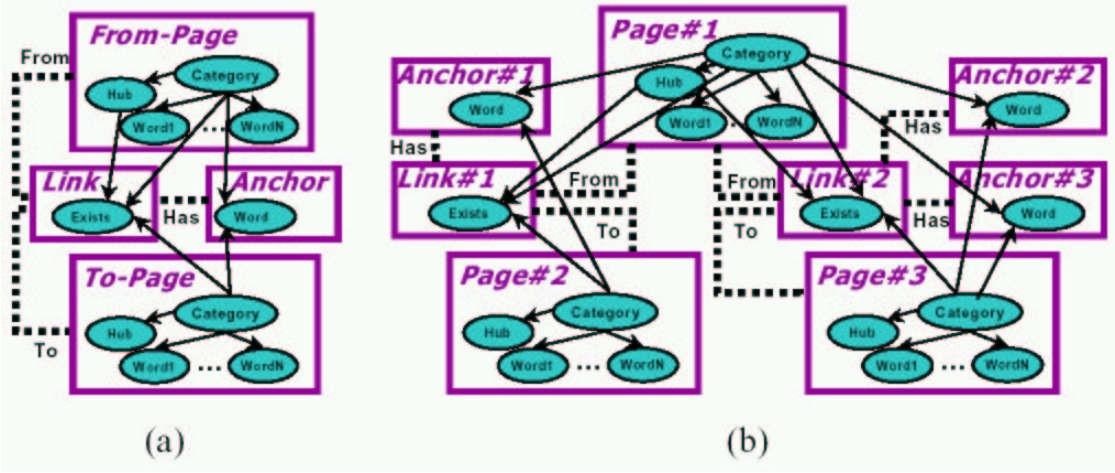


Figure 5.5: (a) PRM schema for WebKB domain; (b) Fragment of unrolled network for WebKB model (taken from Getoor *et al.* [GSTK01])

of the PRM, showing a template for a pair of webpages with a potential link between them. The link may contain certain anchor words. Figure 5.5(b) shows how the template is instantiated for three webpages and the potential links among them. Note how, unlike the BAYES prior, correlations between links involving the same object are not ignored: the two links involving Page1 connect to a single occurrence of that page.

By retaining the identity of entities participating in numerous pairwise relationships, the PRM framework allows inferences to propagate through related objects, capturing the insight that certain pages are more likely to link to certain other pages. For example, a student webpage is more likely to link to a course page or a faculty page, but not vice versa. This affects the corresponding probability of existence of links between those categories. Inferring links between pages is similar to our task where we wish to predict the existence of links between genes based on their individual attributes, such as function or cellular location. Knowing the attributes of the genes affects our prediction since genes tend to interact with other genes of the same function and within the same cellular compartment.

The attraction of PRMs is that the *unrolled* model, an example of which is shown in Figure 5.5(b), is also a Bayesian network and the standard inference and parameter learning algorithms apply. Formally, a PRM defines a template over a relational database where $\mathcal{X} = X_1, \ldots, X_N$ is a set of classes. In the example of Figure 5.5, the set of classes includes the Page, Link, and Anchor classes. Each class X has a set of *descriptive attributes* $\mathcal{A}(X)$ and a set of *reference slots* $\mathcal{R}(X)$. Each of the reference slots are typed so that they can only refer to objects of a particular class. The descriptive attributes of the Page class include the Category of the page, indicators for the Words contained in the page, and an indicator for whether the page is a Hub. Note that the existence of a link is represented explicitly as a random variable Link.Exists. Also, the Link class has a reference slot Has of type Anchor, as well as reference slots From-Page and To-Page, whose type is the class Page.

A PRM *schema* defines the classes, their attributes, and the range types for all the reference slots. A PRM *instantiation* specifies the set of objects of each class, the values of the descriptive attributes and the target of each reference slot for each object. A *relational skeleton* σ denotes a partial instantiation. In the instantiation shown in Figure 5.5(b), there are three objects of the Page class, two Link and three Anchor objects, and each of the attributes is given a particular value (not shown). The reference arcs for Link1 are resolved between Page1 and Page2, and similarly for Link2 between Page1 and Page3. Also note the multiple Anchor objects which are referenced by Link2.Has reference slots.

Given a schema and a skeleton, a *Probabilistic Relational Model* (PRM) specifies a probability distribution over any valid instantiation. Like a Bayesian network, the PRM $P =< S^{\text{PRM}}, \theta^{\text{PRM}} >$ includes a structure component S^{PRM} and a parameter component θ^{PRM}. The dependency structure among attributes is given by associating a set of parents $\pi_{X.A}$ with each attribute $X.A$. In the example, Page.Category is parent to the Page.Word, Page.Hub, and Link.Exists attributes. A conditional probability distribution (CPD) is defined for each attribute given its parents. The probability of Link.Exists is conditional on the values of FromPage.Hub, FromPage.Category and ToPage.Category.

As each class is instantiated, the PRM is effectively unrolled to induce a Bayesian network on the set of attributes. The structural dependencies specified between different classes in the schema must respect the reference structure supplied by the instantiation. For example, in the unrolled network example of Figure 5.5(b), an arc from Page1.Category is replicated for both instances of Link.Exists which reference Page1. Similarly, the arc from Page1.Category and Page3.Category is replicated to both Anchor classes associated with Link2 by following a chain of reference slots.

Viewed as an unrolled network, it is easy to see that the joint probability can be computed by the product rule, as it was for Bayesian networks.[1] Given a skeleton σ, the joint probability distribution over any valid instantiation $\mathcal{I}$ of the PRM $P =< S^{\text{PRM}}, \theta^{\text{PRM}} >$ can be computed as:

$$Pr(\mathcal{I} \mid \sigma, S^{\text{PRM}}, \theta^{\text{PRM}}) = \prod_{X_i} \prod_{A \in \mathcal{A}(X_i)} \left[\prod_{x \in \mathcal{O}^{\sigma}(X_i)} Pr(\mathcal{I}_{x.a} \mid \mathcal{I}_{\pi_{x.a}}) \right],$$

where $\mathcal{O}^{\sigma}(X_i)$ is the set of objects of class X_i specified by $\mathcal{I}$. In a PRM, all object instantiations of a particular class use the same CPD, specified at the class level. This implements a clever use of *parameter tying* during parameter learning: the data for each object's attributes is pooled across instantiations of a class when computing the sufficient statistics to estimate the CPD. The sharing of data is important in applications where there are typically thousands of instantiations of an object yet there is rarely enough data to estimate separate parameters for each object. In the WebKB example, there are potentially millions of pages and links among them, yet parameter tying requires learning only a few CPDs.

To apply the PRM framework to the task of learning a model of gene interactions, consider the example in Figure 5.6 which includes a subset of the possible interaction information types described

[1] Certain restrictions must be placed on allowable values for the reference slots to ensure that the resulting induced Bayesian network remains acyclic. See Friedman *et al.* [FGKP99] for full details.

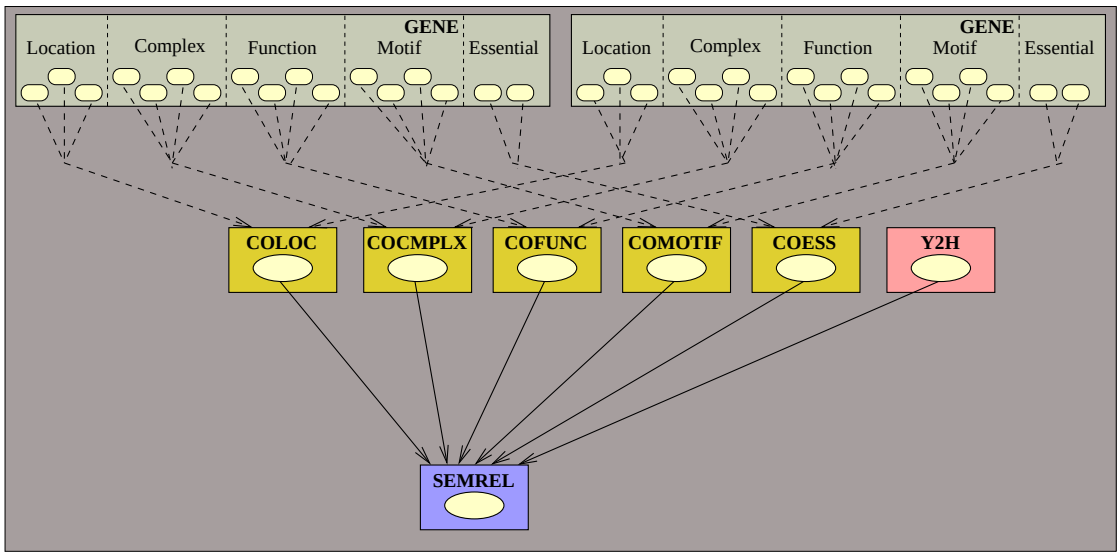


Figure 5.6: PRM schema for Gene Semantic Relationships

in Section 2.2. We define a Gene class which includes a set of attributes describing cellular location, cell cycle phase, structural motifs, function and other properties of the gene which were used to derive implicit interactions (*c.f.* Section 2.2.2). For example, cellular location attributes might include Gene.Cytoplasm, Gene.PlasmaMembrane, and Gene.Nucleus. Typically, each group can have any number of categories so we choose to use binary variables instead of a single variable covering all possibilities. For visual purposes, we do not label the gene attributes individually here, rather we label them in groups based on the attribute type. For example, the two categories Viable and Lethal are shown grouped under the attribute type Essential.

For each expert interaction source, whether implicit or explicit, we include an Interaction class with a single (binary) attribute Interaction.Exists (label not shown). Each interaction class is labelled in Figure 5.6 by the particular expert type. For example, we have the explicit interaction type Y2H and the implicit interaction type COFUNC, both of class Interaction. Note that for explicit interaction types, the Interaction.Exists attribute has no parent, while for the implicit interactions, arcs show the dependence on the particular Gene attributes from which their value derives. Multiple arrows from the same group are fused together for visual clarity. Also, for nodes with only one attribute, arcs from parents of that attribute point to the class, not fully connecting with the attribute, simply to avoid clutter in the figure.

A straightforward representation of the CPD for an Interaction.Exists variable is exponential in the number of parents. For example, co-location might be described by 32 different binary cellular location Gene attributes for each gene, resulting in 2^{64} possible parent configurations for the COLOC.Exists variable. To avoid estimating an unfeasible number of parameters for the derived interaction classes, we take advantage of the special function these nodes are encoding, namely that an implicit interaction exists between the two genes if they are assigned to the same category. Thus,

we fix the parameters of those nodes to a deterministic function of the number of matched categories. The function resembles a *noisy-OR* (Section 3.2.4), yet respects that each gene contributes a matching set of attributes as parents. Using this function, the more categories that match, the higher the probability of the implicit interaction.

If a derived interaction relies on a group of P attributes from each gene, let C be the set of categories in common between the two genes. For example, suppose the co-location of a gene is represented by a group of five Gene attributes ($P = 5$). For a pair of genes, Gene1 and Gene2, if Gene1.Nucleus, Gene1.PlasmaMembrane and Gene1.Cytoplasm are all true, and Gene2.Nucleus, Gene2.Cytoplasm and Gene2.ER are true, then $C = \{$Gene2.Nucleus, Gene2.Cytoplasm$\}$ for the attribute $\mathsf{COLOC}_{1,2}$.Exists. For the case where no categories match, we provide a default inhibition probability q_0 (set to a high value like 0.9) which acts like a category that always matches. An inhibition probability q_p applies when category p matches, and for convenience is set to the same value for all categories (typically a low value like 0.1). Given the inhibition probabilities, the probability of Interaction.Exists for derived interactions can be calculated as:

$$Pr(\mathsf{Interaction.Exists} = 0 \mid \pi_{\mathsf{Interaction.Exists}}) = \prod_{p \in C} q_p,$$

and

$$Pr(\mathsf{Interaction.Exists} = 1 \mid \pi_{\mathsf{Interaction.Exists}}) = 1 - \prod_{p \in C} q_p.$$

Recall that C always contains a category for the leak node. The CPD is essentially a noisy-OR, not of the parents, but of the matching categories. The parameters for this class of attributes are thus fixed and do not change when learning the model using EM.

To capture whether two genes have a semantic relationship, we introduce the class SEMREL with one attribute, SEMREL.Exists (label not shown). Note that if the Gene classes are removed from Figure 5.6, the schema resembles the BAYES model of Figure 5.4 with the arcs reversed. Again, the large number of parents makes a straightforward representation of the CPD for SEMREL.Exists infeasible, so we implement the probability distribution as a noisy-OR. Unlike the fixed parameters of the Interaction.Exists attribute, we wish to learn the inhibition probability for each parent here since it corresponds to the reliability of the corresponding expert. Recall that learning the parameters of a noisy-OR using EM makes use of a structural decomposition, introducing a leak node and a set of hidden nodes Y_i', one for each parent Y_i. The augmented PRM schema is shown in Figure 5.7. The blue nodes whose attributes are parents of SEMREL are the nodes introduced to aid learning of the noisy-OR parameters.

Given the PRM schema of Figure 5.7, an instantiation of the model consists of a set of genes, their individual attributes, and a set of interactions specified by expert information. Replicating the schema for each pair of genes induces a Bayesian network, as shown in Figure 5.8 for three genes. Note that although there are a quadratic number of replications of a rather sizeable schema, there are very few parameters to estimate in reality, given parameter tying and the CPDs for the Interaction.Exists and SEMREL.Exists nodes defined as above.

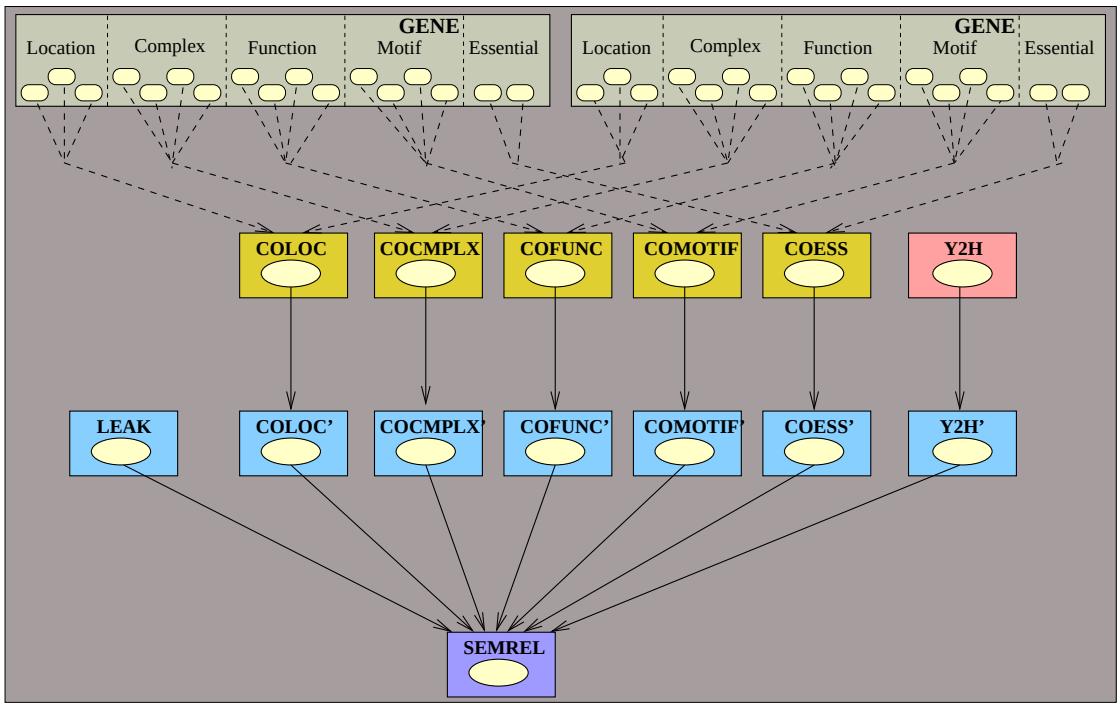


Figure 5.7: PRM schema for Gene Semantic Relationships (with Noisy-OR)

In estimating the parameters, we use the same techniques as for the BAYES consensus likelihood function. We first augment the data with negative assertions for each implicit interaction expert, as discussed previously. With this additional data, we can then create a dataset where the value of SEMREL.Exists is either set or not, based on the number of (original) parents of the noisy-OR asserting negative or positive information as before.

Given the noisy-OR model for the CPD of SEMREL.Exists, note that the only circumstance where an interaction is unlikely is the case where all parents are false. This means that similar to the SEMREL distribution learned for the BAYES model, the marginal probability of SEMREL.Exists with no evidence is nearly 1. Since this implies highly connected graphs when we formulate a structural prior from the model, we also employ the ratio technique to scale the probability.

Let $Pr(\mathsf{SEMREL}_{ij}.\mathsf{Exists} = 1)$ be the marginal probability that a given pair X_i and X_j interacts given no evidence, computed using the parameters learned by the EM algorithm. Let $Pr(\mathsf{SEMREL}_{ij}.\mathsf{Exists} = 1|D)$ be the posterior probability of the interaction given the database D. Note that in a PRM, all pairs are predicted simultaneously. The probability of a particular edge e_{ij} can be computed as

$$Pr(e_{ij}) = \frac{Pr(\mathsf{SEMREL}_{ij}.\mathsf{Exists} = 1|D)}{Pr(\mathsf{SEMREL}_{ij}.\mathsf{Exists} = 1)}. \tag{5.9}$$

Assuming symmetry of orientation, the *PRM consensus likelihood* can be expressed as a probabilistic adjacency matrix $\mathcal{W}^{\text{PRM}}$ such that:

$$\mathcal{W}^{\text{PRM}}_{ij} = \mathcal{W}^{\text{PRM}}_{ji} = \frac{Pr(e_{ij})}{2}.$$

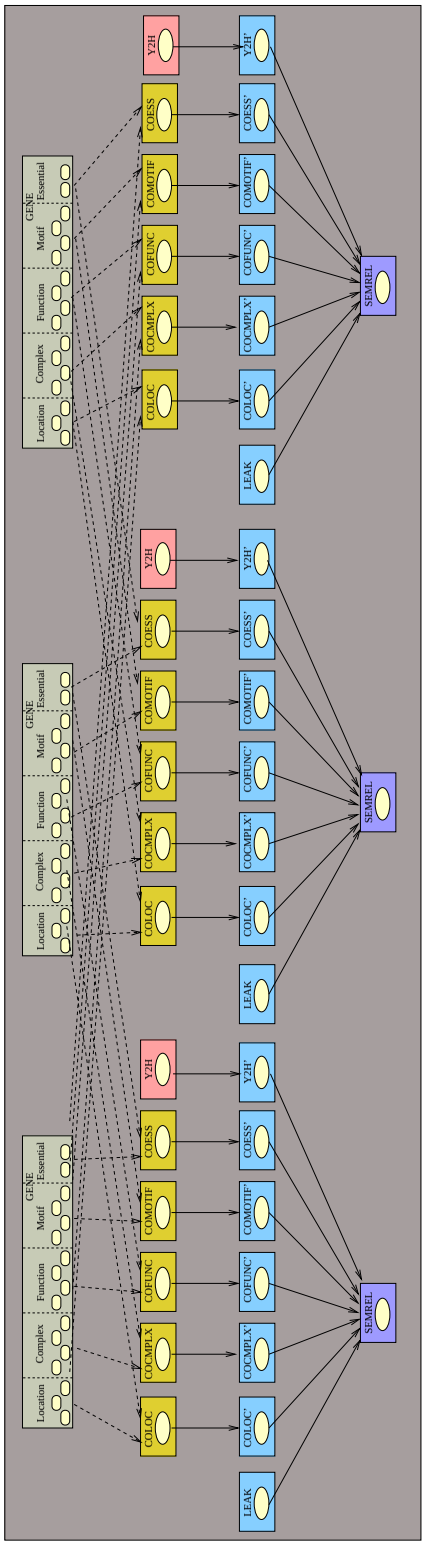


Figure 5.8: Unrolled Model for Gene Semantic Relationships

As with BAYES, in order to ensure non-zero probabilities, a default value $\mathcal{R}_0$ is added to $Pr(e_{ij})$ and renormalized before creating $\mathcal{W}^{\text{PRM}}$. The *PRM consensus likelihood prior over structures*, or PRM prior, can be defined as:

$$Pr(S|\mathcal{W}^{\text{PRM}}) = \prod_{e_{ijk} \in S} \mathcal{W}^{\text{PRM}}_{ijk}. \tag{5.10}$$

Like the BAYES prior, the relative reliabilities of the sources can be extracted from performing inference in the model. Since the parameters are shared across instantiations of genes, to compute probabilities for a pair in general, we first create a smaller PRM containing two genes and all pairwise experts, essentially just the schema shown in Figure 5.6. Associated with this structure is the set of parameters learned by EM for the original PRM. We can then use inference to compute $Pr(\mathsf{SEMREL.Exists} = 1|\mathsf{Interaction}_e\mathsf{.Exists} = 1)$ for each expert e with the corresponding variable $\mathsf{Interaction}_e\mathsf{.Exists}$. Since $Pr(\mathsf{SEMREL.Exists} = 1)$ is typically very high even with no evidence, we compute the expert reliabilities using the ratio technique:

$$\mathcal{R}_e = \frac{Pr(\mathsf{SEMREL.Exists} = 1|\mathsf{Interaction}_e\mathsf{.Exists} = 1)}{Pr(\mathsf{SEMREL.Exists} = 1)}. \tag{5.11}$$

Chapter 6

Computing Consensus Likelihood Functions

In this chapter, we apply the four methods LinOP, NoisyOR, BAYES and PRM of Chapter 5 for computing consensus likelihoods of gene interaction to interaction data from the yeast and mouse genomes. We also include an example from the real-world model of ICU ventilator management (ALARM) [BSCC89], a well-known benchmark for Bayesian network learning. Since the ALARM model includes a target graph structure, we use it to investigate the effect of estimating reliability. We create experts of varying reliability by controlling the level of deviation of the set of edges suggested by a given expert from the actual structure of the target model. Evaluating the ability to correctly estimate reliability and investigating the robustness of the consensus likelihood functions to errors in those estimates are critical issues in our biological domain where it is often unclear how to determine the reliability of any technique for assaying interaction between genes. In the next section, we present results from the ALARM study before detailing the application of the consensus likelihood methods on the real genomes in Section 6.2 and Section 6.3.

6.1 Consensus Likelihood for ALARM

The ALARM model is a real-world example of a Bayesian network designed to represent a medical diagnostic alarm message system for ICU patient monitoring [BSCC89]. The Bayesian network structure over 37 variables with 46 directed arcs is shown in Figure 6.1. Since we do not have independent information about individual variable attributes, as required by the BAYES and PRM consensus likelihood functions for genomic domains, we can only apply LinOP and NoisyOR here (Section 6.1.3). Both functions require a set of experts and an estimate of the reliabilities of those experts. We discuss each in turn in the next two sections.

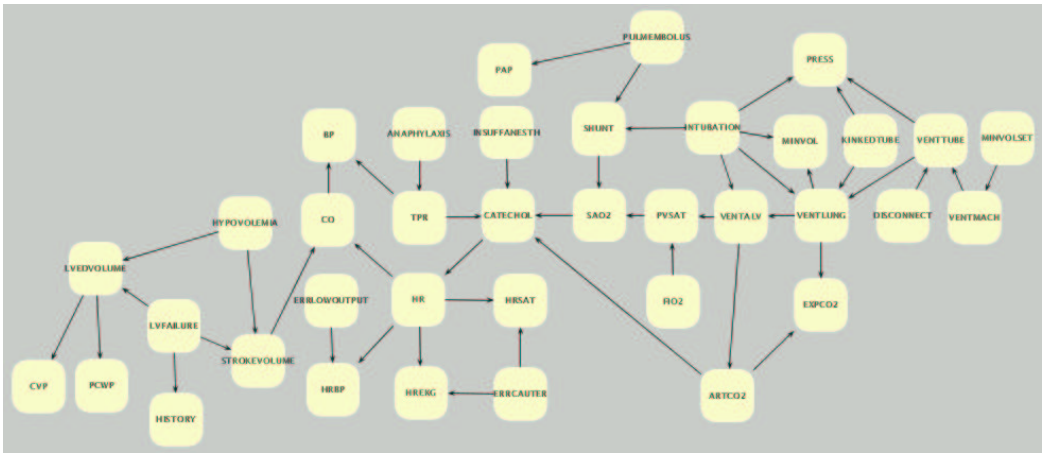


Figure 6.1: Graph Structure of the ALARM network [BSCC89]

6.1.1 ALARM Experts

To emulate the type of data available in the genomic domains we consider later in this chapter, we create a set of experts that vary in reliability with respect to the ability to correctly identify dependencies between variables. We create three classes of experts, denoted Positive, Negative, and Indirect, to capture the fact that biological interaction data sources can either correctly identify dependencies, incorrectly identify dependencies, or correctly identify indirect dependencies, respectively. Though a single real biological source typically contains a mixture of true and false positives as well as direct and indirect interactions, the consensus functions of LinOP and NoisyOR consolidate the set of sources so the origin of error for the edge is irrelevant in the final product. Thus, we can separate the experts into our three distinct classes of unreliability, without loss of generalization. To mimic the differences in scale and density of our real biological sources, we vary the number of edges asserted by each type of ALARM expert.

As representatives of interaction sources asserting correct dependencies, we create a set of 20 Positive experts, each asserting between 1 and 10 (directed) edges that appear in the original graph. The distribution of the number of edges asserted per expert is given in Table 6.1. For example, in the top row there are three Positive experts with eight edges asserted and zero Positive experts with six edges asserted. The union of edges present in the 20 Positive experts cover 42 of the 46 edges in the target model, as shown in the final column of Table 6.1.

False positive interactions are introduced through a set of 20 Negative experts, each asserting between 1 and 10 (directed) edges *not* in the original graph, neither in the given orientation nor the reverse orientation. As can be seen from Table 6.1, the union of Negative experts contains 109 false positive edges. The proportionately greater total number of Negative edges mimics the high false positive rates apparent in real biological sources.

Compared to the Positive and Negative experts, it is more difficult to simulate Indirect interaction sources since many reasons for indirection exist in the biological interaction information. They

# Edges:	1	2	3	4	5	6	7	8	9	10	Edges in Union
Positive Experts	2	4	1	4	2	0	1	3	1	2	42
Negative Experts	1	1	4	2	1	0	5	2	3	1	109
Indirect Experts	3	3	4	2	4	2	0	2			52

Table 6.1: Distribution of number of edges per expert for ALARM

may not be easily factored into recognizable causes, thereby prohibiting accurate representation in our simulated experts. However, we can elaborate on the finding that proteins with pathlengths of two physical interactions in an interaction network are often connected by a single genetic interaction [BCRC04]. For the ALARM experts, we create a set of 20 Indirect experts, half of which assert from 1 to 8 edges between variables that are separated by an undirected path of length 2 in the original graph, while the other half similarly assert from 1 to 5 edges between variables separated by an undirected path of length 3. The final row of Table 6.1 reflects this choice of between 1 and 8 edges overall.

In computing LinOP and NoisyOR consensus likelihood functions, we use the adjacency matrix representation for the experts. Since we assume in this thesis that the experts are indifferent to the orientation of an edge, the matrices are made symmetric to orientation. Figure 6.2 shows a summary of the edges asserted by the different classes of experts. The adjacency matrix for the undirected equivalent of the original ALARM model is shown in the upper left, where blue squares indicate a 0 in the adjacency matrix, while light blue indicates a 1, *i.e.,* an edge. Note that the union of the Positive experts, shown on the upper right, does indeed replicate the strong crossbow-like pattern evident in the matrix for the original graph. A slightly more diffuse version of the crossbow shape can be seen in the matrix for the union of Indirect experts, shown on the lower left. However, when viewing the matrix for the union on Negative experts, the crossbow pattern disappears.

6.1.2 ALARM Reliability Assignments

Since we have access to the correct structure of the model for ALARM and know the types of errors made by our experts, we can investigate the effect of incorrectly assigning reliability to those experts. To do this, we create a set of reliability assignments based on whether a particular class of expert, Negative, Indirect or Positive, is given a Low, Medium, or High reliability assignment. Reliability values range from 0 to 100 such that a reliability of 0 to 33 indicates Low reliability, 34 to 66 indicates Medium reliability while 66 to 100 indicates High reliability. A reliability value $\mathcal{R}_e$ for each of the 60 experts e is chosen randomly from the appropriate range according to the reliability level chosen for the class of the expert.

Five different combinations of reliability assignments are considered. These combinations are referred to by a three integer string N.I.P where N, I, and P represent the reliability levels for the set of Negative, Indirect, and Positive experts, respectively. In the integer string, a value of 0 indicates Low reliability, a value of 1 indicates Medium reliability, and a value of 2 indicates High reliability. The particular combinations used in this thesis are 0.0.2, 0.1.2, 0.2.0, 1.1.1 and 2.1.0, in roughly

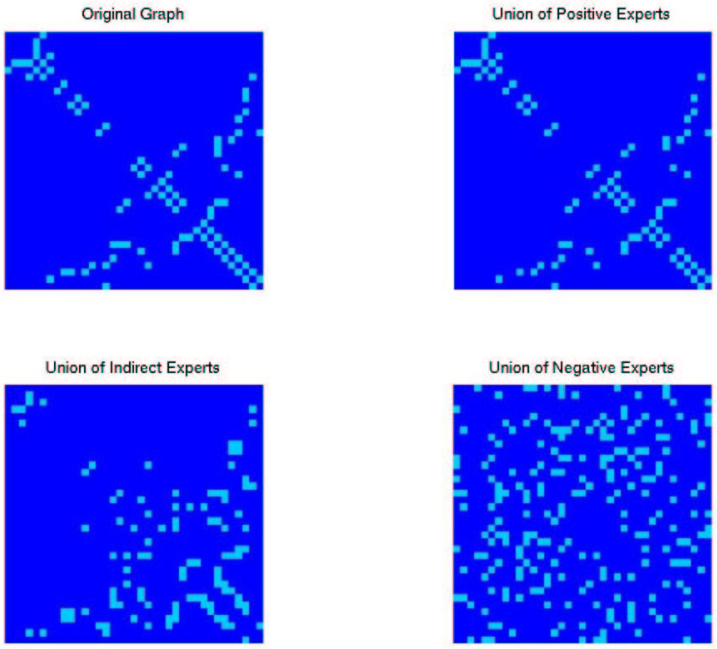


Figure 6.2: Adjacency Matrices for ALARM experts

.

decreasing order of correctness. The first combination N.I.P=0.0.2 considers the case where only Positive experts are correctly assigned High reliabilities. The case 0.1.2 represents the intuitive classification of the experts, namely that the Indirect experts might provide more useful information than the Negative experts. The third case 0.2.0 further investigates the influence from the Indirect experts by assigning High reliability only to them. The combination 1.1.1 considers the case where experts are uniformly assigned Medium reliability. This case is significant since it can be used to quantify the importance of taking into account the varying reliability of our information sources, a key claim in this thesis. The final case 2.1.0 considers the extreme case where all experts are assigned reliabilities completely opposite to their known correctness. This case is particularly important to our study since it can be used to qualify the robustness of our consensus likelihood techniques to grossly incorrect assignments of reliability.

6.1.3 Computing ALARM Consensus Likelihoods

The LinOP and NoisyOR consensus likelihoods are calculated using each of the five reliability assignment combinations described in the previous section. Recall that for these two functions, the probabilistic adjacency matrices $\mathcal{W}^{\text{LinOP}}$ (Eq (5.2)) and $\mathcal{W}^{\text{NoisyOR}}$ (Eq (5.4)) are calculated as a function of the deterministic adjacency matrices $\mathcal{W}^e$ and reliability assignments $\mathcal{R}_e$ for each expert e. Both employ a default expert with the completely connected adjacency matrix $\mathcal{W}^e$ (minus self-arcs) and a reliability assignment $\mathcal{R}_0$ which enforces a default probability for an edge without any information provided by the experts. Experimenting with several values for the default probability,

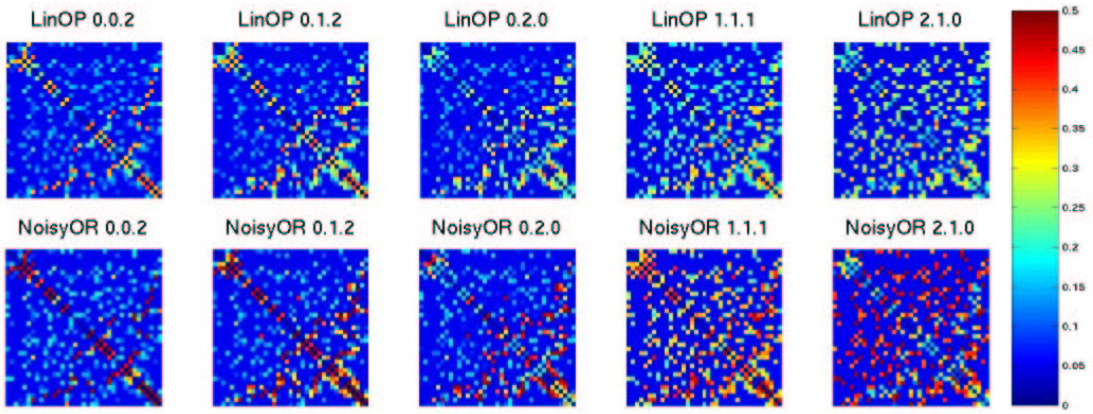


Figure 6.3: Consensus Likelihood Matrices for ALARM

.

we found a value of $\mathcal{R}_0 = 0.1$ to be reasonable, in that graphs sampled from the prior over structures based on consensus likelihoods using 0.1 tend to have a similar number of edges to the target graph (data not shown). Consequently we use the value $\mathcal{R}_0 = 0.1$ in all of our subsequent results for LinOP and NoisyOR.

Figure 6.3 shows the results using each of the five reliability assignments to create probabilistic adjacency matrices for the LinOP consensus likelihood $\mathcal{W}^{\text{LinOP}}$ (top row) and the NoisyOR consensus likelihood $\mathcal{W}^{\text{NoisyOR}}$ (bottom row). The five combinations, namely 0.0.2, 0.1.2, 0.2.0, 1.1.1, and 2.1.0, are arranged left to right in order of (roughly) decreasing correctness of reliability assignment. The colorbar on the right of the figure indicates the value of the probability, where the *hotter* the color, the *higher* the probability. Dark blue indicates zero likelihood for an edge (as can be seen along any diagonal) while dark red indicates a directed edge exists with probability 0.5. Note that since our experts ignore orientation of an edge, a value of 0.5 is actually asserting that an undirected edge exists with probability 1.0 since that probability mass is split evenly across the two orientations.

The first observation from the figure is that the crossbow pattern of the original dag can still be seen in many of the matrices. However, as the reliability assignments become less correct, the crossbow pattern disappears (see right most column). Also immediately apparent from the figure is the fact that both rows have similar patterns. This is a consequence of the fact that the edges asserted by an expert do not change, only their reliabilities. Note however that the NoisyOR results are generally more red than the LinOP results. This reflects the fact that as a multiplicative function, NoisyOR is assigning higher probabilities to values than the additive function LinOP. Thus, when an expert correctly identifies a dependency, the NoisyOR probability will be greater than the LinOP probability for the same edge. However, this has important consequences for the consensus functions since NoisyOR's tendency toward higher probability also makes it more susceptible to incorrect reliability assignments than LinOP. Note the highly red off-diagonal elements in the upper left corner of the NoisyOR 2.1.0 matrix versus the cooler blues of the corresponding LinOP matrix.

The side-by-side comparison of results as shown in Figure 6.3 provides an immediate visual

assessment of the performance of the operators. Already, we can make general statements about the quality of the consensus likelihood distributions. Further judgements are made in Section 7.2 where we quantitate the difference between the operators by evaluating their ability to aid Bayesian network learning when used as a structural prior.

6.2 Consensus Likelihood in Yeast

In terms of available data, the yeast genome is currently the most well studied. As of April 2006, the Saccharomyces cerevisiae Genome Database (SGD) [CAB+98] reports 6603 potential genes, called *Open Reading Frames (ORFs)*, of which 4384 have been verified experimentally. There are 822 dubious ORFs while 1397 remain uncharacterized. Though the uncharacterized ORFs have unknown function, there are often experimentally measured interactions or even computationally predicted relationships for these ORFs based on their known DNA sequences. Creating a consensus likelihood of interaction from such data is valuable because it reveals potential neighbors, thereby providing a semantic context for annotating the unknown genes.

Each assay for measuring or predicting interaction becomes an expert in the yeast domain. However, unlike the ALARM example, we do not know the true set of relationships between variables (genes) in the model. Thus, we do not have a gold standard for assigning reliability and must therefore devise ways to estimate reliability. In the next two sections, we list the set of sources used to derive experts and present several strategies for assigning reliability to those experts. We also list data sources contributing individual attributes of the genes. In Section 6.2.3, we discuss application of the LinOP, NoisyOR, BAYES and PRM consensus likelihood functions to these data.

6.2.1 Yeast Data and Experts

Two types of data are used in the yeast domain to formulate consensus likelihoods. Sources with experimentally measured or computationally predicted interactions form the set of *explicit* experts, as discussed in Section 2.2.1, which assert interactions between pairs of genes. Sources with information about individual gene attributes become *implicit* experts, as discussed in Section 2.2.2, which derive an interaction between two genes based on whether the genes have values in common for those attributes.

Yeast Explicit Experts

To form the explicit experts, we extract pairwise interactions from the following databases, many of which were discussed in Section 2.2.1: BIND [BDW+01], BRITE [KKK97, MFG+02], DIP [XSD+02], MIPS PPI catalog [MFG+02], PREDICTOME [MYC+02] and TRANSFAC [WCH+00]. We also included pairwise interactions from a number of seminal or comprehensive interaction papers [DSB+01, EIKO99, GBK+02, HGH+02, LRR+02, MVR+01, MYC+02, NWK00, LLBB01]. Our data considers the *spoke* and *matrix* representation when the source purifies protein complexes [GBK+02, HGH+02]

Data Source	# Interactions
BIND [BDW+01]	6663
BRITE [KKK97]	46
DIP [XSD+02]	14635
DREES TABLE 1 [DSB+01]	373
ENRIGHT [EIKO99]	11
GAVIN [GBK+02]	25594
HO [HGH+02]	30782
LEE [LRR+02]	4540
LIEB [LLBB01]	769
MATTHEWS [MVR+01]	35
MIPS PPI [MFG+02]	15501
NEWMAN [NWK00]	229
PREDICTOME [MYC+02]	13022
TRANSFAC [WCH+00]	279
Total (including overlap)	112479

Table 6.2: Distribution of interactions per Explicit Data Source for YEAST

(see Section 2.2.1 for a discussion of GAVINM and HOM). Although interactions found in the papers may be included already in the database, some databases do not attribute references to the individual interactions they contain, or the list is incomplete with respect to the original paper. Thus, for completeness we include the papers as well. Because of the cost and effort in interrogating interactions, the papers listed generally represent unique studies for a particular assay type. Thus we attempt to compensate for overlap by creating experts according to *assay type*, not the data source, since most databases include the experimental method. For example, if an interaction is listed as found by tandem affinity precipitation (TAP), it is highly likely to have come from the paper by Gavin *et. al* [GBK+02]. Therefore, although the interaction may be listed by several databases, it will only be accounted for once by the TAP (or GAVINM) expert. Table 6.2 highlights the differences in coverage among data sources by showing the number of pairwise interactions contributed by each.

To create explicit experts from these sources, we extract the names of the experimental assays used to detect or predict interaction. Since many of the assay types occurred infrequently in the data or occurred with variants in spelling, we consolidate assay types with similar experimental bases. For example, <CONDITIONAL-SYNTHETIC-LETHAL-TEMPERATURE-SENSITIVITY>; <CONDITIONAL-SYNTHETIC-LETHAL>; <SYNTHETIC-LETHAL>; <SYNTHETIC-PHENOTYPE>; and <SYNTHETIC-GROWTH-EFFECT> are all combined into a type labelled SYNPHEN. After this transformation, any type with fewer than 200 occurrences is subsumed by the type labelled OTHER, which previously included interactions listed as OTHER by the data sources themselves. Table 6.3 lists the remaining assay types, each of which becomes an expert in our domain (the expert name is listed in the second column). Overall, the set of 21 experts assert 130091 pairwise interactions (67420 unique) involving 5690 distinct ORFs.

Assay Type	Expert Name	# Interactions
AFFINITY-CHROMATOGRAPHY	AFFCHR	1151
COPURIFICATION	COPURE	200
GENEFUSION	GENEFUSION	11709
GENETIC	GENETIC	226
HIGH-THROUGHPUT-EXPERIMENT	HITHRUEXP	11882
HMPCI_MATRIX	HOM	30781
IMMUNOPRECIPITATION	IMMUPREP	6453
IN-VITRO-BINDING	INVITRO	297
INDIVIDUAL-EXPERIMENT	INDIVEXP	812
LOCANALYSIS	LOCANA	4540
OTHER	OTHER	1073
PHYLOPROF	PHYPROF	1306
PHYSICAL	PHYS	300
PUBMED-UID	PMUID	4807
SUPPRESSION	SUPPR	431
SYNTHETIC-PHENOTYPE	SYNPHEN	3406
TAP	TAP	3242
TAP_MATRIX	GAVINM	25593
TRANSFAC	TRANSFAC	279
UNKNOWN	UNKNOWN	603
Y2H	Y2H	21000
Total (including overlap)		130091

Table 6.3: Distribution of interactions per Explicit Expert for YEAST

Attribute Type	Expert Name	# ORFs
Protein Complex [MFG+02]	COCOMPLEX	2738
Protein Function [MFG+02]	COFUNC	4541
Sequence Motif [MFG+02]	COMOTIF	2119
Protein Class [MFG+02]	COPROCAT	1013
Essentiality [MFG+02]	COESSENT	5983
Growth Phenotype [MFG+02, GCN+02]	COPHENO	5141
Cellular Location [KAH+02, DSB+01]	COLOC	5084
Cell Cycle Phase [SSZ+98]	COPHASE	798
GO Cellular Component [ABB+00, CAB+98]	GO_CC	5868
GO Molecular Function [ABB+00, CAB+98]	GO_MF	5868
GO Biological Process [ABB+00, CAB+98]	GO_BP	5868

Table 6.4: Distribution of coverage per Implicit Expert for YEAST

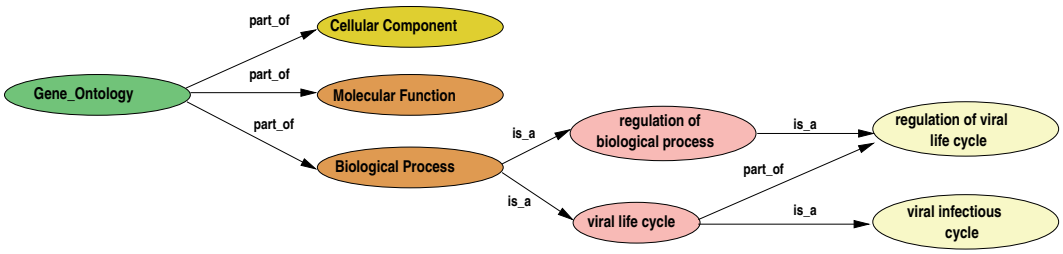


Figure 6.4: Excerpt of the Gene Ontology (taken from [WABD04])

.

Yeast Implicit Experts

Various information sources include attributes about individual genes, such as location or function. As discussed in Section 2.2.2, these sources can be used to derive pairwise interactions if two genes share the same category for a particular attribute. The MIPS catalogues [MFG+02] are used for describing protein complex, protein function, sequence motifs, protein class, and essentiality. Growth phenotypes are listed in MIPS as well as in Giaever *et al.* [GCN+02]. Cellular location is given by MIPS in conjunction with two other studies [KAH+02, DSB+01]. Cell cycle phase is given by Spellman *et al.* [SSZ+98], where for our purposes, we consider both known and predicted phase as truth. Table 6.4 lists the attribute type along with references, the corresponding expert designation, and the number of ORFs annotated with the attribute.

Three of the experts in Table 6.4, namely GO_CC, GO_MF and GO_BP, involve the Gene Ontology (GO) [ABB+00], a directed acyclic graph of biological terms related by *is_a* or *part_of* relationships. An excerpt of the ontology is given in Figure 6.4 where the root term branches into three distinct ontologies describing cellular component, molecular function, and biological process, respectively. Determining whether two ORFs share a common GO term is complicated by the fact that the structure of the ontology allows terms to inherit from their parents. Thus, two ORFs may not share identical term assignments but their assigned terms may be closely related in the ontology. In such a case, we would like to assert that a relationship exists. Generally, less specific terms are closer to the root of the ontology, yet idiosyncrasies of biological concepts and the annotation process result in nonuniform expansion across the ontology. Thus, similarity of two ORFs by the terms assigned to them cannot simply be computed as the path length of the nearest common ancestor term from the root, as is commonly done in other hierarchical ontologies. Instead, we use an information-based metric JIANG [LSBG03, JC97] which captures the intuition that less frequently used terms are more informative.

Originally a distance metric, the JIANG measure is reformulated to calculate the similarity of GO terms assigned to two ORFs. Note that an ORF may be assigned several terms. Given a branch of the ontology and a set of ORFs with GO term assignments, we calculate the total occurrence t of all terms under that branch by summing over the set of all terms assigned to each ORF, summed over all ORFs. The probability p_c of any term c is found as the number of occurrences of that term

for the set of ORFs, divided by the total t. Thus, the probability of the root of each of the three branches is 1. The value $-\log p_c$ is known as the information content of the term c. For two terms c_1 and c_2, we can calculate the *probability of the minimum subsumer* p_{ms} as follows:

$$p_{ms} = \min_{c \in S(c_1, c_2)} \{p_c\},$$

where $S(c_1, c_2)$ represents the set of ancestor terms shared by both c_1 and c_2.

The original distance formulation of JIANG yielded a maximum value of $2 \log t$ given a corpus. Thus, we modify the JIANG measure to calculate similarity as follows:[1]

$$\text{sim}(c_1, c_2) = 2 \log t - [2 \log p_{ms} - (\log p_1 + \log p_2)]. \tag{6.1}$$

If $\log p_{ms} = 0$, we set $\text{sim}(c_1, c_2)$ equal to 0.

Given a pair of ORFs with (multiple) assigned terms, we chose the pair $\{c_1, c_2\}$ from each ORF, respectively, which results in the maximum similarity. To determine the appropriate threshold for the JIANG measure in order to assert that two ORFs were indeed related according to the GO branch expert, we examine the correlation between the similarity values and the set of interactions indicated by the other expert data sources (see Appendix D). Using all 5868 yeast ORFs with GO terms assigned by SGD as of November 2005 [CAB+98], we found that a value of 16 for the Cellular Component and Biological Process branches and a value 15 for Molecular Function were reasonable choices with respect to distinguishing the distribution of similarity over all pairs from the distribution of positive assertions made by each of the other experts.

6.2.2 Yeast Reliability Assignments

In order to apply the LinOP and NoisyOR consensus likelihood functions, we consider several strategies for estimating reliability for each of the yeast experts, explicit or implicit. As in the ALARM domain, we wish to explore the robustness of the functions to incorrect reliability assignments. However, unlike in the ALARM domain, determining the reliability of the yeast experts cannot assume the true set of interactions is known. Though we have no gold standard for yeast, we can still explore the space of reliability assignments using the following four policies for assigning reliabilities:

1. Assign random reliabilities (**RAND1** and **RAND2**)
2. Assign uniform reliability of 5 on a 10 point scale (**UNIF5**)
3. Solicit reliabilities from a real biologist (**MAIR**)
4. Assign reliabilities by agreement to consensus (**CONS**)

The first and second methods populate the range of correctness of reliabilities by mimicking the methods for ALARM. By assigning random reliabilities to the experts, we can see the effect of

[1] The formula for JIANG in Lord *et al.* [LSBG03] has a typographical error, incorrectly including a minus sign in front of 2 $\log p_{ms}$.

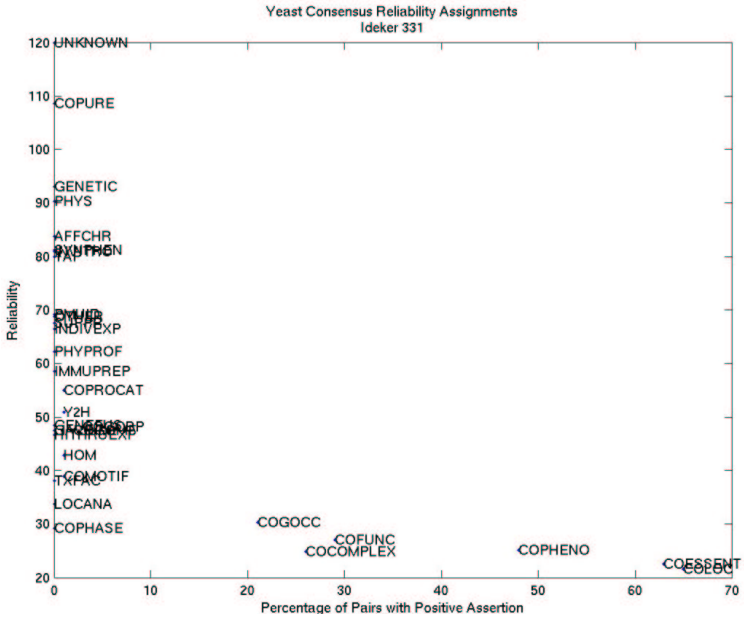


Figure 6.5: Comparing coverage and **CONS** reliability in YEAST

.

a largely incorrect assignment, akin to the 2.1.0 assignment in ALARM. The uniform assignment corresponds to the 1.1.1 ALARM assignment, where we can see the effect of ignoring the relative reliability of experts.

The third method assesses reliability by presenting a biologist, Dr. Mair Churchill of the University of Colorado at Denver and Health Sciences Center, with a list of assay techniques and asking her to rate the ability of the technique to detect interaction between genes, on a 10 point scale. The strategy is thus named in her honor. In cases where Dr. Churchill provides a range, the average reliability value is taken. For the experts not included on the list given to Dr. Churchill, we give intuitive assignments based on the scale and coverage of the technique using our knowledge from working with these types of data. The full list of reliabilities is shown in Table 6.5.

The last method implements the intuition that an interaction seen by multiple methods is more likely to be correct, as shown in several prior studies [vMKS+02, DSXE02, BH02, HSvMB03]. Note that this assumes the collection does not consist entirely of fallacious experts. Examples of assessing reliability without a gold standard using consensus theory can also be found in judging inter-annotator agreement on corpora annotation in natural language processing [PGK05, KT06] and in evaluating rater performance on medical diagnoses [WM97]. Since our experts only make positive assertions, we do not directly use the methods in those resources. Instead for the **CONS** reliability assignment, we calculate the sum S_{ij} of the number of experts asserting each edge e_{ij} (between 0 and 32 in the case of our yeast example). Then for an individual expert e, we take as the reliability R_e, the average (times 100) over $(S_{ij} - 1)$ for the set of edges e_{ij} asserted by the expert.

Assay Type	Expert Name	MAIR	CONS
AFFINITY-CHROMATOGRAPHY	AFFCHR	9	83
COPURIFICATION	COPURE	8	108
GENEFUSION	GENEFUSION	3	48
GENETIC	GENETIC	6	93
HIGH-THROUGHPUT-EXPERIMENT	HITHRUEXP	2	46
HMPCI_MATRIX	HOM	5	42
IMMUNOPRECIPITATION	IMMUPREP	8	58
IN-VITRO-BINDING	INVITRO	8	80
INDIVIDUAL-EXPERIMENT	INDIVEXP	9	66
LOCANALYSIS	LOCANA	3	33
OTHER	OTHER	8	68
PHYLOPROF	PHYPROF	3	62
PHYSICAL	PHYS	6	90
PUBMED-UID	PMUID	4	69
SUPPRESSION	SUPPR	6	67
SYNTHETIC-PHENOTYPE	SYNPHEN	6	81
TAP	TAP	5	80
TAP_MATRIX	GAVINM	5	47
TRANSFAC	TRANSFAC	8	38
UNKNOWN	UNKNOWN	3	120
Y2H	Y2H	7	50
Cellular Location	COLOC	2	21
Protein Complex	COCOMPLEX	1	24
Protein Function	COFUNC	5	27
Sequence Motif	COMOTIF	2	38
Growth Phenotype	COPHENO	2	25
Essentiality	COESSENT	1	22
Protein Class	COPROCAT	2	54
Cell Cycle Phase	COPHASE	1	29
GO Cellular Component	GO_CC	2	30
GO Molecular Function	GO_MF	5	47
GO Biological Process	GO_BP	5	48

Table 6.5: Expert Reliability Assignments for YEAST

Expert Name	CONS	BAYES UNSET	BAYES SRG0	BAYES SRG3	PRM UNSET	PRM SRG0	PRM SRG3
OTHER	68	59	94	63	69	64	63
COLOC	21	53	88	55	69	74	70
COCOMPLEX	24	47	99	51	73	63	61
COFUNC	27	56	82	62	42	46	46
COMOTIF	38	63	89	68	76	77	76
COPHENO	25	98	99	99	42	47	47
COESSENT	22	95	99	98	82	84	82
GO_CC	30	59	91	64	88	86	85
GO_MF	47	64	94	78	90	83	83
GO_BP	48	66	99	81	90	86	86
Corr. to CONS	1.0	-0.19	0.09	-0.06	0.29	0.17	0.21

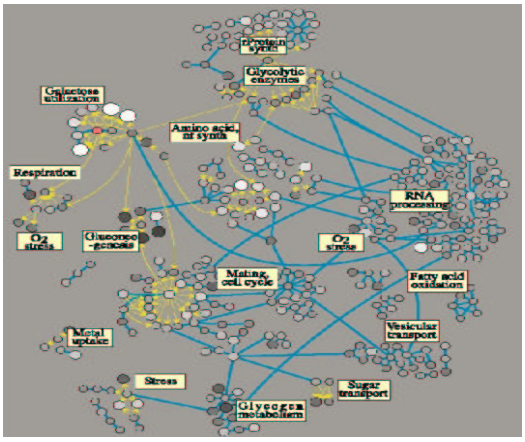
Table 6.6: Expert Reliability Assignments for YEAST (continued)

Figure 6.5 shows a graph of the coverage of an expert as the percentage of pairs asserted by the expert versus the **CONS** reliability, based on a set of 331 yeast genes (discussed below). Since the high-throughput and implicit expert sources have many edges, their reliability according to **CONS** will be averaged over a highly variant distribution. The resulting lower reliability of these sources (lower right of graph) is in agreement with the intuition that these sources are less correct for predicting interaction. The numerical **CONS** reliability values from the 331 gene dataset are given in the right-hand column of Table 6.5. There is a reasonably strong correlation of **CONS** to the **MAIR** assignment (r=0.54). Correlations of **CONS** to the two random assignments were 0.1629 and -0.2390, respectively (data not shown).

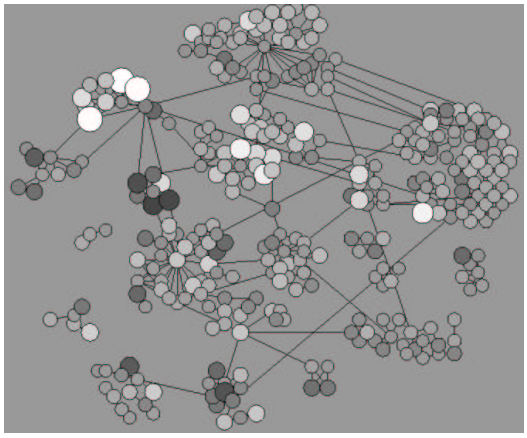
With forward reference to some terms presented in the next section, Table 6.6 contains the reliabilities calculated from each model learned using the BAYES and PRM consensus likelihood functions for the experts remaining using $MV = 5000$. Reliabilities are calculated using Eq (5.6) for BAYES and Eq (5.9) for PRM, scaled from their original 0-1 point scale to a 0-100 point scale. Three different datasets are used to learn the models, **UNSET**, **SEMRELG0** and **SEMRELG3**, which is reflected in the column headings of Table 6.6 with **SEMREL** abbreviated to **SR**. Note the poor correlation of the BAYES reliabilities with **CONS** and the relatively higher correlation for PRM.

6.2.3 Computing Yeast Consensus Likelihoods

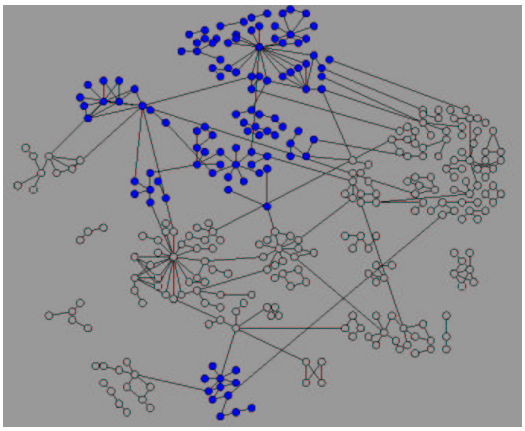
As a concrete example for the yeast genome, the four consensus likelihood functions (LinOP, NoisyOR, BAYES, PRM) are calculated for the 331 genes appearing in the galactose study by Ideker *et al.* [ITR+01], discussed briefly in the Introduction. Figure 6.2.3 shows the set of genes in the context of their functional category and protein-protein or protein-DNA interactions in four different graphs. For convenience and comparison, Figure 6.2.3(a) repeats the graph shown in Figure 1.1, taken from Ideker *et al.* [ITR+01]. Figure 6.2.3(b) shows the same graph visualized using the Cytoscape graphing package [SMO+03]. We refer to this gene set as IDEKER331.



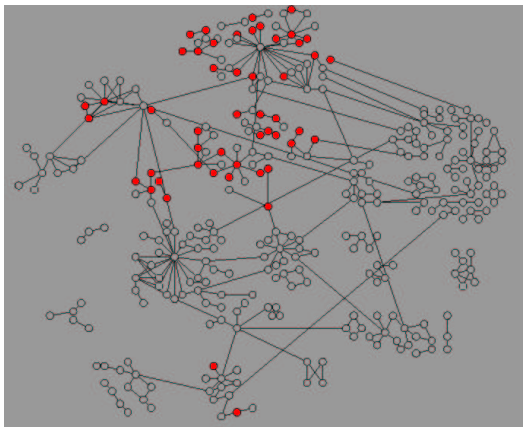
(a) 331 yeast genes [ITR$^+$01].



(b) IDEKER331



(c) IDEKER114 (blue circles)



(d) IDEKER50 (red circles)

Figure 6.6: YEAST IDEKER Gene Sets

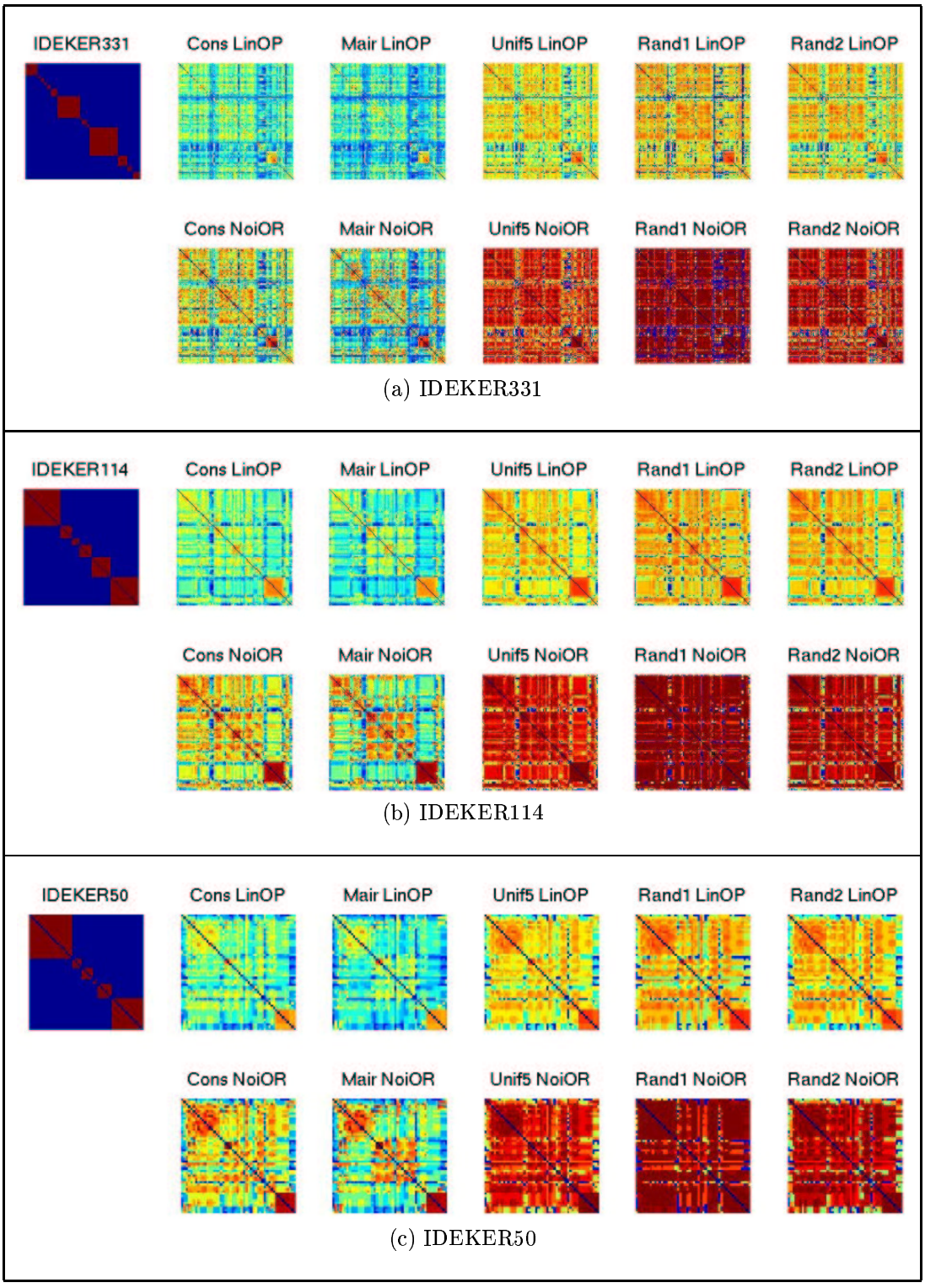
Functional categories are assigned to each gene by manual inspection of labels from Figure 6.2.3(a). Note that the ambiguity in spacial placement of the nodes might result in some noise in the resulting labels. The set of functions is used to create two smaller datasets, IDEKER114 and IDEKER50, which are shown in Figure 6.2.3(c) and Figure 6.2.3(d), respectively. The gene set IDEKER114 retains all genes with the functional categories Amino acid synthesis, Galactose utilization, Gluconeogenesis, Glycogen metabolism, Glycolytic enzymes, or rProtein synthesis. This set of genes is chosen since it is reasonably well covered by the experts in terms of assertions of interaction. The set IDEKER50 is chosen randomly from within the IDEKER114 gene set.

Consensus likelihoods are calculated on the IDEKER331 gene set using the set of experts listed in Section 6.2.1 together with the four functions LinOP, NoisyOR, BAYES, and PRM. We present each of the four methods in the following sections. We also include two other consensus likelihood matrices offered by other groups, namely the matrix MAGIC from the work of Troyanskaya *et al.* [TDO+03] seen previously in Figure 5.3, and the matrix STRING from the work of von Mering *et al.* [vMHJ+03, vMJS+05] mentioned in Section 2.2.1. All results for the IDEKER114 and IDEKER50 gene sets were extracted from the consensus likelihood functions learned on the full set of 331 genes (IDEKER331).

LinOP and NoisyOR Consensus Likelihood for YEAST

The LinOP and NoisyOR is applied for each of the five reliability assignments, **UNIF5, RAND1, RAND2, MAIR** and **CONS** to yield a set of 10 corresponding probabilistic adjacency matrices $\mathcal{W}^{\text{LinOP}}$ and $\mathcal{W}^{\text{NoisyOR}}$. The matrices are shown for all IDEKER331 genes in Figure 6.2.3(a) using the same color scale as Figure 6.3. The matrix on the far left is the deterministic adjacency matrix formed by placing an edge between two genes with a common functional category, as assigned manually from Figure 6.2.3(a). The size of a box along the strong red diagonal indicates the number of genes annotated to a particular function. Recall that this assignment is subject to noise due to the challenges of assigning a category by visual inspection of the original figure. Nonetheless, it does provide a useful organization of the results since all matrices use the same ordering of genes along the rows and columns of the adjacency matrix. This allows us to give qualitative assessments regarding the ability of a method to recreate the original functional categories. Graphs for IDEKER114 and IDEKER50 in Figure 6.2.3(b) and Figure 6.2.3(c) offer additional clarity.

Evident in Figure 6.2.3 is the similarity in pattern across reliability assignments, with the LinOP matrices generally consisting of lower probability than the corresponding NoisyOR matrices, echoing the same results found for the ALARM dataset. Note that a strictly dark red matrix denotes a completely connected graph, *i.e.,* providing no useful information about the probability of an edge as all edges are asserted. The **UNIF5** and **RAND** examples are significantly more red than the other matrices (median matrix probability in the range 0.44 to 0.49 versus 0.20 for **MAIR**) indicating that these reliability assignments are less discriminating than the **CONS** and **MAIR** assignments. The **CONS** results are interesting since its matrices are comparable to those of the hand-assignment of reliability by **MAIR**. This suggests that it is a reasonable alternative for automatically assigning



(a) IDEKER331

(b) IDEKER114

(c) IDEKER50

Figure 6.7: LinOP and NoisyOR Consensus Likelihood Matrices for YEAST

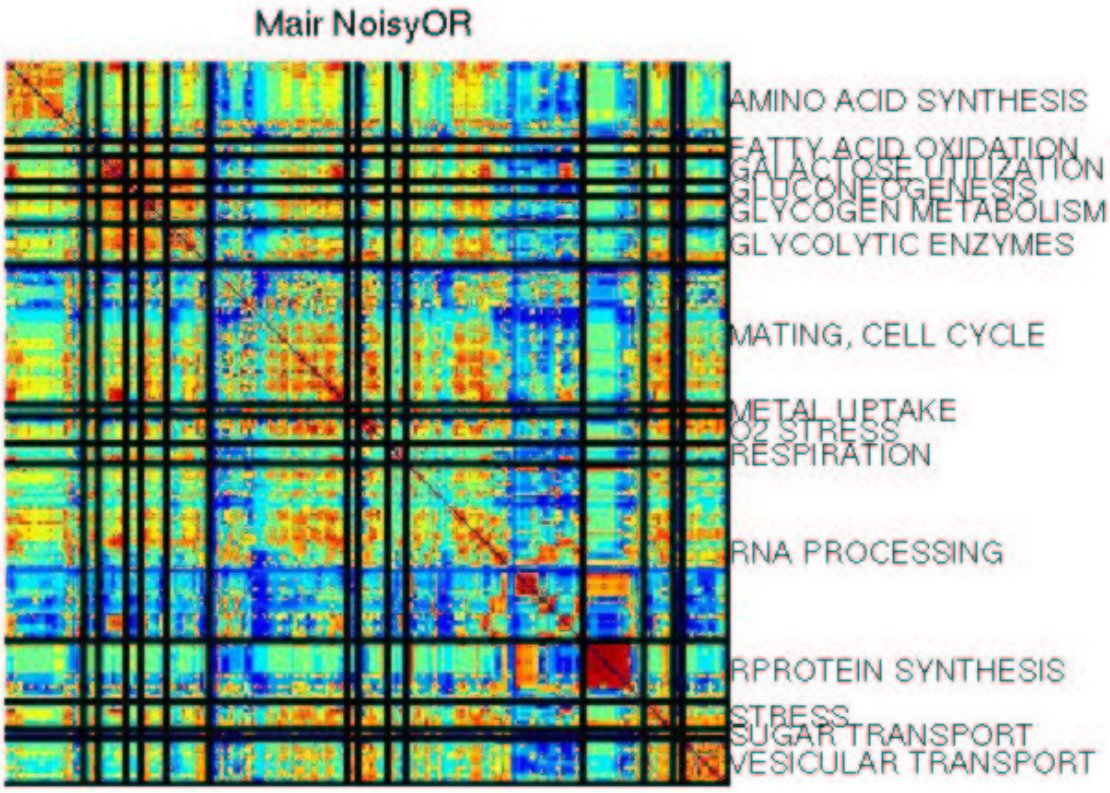


Figure 6.8: Example of correspondence of consensus likelihood to functional categories in IDEKER331 YEAST (using MAIR NoisyOR).

reliabilities. Also, there is a nice interplay between **CONS** and NoisyOR. Recall that NoisyOR is susceptible to incorrect reliability assignments. By requiring consensus, the **CONS** function conservatively assigns reliabilities, ensuring that NoisyOR is not strongly affected.

Also of note in Figure 6.2.3 is the presence of red boxes along the diagonal, seen most clearly with the IDEKER114 set shown in Figure 6.2.3(b). These clusters roughly capture a few of the functional categories. Figure 6.8 shows the correspondence of **MAIR** NoisyOR to the functional categories for IDEKER331. Failure of a method to distinguish the categories may be due in part to the fact that the experts from which it was created represent more types of relationships than just than functional relationships.

BAYES on YEAST

For the BAYES consensus likelihood function, the experts contribute variables to the model. The BAYES model for yeast is shown in Figure 6.9, discussed originally in Section 5.3 and repeated here for convenience. Note that the coverage of experts with respect to the number of asserted interactions degrades as the set of genes grows smaller. To avoid including the corresponding variables which have an abundance of missing values and therefore incur the cost of inference during learning, experts

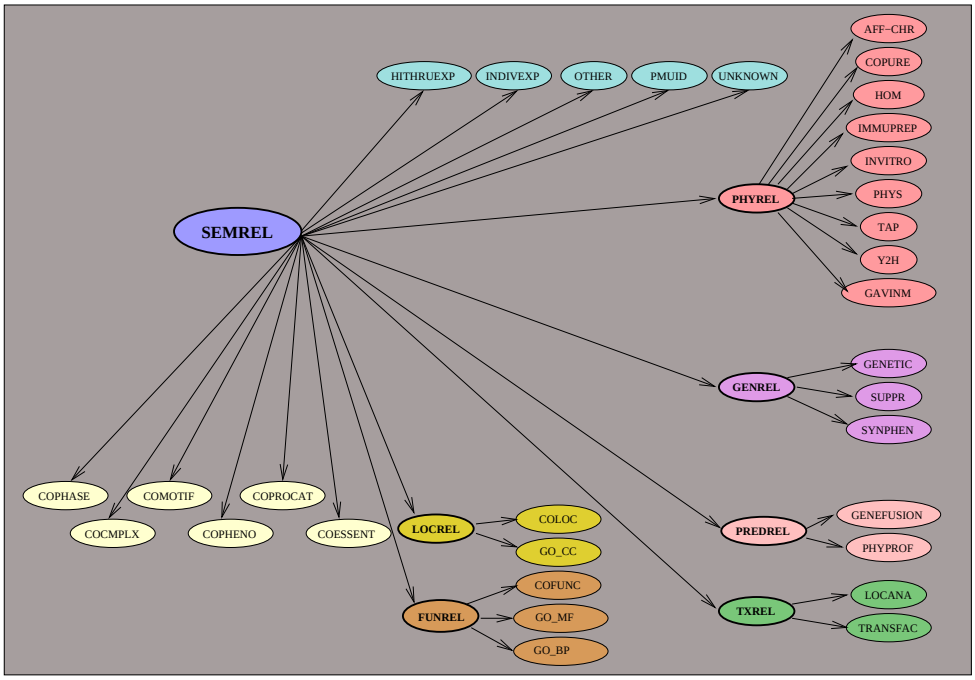
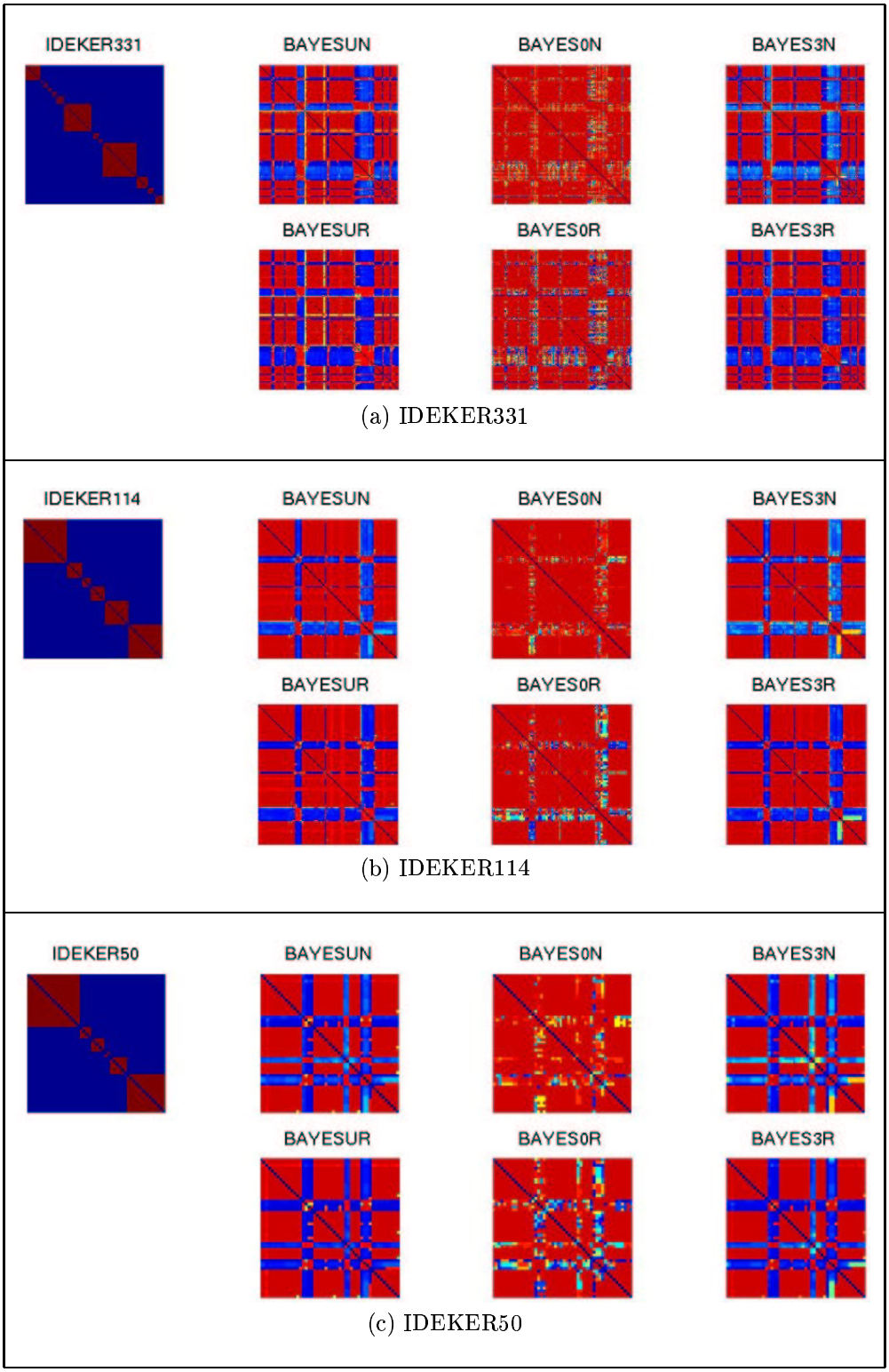


Figure 6.9: BAYES Model for Computing Consensus Likelihood in YEAST

with fewer than a *minimum value* (MV) of interaction assertions are absorbed by the OTHER expert type in the BAYES model. For the 32 experts on the IDEKER331 gene set, the number of assertions per expert originally ranged from 2 to 35341. Since there are 54615 pairs among 331 genes, we choose a reasonably large minimum value. Using MV=5000, 10 experts remain, namely COLOC, COCOMPLEX, COFUNC, COMOTIF, COPHENO, COESSENT, GO_CC, GO_MF, GO_BP, and OTHER. Essentially, the explicit experts are assimilated into the OTHER expert. Note that this has the consequence that an interaction seen by more than one explicit expert will only be observed once by the OTHER expert.

Using the set of 10 remaining experts, three datasets are created for learning the BAYES model, denoted **UNSET**, **SEMRELG0** and **SEMRELG3**. The variants refer to whether or not the SEMREL variable was set, as discussed in Section 5.3. The dataset **UNSET** does not include values for SEMREL while **SEMRELG0** and **SEMRELG3** refer to how many positive assertions of interaction for a gene pair must be present in order to set the corresponding SEMREL variable to 1 (true). The name **SEMRELG0** means greater than 0 experts must assert an interaction to cause SEMREL to be set (*i.e.,* any expert asserts an interaction) while the name **SEMRELG3** means greater than 3 experts must make assertions. Setting SEMREL to 0 (false) follows the same policy outlined in Section 5.3. The idea is to demonstrate the benefits of providing partial estimates of SEMREL and testing whether it is enough that one expert asserts an edge or must the edge be asserted multiple times.

With 54615 possible pairs among 331 ORFs, a complete dataset over the 13 variables of the



(a) IDEKER331

(b) IDEKER114

(c) IDEKER50

Figure 6.10: BAYES Consensus Likelihood Matrices for YEAST

BAYES model would have 709995 values. For comparison, the **UNSET** dataset had 346242 values, the **SEMRELG0** dataset had 400850 values and the **SEMRELG3** had 365138. Using each of the three datasets, the EM algorithm is applied using the structure (or subset thereof) shown in Figure 6.9, computing MAP estimates of the parameters given the BDeu prior with $N' = 1$ (see Section 3.2.3). We set the maximum number of iterations of EM to 10, with 10 interior iterations of loopy belief propagation per EM iteration.

Using the parameters learned for the BAYES model, we calculate the probability $Pr(e_{ij})$ that an interaction exists between genes X_i and X_j. This value can be extracted by computing the marginal probability $Pr(\mathsf{SEMREL} = 1|d_{ij})$ where d_{ij} is the datacase representing the pair X_i and X_j. The value $Pr(e_{ij})$ can also be calculated as the ratio of the marginal to the prior probability of SEMREL without evidence, as in Eq (5.6). We consider both versions, where in the models learned using **SEMRELG0** and **SEMRELG3**, $Pr(\mathsf{SEMREL} = 1|d_{ij})$ is calculated with SEMREL unset. Our default probability of an edge is set to $\mathcal{R}_0 = 0.1$. With all three datasets and both ways of calculating probabilities, we obtain six probabilistic adjacency matrices $\mathcal{W}^{\text{BAYES}}$ for the BAYES consensus likelihood function.

The six matrices are shown in Figure 6.2.3, where the title above the matrix denotes which dataset and which method of calculating probabilities was used. For example, the name BAYESUN denotes learning from the **UNSET** dataset using **No ratio** while the name BAYES3R denotes learning from the **SEMRELG3** dataset using the **Ratio** computation. We can see from the figure that there is little correspondence to the assigned functional categories since the red boxes are not confined to the diagonal. Certain categories are preserved but the strong presence of red boxes off the diagonal indicate that the number of (potential) false positives is high with respect to functional categorization. Also, the distribution of probabilities is mostly bi-modal at 0 and 0.5 for the UNSET cases. The **SEMRELG0** examples show more intermediate values, reflecting the fact that more of the SEMREL values were set than in the corresponding **SEMRELG3** matrices, thereby influencing the inference procedure more heavily. Also, the effect of applying the ratio technique results in a slight shift in the probability distribution by transforming the values more towards the extremes of 0 and 0.5. Compared with the LinOP and NoisyOR matrices on the corresponding gene sets, BAYES proves to be less accurate in recovering functional designation and less discriminating in the set of interactions asserted.

PRM on YEAST

The PRM consensus likelihood function also models the individual attributes used to derive interactions asserted by the implicit experts, in addition to identifying the experts as in BAYES. Figure 6.11 shows an excerpt of the full PRM schema, discussed initially in Section 5.4 and replicated here for convenience. A subset of the original 32 experts are shown, though in reality the same 10 experts as in BAYES resulting from setting MV=5000 are used for the IDEKER331 gene set.

As can be seen from the figure, each gene contributes a set of (binary) attribute variables representing the categories for each type of implicit expert, such as Location or Motif. As in BAYES,

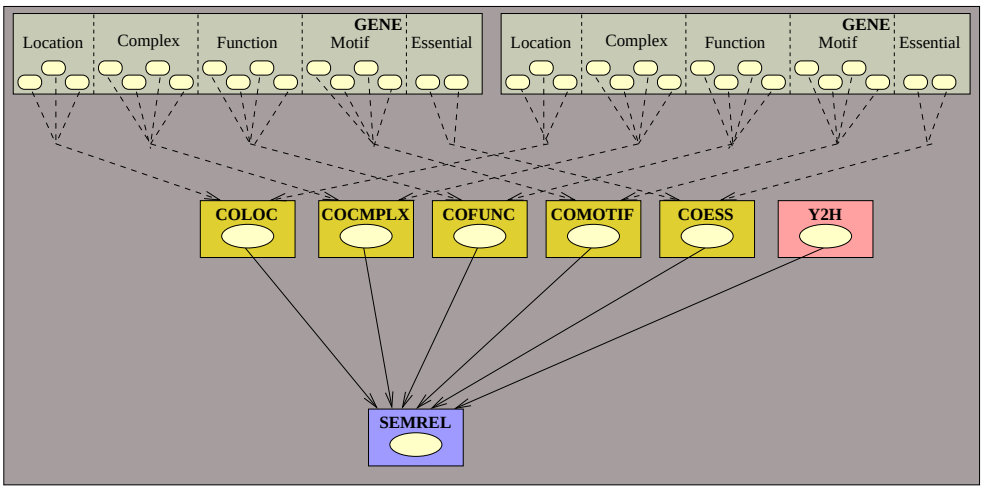
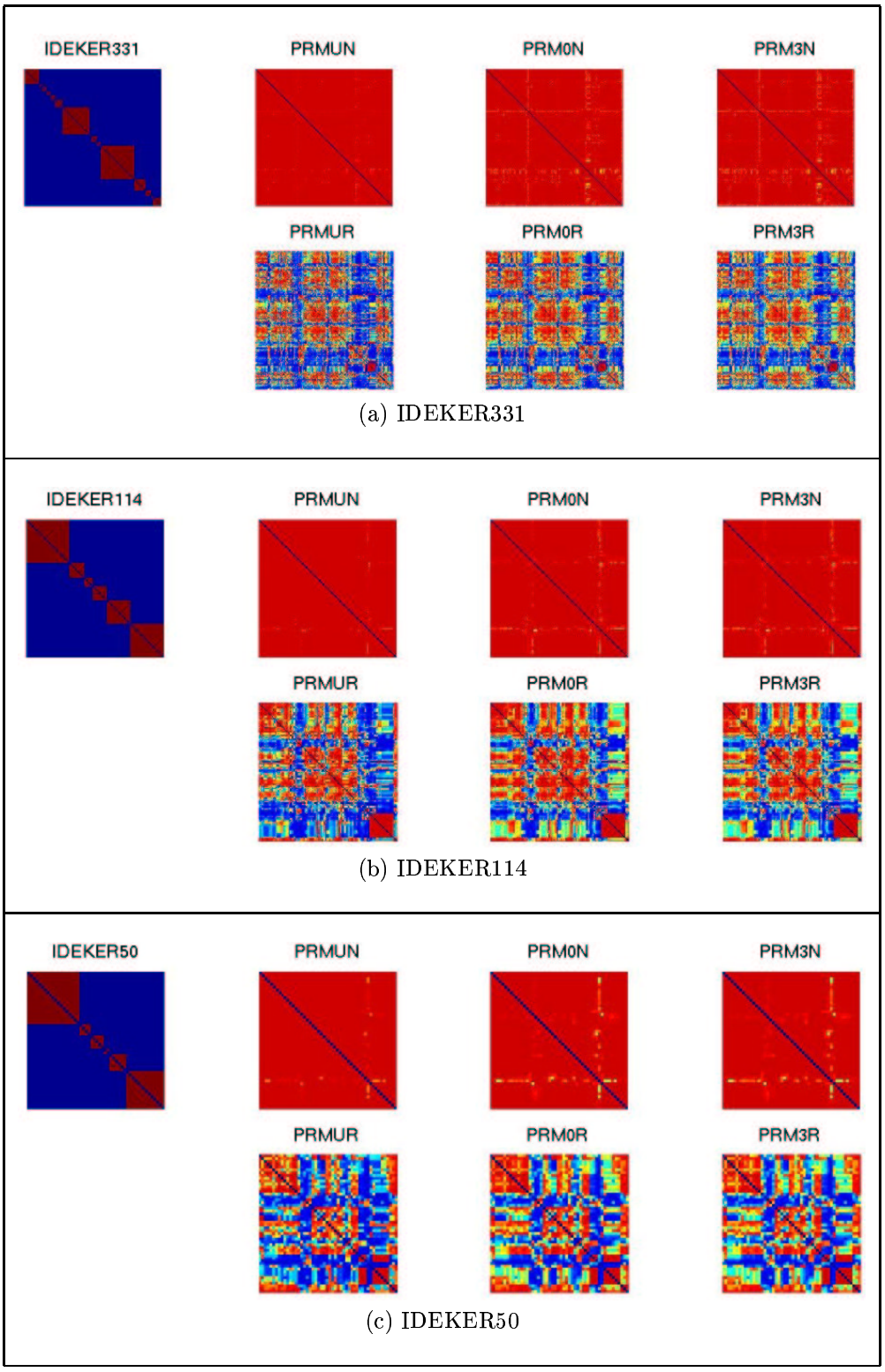


Figure 6.11: PRM schema for YEAST

we do not want to model a large number of variables with very little data available for learning the model parameters. Since the number of categories per individual gene attribute range from 2 to 511, we employ a *maximum attribute* (MA) threshold for the number of categories allowed. All other categories are subsumed by the category *Other*. We use MA=5 since that value is large enough to capture several highly populated categories yet low enough to avoid an unmanageable explosion in the number of variables. Note that the assertion of an interaction between two genes by an implicit expert based on common categories is made before the identity of the implicit expert is absorbed into the *Other* category. This means that although two genes may technically be labelled *Other* for a particular attribute, they may not necessarily be considered as interacting by the implicit expert since they did not share a common category initially.

As with the BAYES model, three datasets **UNSET**, **SEMRELG0** and **SEMRELG3** are created for learning the PRM model parameters, based on how the variables $\mathsf{SEMREL}_{ij}.\mathsf{Exists}$ are set (or not) for all pair of genes X_i and X_j. The unrolled PRM model has 1212122 variables yet with parameter tying and special types of conditional probability distributions at many of the nodes, only 89 parameters (for 54 conditional probability distributions) must be learned. The coverage of the 1212122 variables for each of the datasets is 349865, 404472, and 368760 for **UNSET, SEMRELG0** and **SEMRELG3**, respectively. Though all datasets are sparse in comparison to the number of variables, recall that many variables are added for the structural decomposition of the noisy-OR node SEMREL.Exists and these variables will never have values in a dataset.

To learn the model parameters from each of these datasets, EM is applied using BDeu priors with $N' = 1$. We set the maximum number of iterations of EM to 10, with 10 interior iterations of loopy belief propagation per EM iteration. The probability $Pr(e_{ij})$ of an edge can either be inferred directly as $Pr(\mathsf{SEMREL}_{ij}.\mathsf{Exists} = 1|D)$ for dataset D or as the ratio of this value to the prior probability computed without evidence, as in Eq (5.9). The default probability of an edge is again set to $\mathcal{R}_0 = 0.1$. Analogous to the six matrices computed for $\mathcal{W}^{\text{BAYES}}$ by varying dataset



(a) IDEKER331

(b) IDEKER114

(c) IDEKER50

Figure 6.12: PRM Consensus Likelihood Matrices for YEAST

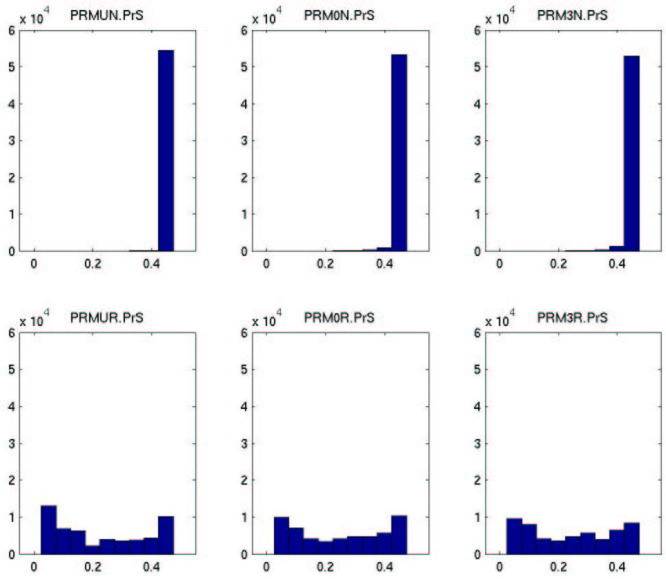


Figure 6.13: Distribution of Probability for PRM matrices on IDEKER331 YEAST dataset

.

or method of computation using SEMREL.Exists, the probability of an edge can be used to create six probabilistic adjacency matrices $\mathcal{W}^{\text{PRM}}$ for the PRM consensus likelihood function, shown in Figure 6.2.3.

Immediately apparent from Figure 6.2.3 is the effect of applying the ratio method for calculating probabilities. The nearly completely red matrices on the top row are transformed into matrices with distributions of probabilities more like the other consensus likelihood matrices. The effect of using the ratio for PRM is much more dramatic than for the BAYES matrices. With respect to reproducing the functional categories, there is a strong red box on the lower right of each figure and several others along the diagonal but overall, there are also many off-diagonal red areas, indicating crosstalk between functional categories. However, the performance of PRM is better than BAYES on this example.

It is difficult to determine by visual inspection of the figure, here and in the equivalent BAYES figure, whether the SEMREL.Exists variable should be set with only one expert or more than three. For comparison, we look for differences between the PRM0N/PRM0R matrices learned from the **SEMRELG0** dataset and the corresponding PRM3N/PRM3R matrices learned from **SEMRELG3**. Close observation reveals some differences but to gain a better understanding, the distributions of probabilities in the IDEKER331 matrices are shown as histograms in Figure 6.13. In the **No Ratio** versions, the probabilities heavily populate the upper range while in the **Ratio** versions, the distribution is more uniform. Comparing PRM0R to PRM3R, there are more values in the 0.3 area in PRM3R than in PRM0R, reflecting greater discrimination for edges with less than three but greater than 0 expert assertions. From these plots, we can conclude that PRM3R in the most discriminating version of the PRM consensus likelihood function.

Other Consensus Likelihood Matrices for YEAST

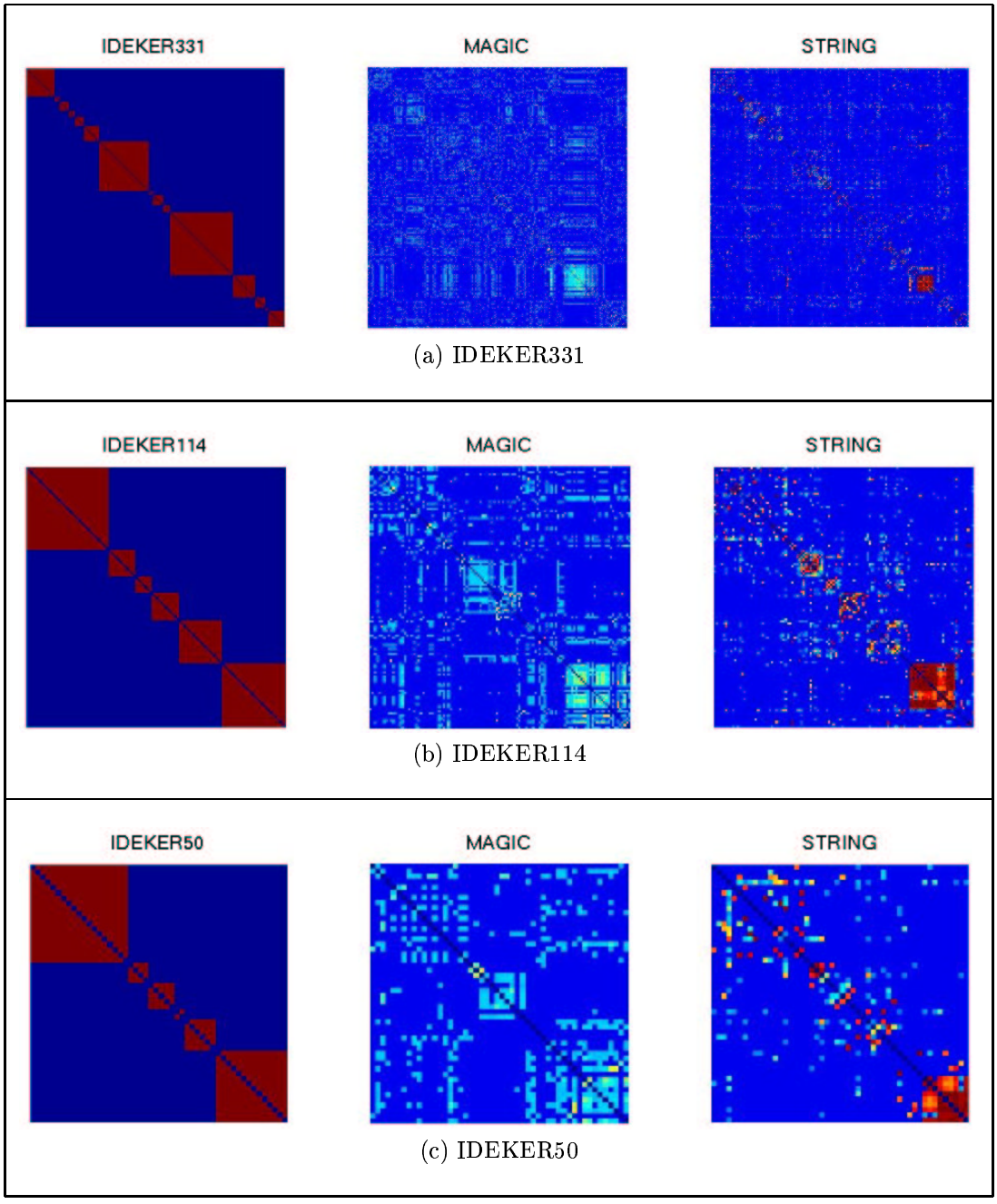
Each of the four consensus likelihood functions presented in this thesis are probabilistic characterizations of pairwise interactions. Two previous studies involving yeast offer similar pairwise distributions: the MAGIC model from Troyanskaya *et al.* [TDO+03] seen previously in Figure 5.3, and the STRING database from the work of von Mering *et al.* [vMHJ+03, vMJS+05] mentioned in Section 2.2.1. We discuss each below.

MAGIC (Multisource Association of Genes by Integration of Clusters) [TDO+03] uses a Bayesian network structure with conditional probabilities solicited from yeast biology experts. The purpose of the model is to predict functional relationships between two genes by integrating explicit sources measuring transcription, physical, and genetic associations, together with implicit sources measuring co-localization and co-expression. A list of 4457886 pairwise predictions can be downloaded from `http://genome-www.stanford.edu/magic/predictions.txt`. The set of interactions for IDEKER331, IDEKER114, and IDEKER50 are extracted from this file to create a probabilistic adjacency matrix $\mathcal{W}^{\text{MAGIC}}$. To ensure that the resulting probability matrix contains non-zero values (excluding the diagonal), a default probability of an edge $\mathcal{R}^0 = 0.1$ was added to the matrix before renormalizing to yield a proper set of probabilities. Figure 6.2.3 illustrates $\mathcal{W}^{\text{MAGIC}}$ for the three gene sets.

The STRING (Search Tool for the Retrieval of Interacting Genes/proteins) database [vMHJ+03, vMJS+05] contains known and predicted protein-protein interactions which make use of 179 genomes, primarily bacterial, thereby capturing highly conserved interactions. Interactions are based on co-occurrence in protein names in PubMED [2] literature abstracts, on databases of experimental interactions (GRID [SBR+06], MINT [ZMQ+02], BIND [BDW+01], and DIP [XSD+02]), on databases of functional interactions (MIPS [MFG+02] and KEGG [KGH+06]), on co-expression and on computationally predictive methods based on gene neighborhood, gene fusion, and gene co-occurrence (phylogenetic profiles). Like in our formulations of a consensus likelihood of interaction, predictions in the STRING database are made by combining reliability estimates of each method. A reliability score for each method is estimated based on the percentage of predicted pairs known to be assigned to the same KEGG pathway. The reliability scores are then combined into a probability measure via a Noisy-OR function of the reliabilities (see Section 5.2). The set of prediction for our experiments involving the IDEKER gene sets are extracted from the file downloaded from `http://string.embl.de/version_6_2/`. Again, we employ a default edge probability of $\mathcal{R}_0 = 0.1$ to insure a positive adjacency matrix. Figure 6.2.3 shows the resulting consensus likelihood matrices $\mathcal{W}^{\text{STRING}}$ for the three gene sets.

The first observation from Figure 6.2.3 is that MAGIC and STRING assert many fewer edges than the other methods, perhaps reflecting the fact that they use fewer implicit sources. Also, there are boxes of high probability along the diagonal, especially in the STRING graph where these correspond closely to the original functional categories. The experiments of Chapter 7 investigate the effect of having fewer, and strongly functional, interactions with respect to the ability to provide priors for Bayesian network learning.

[2] `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?DB=pubmed`



(a) IDEKER331

(b) IDEKER114

(c) IDEKER50

Figure 6.14: MAGIC and STRING Consensus Likelihood Matrices for YEAST

6.3 Consensus Likelihood in Mouse

In contrast to the genome of the single-celled yeast, the mouse genome provides the opportunity to study interaction data in a multi-cellular, multi-tissue environment. Often relationships between genes depend on the tissue type, thus we can study as a baseline how useful are consensus likelihoods calculated from data regardless of the cellular context. According to the Mouse Genome Database (MGD) [EBK+05] as of April 2006, there are 30917 mouse genes with either DNA or protein sequence information, 15703 of which have orthology to human genes. A large number of genes have little known characterization, evidenced by the fact that only 16764 gene markers have been assigned terms from the Gene Ontology (GO) [ABB+00] as of November 2005. We would like our consensus likelihood to provide potential context for the unknown genes. Our investigation is similar to that in yeast only the data sources for the experts change. We discuss the differences in the sections below.

6.3.1 Mouse Data and Experts

As with yeast, experts in the mouse domain correspond to assay types. Explicit experts are provided by the references listed in Table 6.7(a) along with their coverage in terms of number of interactions found. Immediately apparent from all tables in this section is the smaller number of interactions for mouse relative to the yeast domain, even though mouse has more genes than yeast. The IntAct data contains pairwise interactions where protein complexes were expanded using the spoke model. Therefore to be more complete, we create a matrix model by asserting interactions between the proteins sharing a common bait (recall definitions in Section 2.2.1). The DRABKIN results are mined from mouse genes annotated with the GO term Protein binding (GO:0005515) using the IPI (inferred from physical interaction) evidence code which often provides an identifier for the binding partner. Note that this information is different than interactions extracted using the GO implicit experts who instead assert relationships among fellow terms annotated with Protein binding (GO:0005515).

For the explicit experts, we employ the same methods as in yeast for consolidating assay types with similar experimental bases, using a threshold of $MV = 50$ for the minimum number of occurrences for the assay type not to be subsumed by OTHER. The resulting nine experts and their names as used in our models are listed in Table 6.7(b).

The implicit experts are listed in Table 6.7(c), where the number of markers is listed instead of genes, analogous to the listing of ORFs instead of genes for the yeast domain. Numbers for the PFAM and PROSITE sources are not provided as those databases were queried only for the specific genes used in our mouse gene sets. For all three branches of the Gene Ontology used as implicit experts, the JIANG threshold was set to 16. Note that four resources, namely KEGG, GenMapp, GO Molecular Function and GO Biological Process, provide a characterization of the cellular role of proteins, thereby introducing a strong bias towards functional classification in the implicit expert data sources. Also, the PFAM and PROSITE sources have a high overlap with each other. These facts have an influence when using the **CONS** reliability assignment on the mouse gene sets, covered in the next section.

Explicit Expert Data Source	# Interactions
DRABKIN [DHHB05][3]	1314
BIND [BDW+01]	1310
DIP [XSD+02]	29
IntAct [HML+04] (matrix)	7440
MINT [ZMQ+02]	1081
RIKEN [SFK+01][4]	145
Total (including overlap):	11319

(a) Explicit Data Sources

Assay Type	Expert Name	# Interactions
3D-STRUCTURE	3DSTRUCT	12
AFFINITY-CHROMATOGRAPHY	AFFCHR	1554
IMMUNOPRECIPITATION	IMMUPREP	49
BLOTTING-ASSAY	BLOTASY	784
GO-PBIND-IPI	GOPBINDIPI	2608
OTHER	OTHER	183
TAP	TAP	325
UNKNOWN	UNKNOWN	27
Y2H	Y2H	416
Total (including overlap)		5958

(b) Explicit Experts

Attribute Type	Expert Name	# Markers
KEGG Pathway [KGH+06]	COKPATH	3450
PFAM Family [BCD+04]	COPFAM	n/a
PROSITE Domain [HSL+04]	COPROSITE	n/a
GenMAPP Pathway [DSV+02]	COGMAPP	8205
GO Cellular Component [ABB+00, EBK+05]	GO_CC	5868
GO Molecular Function [ABB+00, EBK+05]	GO_MF	5868
GO Biological Process [ABB+00, EBK+05]	GO_BP	5868

(c) Implicit Experts

Table 6.7: Data Sources and Distributions for Explicit and Implict Experts for MOUSE

Expert Name	CONS	BAYES UNSET	BAYES SRG0	BAYES SRG3	PRM UNSET	PRM SRG0	PRM SRG3
COKPATH	22	72	96	83	68	54	68
COPFAM	43	94	94	98	68	67	68
COPROSITE	41	84	94	98	72	51	72
COGMAPP	19	74	99	85	69	54	69
GO_CC	10	63	99	60	80	56	80
GO_MF	13	60	99	67	87	82	87
GO_BP	14	66	99	75	86	78	86
3DSTRUCT							
AFFCHR							
IMMUPREP	20						
BLOTASY							
GOPBINDIPI	70						
OTHER							
TAP							
UNKNOWN	20						
Y2H							
Corr. to CONS	1.0	0.95	-0.94	0.93	-0.65	-0.31	-0.65

Table 6.8: Expert Reliability Assignments $\mathcal{R}_e$ for MG122 MOUSE

6.3.2 Mouse Reliability Assignments

For the application of the LinOP and NoisyOR consensus likelihood functions in mouse, we use three of the strategies introduced in Section 6.2.2 for yeast, namely random (**RAND1, RAND2**), uniform (**UNIF5**), and agreement to consensus (**CONS**). In the section below, we consider two gene sets MG122 and MG449 with 122 and 449 markers for mouse genes, respectively. The **CONS** reliability values $\mathcal{R}_e$ for both gene sets are given for several experts e in Table 6.8 and Table 6.9. The empty values for some **CONS** rows signal that no interactions were found with that expert for the given dataset. For the MG122 gene set, the correlation r of these values to the values assigned by **RAND1** and **RAND2** is $r = 0.45$ and $r = -0.54$, respectively. For the MG449 dataset, the correlation of **CONS** values to **RAND1** and **RAND2** is $r = -0.47$ and $r = 0.24$.

The reliability values for BAYES and PRM are computed using inference on the learned models. Table 6.8 and Table 6.9 show the reliabilities assigned (on a 100 point scale) using Eq (5.6) for BAYES and Eq (5.9) for PRM for models learned on three versions of the dataset, **UNSET**, **SEMRELG0** and **SEMRELG3**, with **SEMREL** abbreviated to **SR**. Empty entries indicate the corresponding expert did not appear in the model. For both MG129 and MG449, BAYES **UNSET** and **SEMRELG3** show strong correlation with **CONS**, indicating that even in the case of **UNSET**, the BAYES model learns a similar function of consensus among experts. The reliabilities for BAYES **SEMRELG0** are all nearly perfect (0.94 to 0.99), showing no discrimination between experts. In Table 6.8, the PRM versions are strongly anti-correlated with **CONS**, due mainly to the high reliability assignments to the GO experts.

Expert Name	CONS	BAYES UNSET	BAYES SRG0	BAYES SRG3
COKPATH	23	79	94	86
COPFAM	37	93	95	97
COPROSITE	29	90	94	95
COGMAPP	17	82	99	90
GO_CC	8	71	99	69
GO_MF	9	59	99	64
GO_BP	14	70	99	75
3DSTRUCT				
AFFCHR	25			
IMMUPREP	34			
BLOTASY				
GOPBINDIPI	45			
OTHER	50			
TAP				
UNKNOWN				
Y2H	25			
Corr. to CONS	1.0	0.90	-0.82	0.90

Table 6.9: Expert Reliability Assignments $\mathcal{R}_e$ for MG449 MOUSE

6.3.3 Computing Mouse Consensus Likelihoods

As examples for the mouse genome, we consider two gene sets MG122 and MG449, with 122 and 449 gene markers, respectively. The genes in the MG122 set were hand-selected by our collaborators, Michael Rudolph and Dr. Peggy Neville of the University of Colorado at Denver and Health Sciences Center, based on a study of differential gene expression in mouse mammary gland during 11 timepoints spanning from virgin mouse to post-lactation [RMH$^+$03, RMP$^+$]. Considering the significantly over-expressed and under-expressed genes between the timepoints of Pregnancy Day 17 (P17) and Lactation Day 2 (Lac2), they identified 122 genes from related metabolic pathways affected during the Pregnancy/Lactation switch. We choose this gene set as an example which is very well studied by our colleagues with respect to observed changes correlated to a biological phenomenon. Given their knowledge about the system, we can then explore and compare the ability of our consensus likelihood functions to uncover relationships between the set of genes, which also have many well known interactions in the prior information sources. The MG122 gene set will be considered more closely in Chapter 8.

The gene set MG449 also originates from the Rudolph and Neville study of genes affected during the Pregnancy/Lactation switch. Using our colleagues' gene expression data measured for over 12000 mouse gene markers between P17 and Lac2, we obtain a list of the 2009 most significantly differentially expressed genes (either over-expressed or under-expressed). Among the 2009, we choose the 74 most highly differential genes using a $\log_2$ ratio threshold of 2.1 (or -2.1) and expand that list by considering all genes on the pathways in common with the 74 genes also in the list of 2009 genes, where pathway assignments are given by the GenMAPP [DSV$^+$02] resource. We refer to the

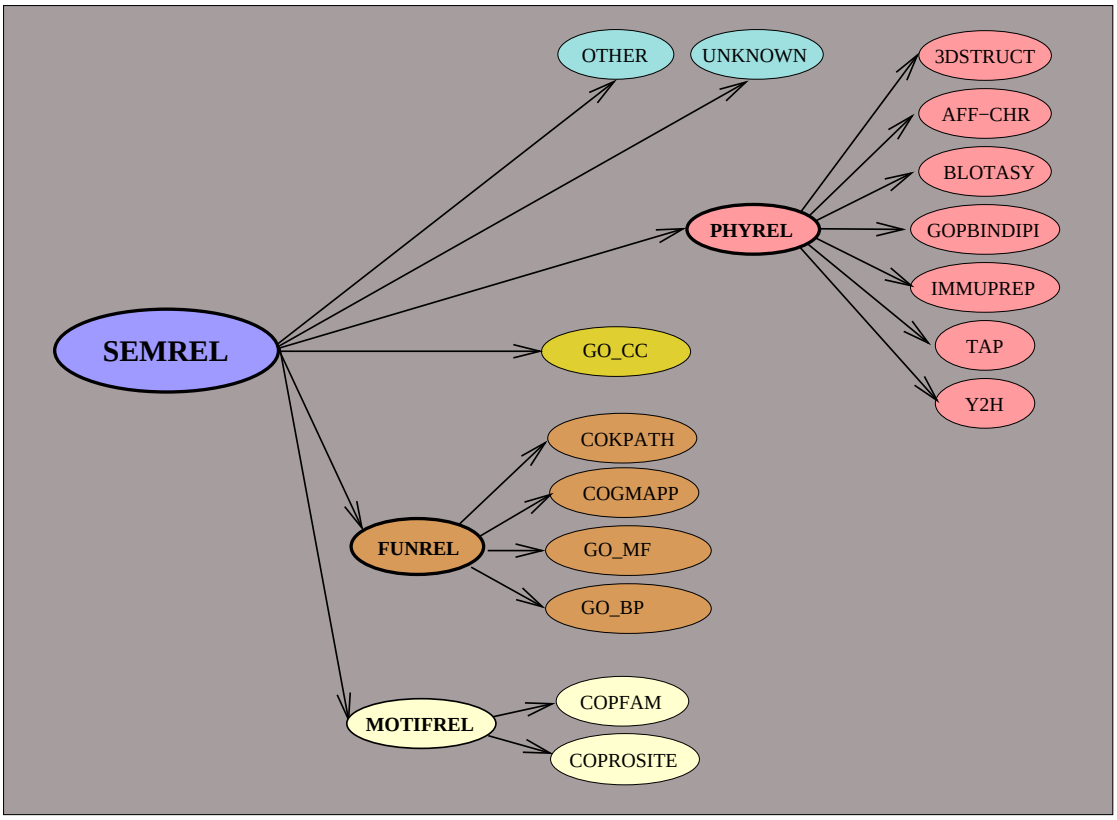


Figure 6.15: BAYES Model for Computing Consensus Likelihood in MOUSE

resulting set of 449 genes as MG449, which we also consider more fully in Chapter 8.

For the two datasets MG122 and MG449, we apply the consensus likelihood functions LinOP, NoisyOR, BAYES and PRM. For LinOP and NoisyOR, we use the four reliability assignments **CONS, UNIF5, RAND1** and **RAND2**. Probabilistic adjacency matrices $\mathcal{W}^{\text{LinOP}}$ and $\mathcal{W}^{\text{NoisyOR}}$ are created with a default edge probability $\mathcal{R}_e = 0.1$. For the BAYES and PRM models, the minimum number of occurrences for an expert to remain named is set to $MV = 5$ for MG122 and $MV = 100$ for MG449. The structure of the mouse BAYES model is given in Figure 6.15. The PRM model schema is similar to Figure 6.11 except using the experts and their corresponding attributes from Table 6.7. The maximum number of categories per attribute modelled in the PRM is set at $MA = 5$. Probabilistic adjacency matrices $\mathcal{W}^{\text{BAYES}}$ and $\mathcal{W}^{\text{PRM}}$ are created with a default edge probability $\mathcal{R}_e = 0.1$, using **UNSET, SEMRELG0** and **SEMRELG3** datasets together with the ratio or non-ratio technique for calculating $Pr(e_{ij})$.

We can compare the number of missing values in each dataset to have some idea of data sparsity and the consequences for inference. For BAYES, there were 10 variables (4 hidden) in the model for MG122, yielding a potential of 82560 possible values. The dataset **UNSET** contains 38782, **SEMRELG0** contains 46163 and **SEMRELG3** contains 41259 for MG122. The corresponding datasets for the PRM with 135192 variables (40 CPDs with 66 parameters) have 39348, 46728, and

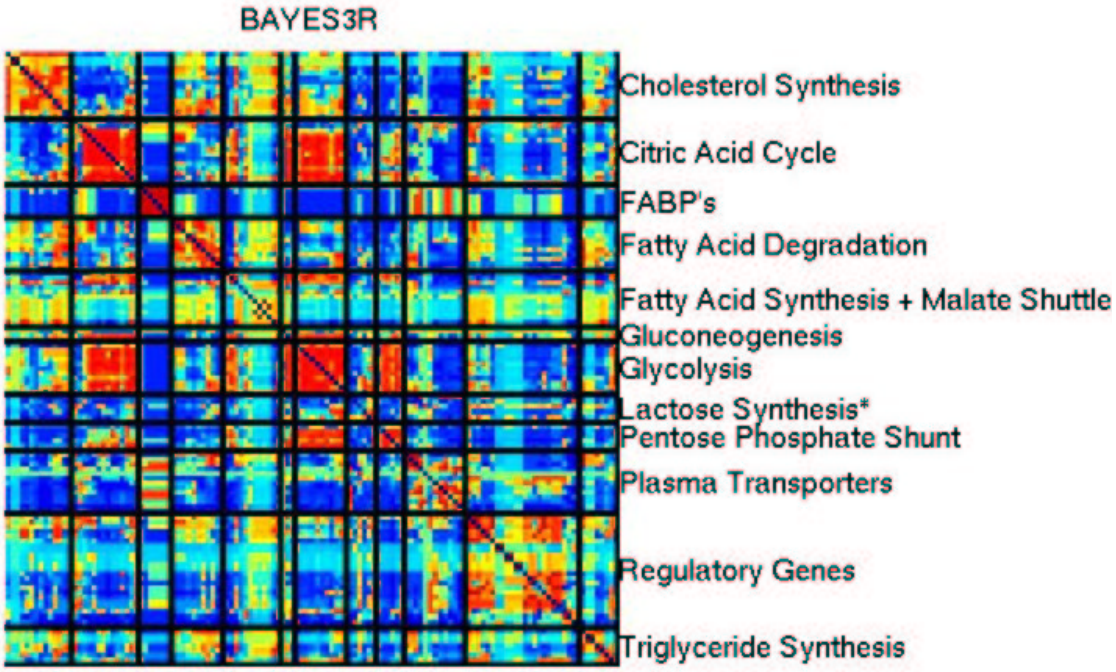


Figure 6.16: Example of correspondence of consensus likelihood to functional categories in MG122 MOUSE dataset (using BAYES3R)

41824 values for **UNSET**, **SEMRELG0** and **SEMRELG3** respectively. Results using the PRM consensus likelihood are not show for MG449. To learn either the BAYES or PRM model parameters from these datasets, EM is applied using BDeu priors with $N' = 1$. We set the maximum number of iterations of EM to 10, with 10 interior iterations of loopy belief propagation per EM iteration, as in the yeast examples.

Figure 6.17 shows the consensus likelihood matrices for all combination functions on MG122, including $\mathcal{W}^{\text{STRING}}$ using the probabilities from the STRING database [vMHJ+03, vMJS+05]. The dark red boxes along the diagonal in the (deterministic) matrix labelled MG122 shows membership in functional categories, as assigned by our colleagues. Similar conclusions can be drawn as in the yeast results. The NoisyOR matrices are generally higher probability (more red) than the corresponding LinOP matrices and are thus more susceptible to incorrect reliability assignments. The STRING matrix has very few non-zero entries.

Select functional categories are captured as hotspots along the diagonal for many of the matrices. The effect is more pronounced than in the yeast examples, owing in part to the strong bias towards functional categorization in our collection of experts. The **UNIF5** reliability assignments are not as effective as the **CONS** reliability, with respect to recapturing functional categories. This can be seen by the higher probabilities (hotter colors) off the diagonal. By the same criterion, the **RAND1** and **RAND2** assignments are not much worse than **UNIF5** for the LinOP function but suffer greatly under the NoisyOR function. In contrast to the yeast results where all BAYES and PRM matrices

were predominately red, both consensus likelihood functions provide a reasonable spread across the range from 0 to 0.5 in the mouse gene sets. The BAYES **SEMRELG0** versions are the exception, where both the ratio and non-ratio matrices are significantly red. This indicates that the policy for assigning SEMREL to true with only one expert assertion is too promiscuous for this mouse geneset, since the other BAYES versions were able to recover the functional categories. Presumably the improved behavior of BAYES and PRM in mouse is a result of the fact that the collection of experts available in the yeast did not have the type of overlap that the mouse experts exhibit, thereby not providing enough force of evidence in the data for the BAYES and PRM model learning algorithms to uncover discriminations between different gene pairs.

An example of correspondence of the consensus likelihood with the functional categories assigned by our colleagues is illustrated for BAYES3R in Figure 6.16. Black lines divide the rows and columns into the 12 functional categories. Note the presence of subgroups within the Regulatory Genes category, indicating protein complexes within that group. Also, note the off-diagonal cluster of Glycolysis for Citric Acid Cycle, capturing the known fact that they are closely related biological functions. The same is true for Glycolysis and the Pentose Phosphate Shunt; in Chapter 8 we will explore these points further placing each of these functions in context to the others through the use of visualization tools.

The MG449 consensus likelihood matrices are shown in Figure 6.18. The matrix labelled MG449 shows functional category membership according to the GenMAPP categories used to extract the gene set from the full 2009 set initially. The use of GenMAPP as an expert information source as well as the criterion for ordering the matrix results in more coherent diagonal elements for all matrices. The most remarkable performance is the BAYES3R function which very strongly recreates the GenMAPP functional categories. The probabilities are actually rather precise to that categorization since the off-diagonal elements are also fairly cool colors, indicating low probabilities. Viewing in more detail the functional category assignments for the most populated categories against BAYES3R, Figure 6.19 shows boxes of red on the off-diagonal, indicating related functional categories. For example, MITOTIC CELL CYCLE is found to be related to REGULATION OF CELL CYCLE, with slight correspondence to genes in CELL CYCLE.

Interestingly, the **RAND2** matrices for both LinOP and NoisyOR favorably recreate the functional categories. Recall that **RAND2** is formulated using reliabilities with correlation $r = 0.24$ to the **CONS** reliability assignments. In Chapter 8, we use the **CONS** NoisyOR matrix to create a graph whose edges are the high-probability interactions and use that graph within a visualization tool to provide context for gene expression changes in the Rudolph and Neville mouse mammary gland expression dataset.

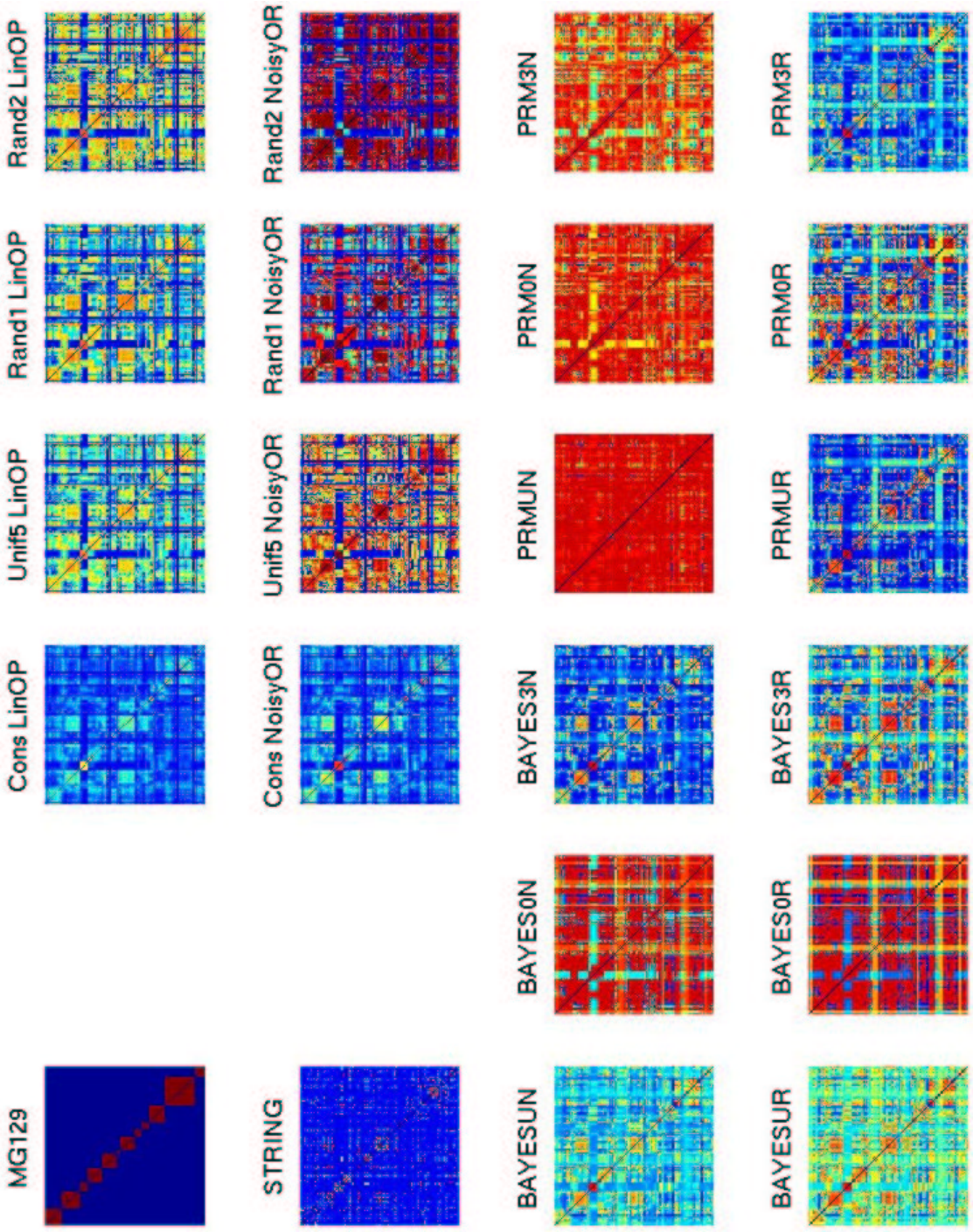


Figure 6.17: All Consensus Likelihood Matrices for MG122 MOUSE dataset

.

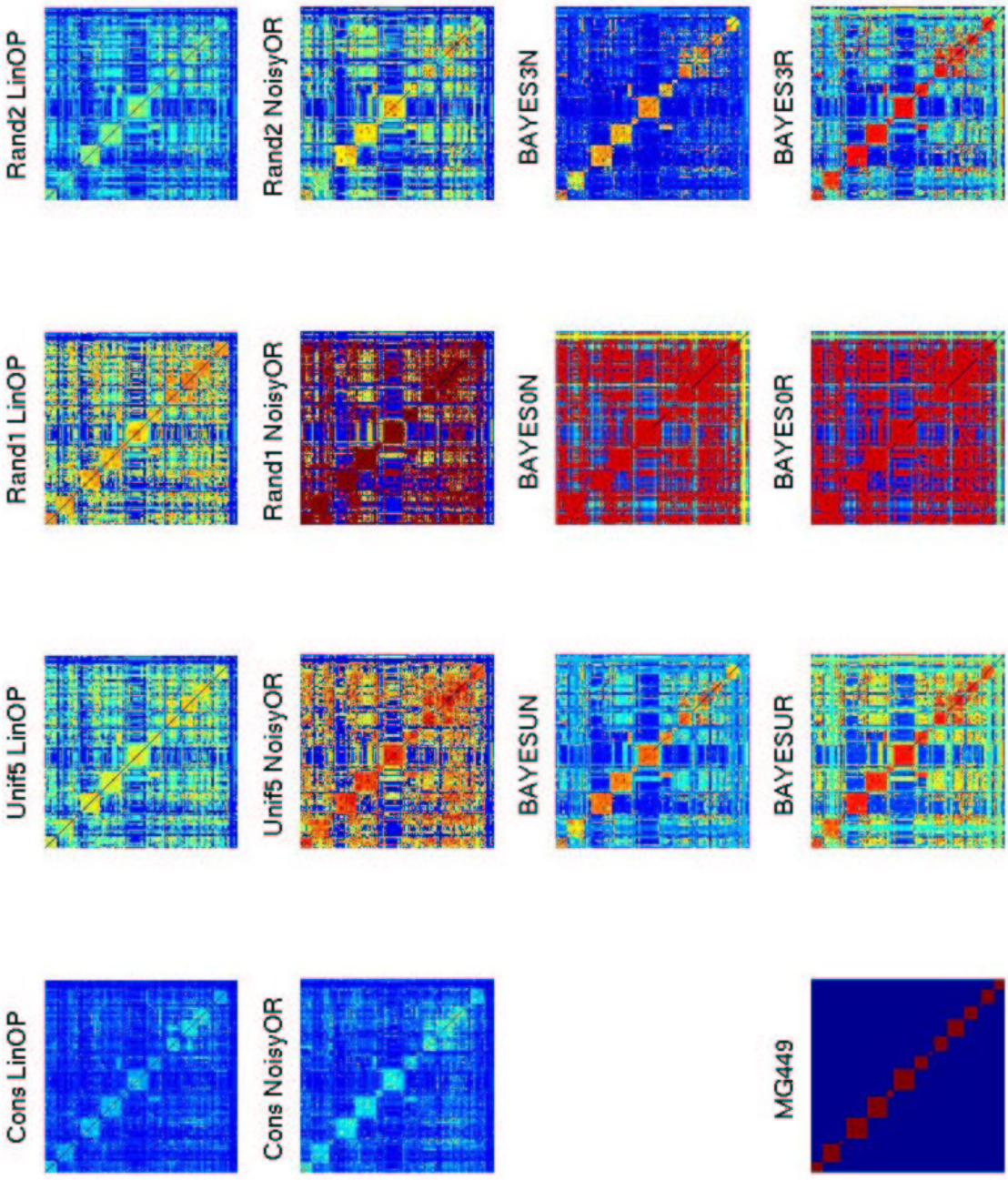


Figure 6.18: All Consensus Likelihood Matrices for MG449 MOUSE dataset

.

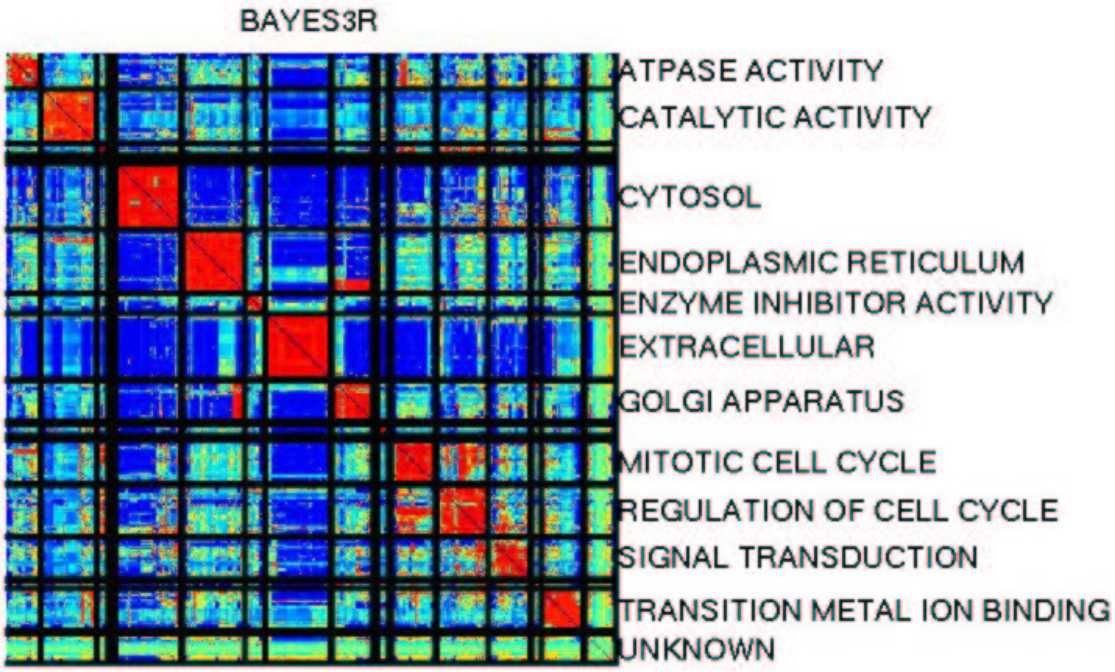


Figure 6.19: Example of correspondence of consensus likelihood to functional categories in MG449 MOUSE dataset (using BAYES3R)

Chapter 7

Learning Gene Regulatory Networks

In this chapter, we demonstrate how a consensus likelihood, found by any of the four methods of Chapter 5, is used as a structural prior during Bayesian network learning. We present the experimental setup for two domains: a real-world model of ICU ventilator management (ALARM) [BSCC89], as well as a real-world biological example using the yeast genome.[1] We describe the general experimental setup and evaluation criteria before presenting results of Bayesian network learning in both domains using the consensus likelihood functions. Our objective is to demonstrate how using a prior distribution over structures can significantly increase the speed and quality of search for finding high scoring Bayesian Networks.

7.1 Experimental Design

Bayesian network learning, as presented in Chapter 3, involves searching among the space of possible structures of the model, scoring each against prior knowledge of structure and proper fit to dependencies contained in a dataset. The complete learning framework is shown in Figure 7.1. Prior structural knowledge is used by a combination algorithm to create a consensus likelihood matrix which is then used to formulate a prior over structures, as described in Chapter 5 and Chapter 6. Together with a dataset of samples for each of the variables of the model, the given prior over structures is used by a search and score algorithm to generate high scoring models. We present the details of the learning scheme in the next section, followed by a list of the evaluation criteria.

[1] The mouse genome was not used for Bayesian network learning due to the limited availability of publically available expression data.

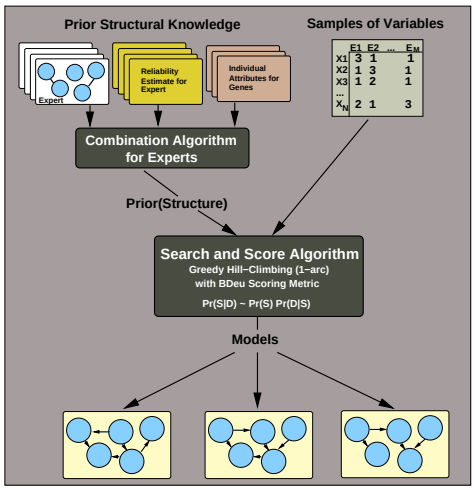


Figure 7.1: Complete Bayesian Network Learning Framework

7.1.1 Learning Algorithm Details

To implement the Bayesian network learning algorithm, we use a greedy hill-climbing search over all possible 1-arc changes to the graph (addition, deletion or reversal of an edge) together with the BDeu metric (Bayesian metric with Dirichlet priors, likelihood equivalence and uniform joint) [HGC95]. The BDeu score uses the Bayesian scoring metric:

$$Pr(S \mid D) \propto Pr(S)Pr(D \mid S),$$

which calculates the posterior probability $Pr(S|D)$ of the structure S given the data D as the product of the prior over structure $Pr(S)$ and the marginal likelihood $Pr(D|S)$ of the data given the model. Use of the BDeu version of the metric allows the likelihood term, which involves integrating over all possible parameterizations compatible with the structure, to be calculated in closed form.

Several conditions must be met in order to apply the BDeu metric. First is that the metric requires a complete dataset (*i.e.,* no missing values) of discrete variables. For the ALARM model, the variables are already discrete and a complete dataset can be sampled from the target model. In yeast, the expression data is continuous and contains missing values. Thus, we discuss one method in Section 7.3.1 we have developed for imputing missing values in cases, like the yeast domain, where the data is not missing at random. For the requirement of discrete data, Appendix A examines a large host of discretization algorithms for the case where (some of) the variables are continuous-valued, including three techniques we developed that outperform existing methods. For purposes of this chapter, we briefly describe in Section 7.3.1 the two techniques used for the yeast experiments.

The second requirement for calculation of the BDeu once given a complete, discrete dataset is that the user must specify a parameter N' for the equivalent sample size. This parameter controls the strength of the user's belief in the parameter prior. A value of $N' = 10$ is commonly found in experimental validations [FNP99, Chi96]. In our experiments, for ALARM we use $N' = 1$ (low confidence in prior); for yeast, we consider both $N' = 1$ and $N' = 10$ (reasonable confidence).

Given a dataset of samples and a structural prior, we generate a collection of models by restarting the greedy hill-climbing search from multiple initial models sampled from a given prior distribution over structures. The structural prior is then also used *during* learning to score each model. One set of starting models is sampled *uniformly at random* from the space of models. We refer to this set of models as Random. During learning, implementing the use of a uniform structural prior is accomplished by ignoring the probability over structure term $Pr(S)$ when calculating the score. A second set of starting models is sampled from the space of graphs using an *informed* structural prior created from one of the consensus likelihood functions of Chapter 6. We refer to such sampled models as Informed models. During learning, the same informed structural prior is used in the scoring function to guide the search.

Given a starting model sampled from some distribution, the greedy hill-climbing search proceeds to maximize model score by applying 1-arc changes, until no further improvement can be made. We thus create a set of final models by collecting the results of the searches from each starting point. In other applications of Bayesian network learning, the maximum scoring final model (or some combination of the set) is typically chosen as the output of the algorithm. Here we use the set of individual models to compare performance the various consensus likelihood functions. We compute each evaluation metric listed below on a set of Informed models versus a set of Random models.

Ideally we would like a large set of Informed and Random models to accurately sample the distribution of model scores. However, for yeast we are comparing 24 consensus likelihood functions using two N' settings and two different discretization techniques, which results in a large number of experiments. We must therefore employ a number of techniques for controlling the tractability of our study. First of all, to make learning in the ALARM model comparable with the types of data available in the yeast domain, we chose similar (tractable) dimensions for the learning problem in both domains. The ALARM model has 37 variables and successful reconstruction typically uses 5000 to 10000 samples [CH92, HGC95]. However, the currently publically available expression data we obtain for yeast is limited to fewer than 2000 samples. Thus for the ALARM domain, we choose to use 2000 samples on the 37 variables for learning to mimic the dimensions for the yeast domain, where we choose to use 50 genes (IDEKER50 from Section 6.2.3) with the dataset of 1738 samples.

To further manage the complexity of a search, we limit the search space by controlling the number of parents for any node, using a max_fan-in=4 for ALARM and max_fan-in=3 for yeast. Also, for learning in yeast, the number of edges in the graph were constrained to be between 50 and 100. Using the entire collection of strategies for controlling the complexity of the search, we found that using between 50 and 100 models (either Informed or Random) is a reasonably sized set for both ALARM (50 models) and yeast (50 or 110 models), allowing a fairly accurate distribution of scores without incurring unmanageable computational expense. In the next section, we provide a figure demonstrating that using only five models provides the same qualitative results as using the full set of 100 models in ALARM.

7.1.2 Individual Evaluation Criteria for Structural Priors

We wish to demonstrate the benefit of including structural prior information during learning. We claim that use of informed structural priors can help Bayesian network learning algorithms find higher quality models and speed the time to find those models during search. In order to evaluate those claims for each individual consensus likelihood function used as a structural prior, we develop three *individual evaluation criteria* to measure quality and speed of the search using an informed versus using an uniformed structural prior. Figure 7.2 demonstrates an ideal example of each of the three evaluation criteria listed below for a set of Random models versus a set of Informed models. The behavior of the criteria will change depending on the particular consensus likelihood function used to derive the structural prior.

C1: Compare quality of start/end models

The Bayesian score consists of two terms, the prior over structures $Pr(S)$ and the marginal likelihood of data $Pr(D|S)$. Thus, maximizing both terms will maximize the total model score. Recall that during learning with the Random models, the $Pr(S)$ term is ignored in calculating the score. For display purposes, the prior is calculated for the Random models using the same informed structural prior used during the Informed searches, which allows us to visualize the correspondence of the Random models with the knowledge encoded in a particular structural prior.

The top plot in Figure 7.2 illustrates the difference in quality of the start and end models for search using either Random (red plus) or Informed (blue circle) models by plotting the (log) prior on the vertical axis versus the (log) likelihood on the horizontal axis. Each point in the plot represents one model, where ideally we want points in the upper right corner. The clouds of points to the lower left shows the components for the Bayesian score for the *starting* models of the greedy hill-climbing search. Analogously, the point clouds at the upper right of the plot show the score components for the *final* models of the search, as the score during search improves mainly in terms of data likelihood.

We use this criterion to support our claim that *sampling from an informed structural prior distribution results in higher quality models than sampling from an uninformed distribution.* We show that we begin exploration at a higher peak of the objective function landscape. Note that the example plot exhibits precisely this idealized behavior, where the collection of Informed starting models score higher than the corresponding Random starting models on both axes, thereby scoring higher on the overall score. Comparing the clouds of final models, we see that the Informed models do not cause the search to become restricted to a certain area of the objective function, rather the plot shows that both Informed and Random searches eventually arrive at a comparable peak of the objective function. Examining only the scores for the final models might lead us to the conclusion that using an informative prior provides no advantage to using the uninformed prior. However, the fact that search begins at a higher likelihood prompts us to create two more evaluation criteria which monitor the efficacy of those searches across all iterations.

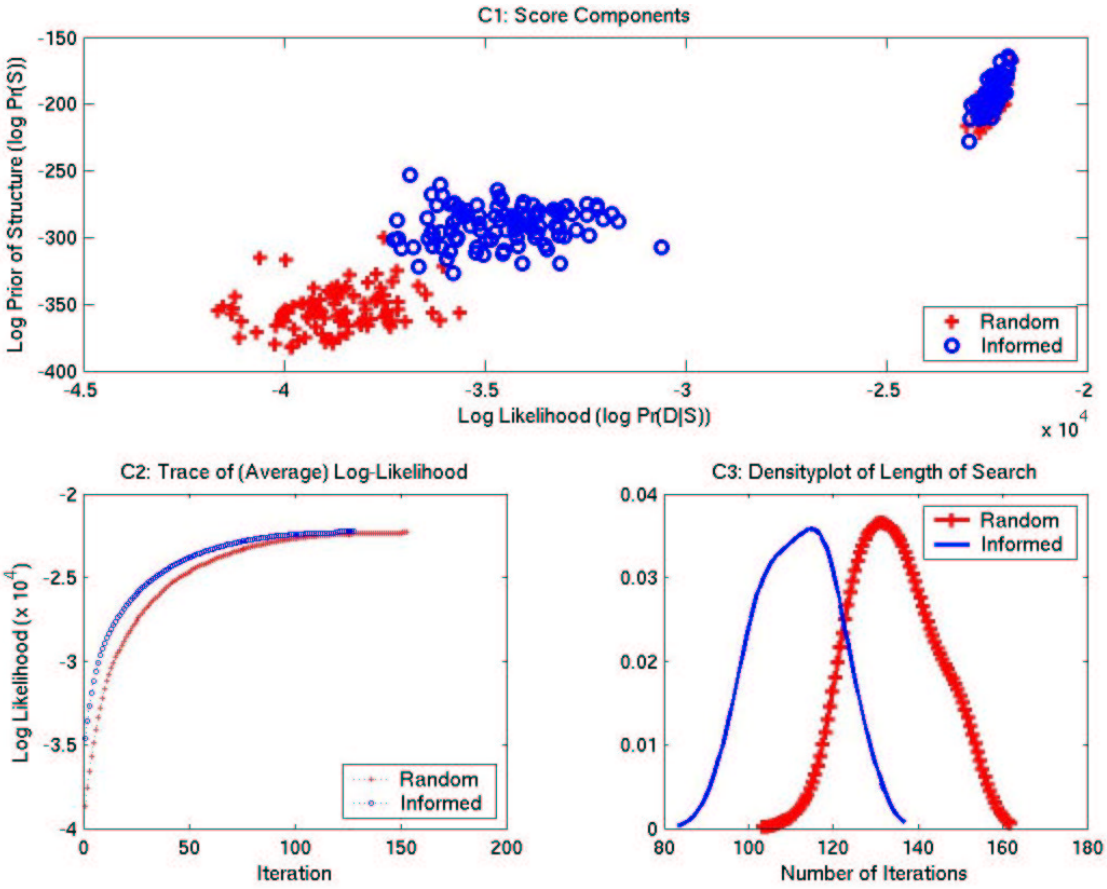


Figure 7.2: Example Illustration of Individual Evaluation Criteria

C2: Compare quality during search

Though the first criterion is used to demonstrate that Informed models start at higher scores, it also shows that the final Informed and Random models have similar scores. Thus, the searches starting from Random models are able to recover the difference at some point during search. The question remains where along the search did the gap in performance close. We investigate this question by looking at the likelihood $Pr(D|S)$ for each model at each iteration in the search, averaged over the set of models. The bottom left plot of Figure 7.2 shows the iteration along the horizontal axis and the average (log) likelihood along the vertical axis. Again, the performance of the Random models is shown in red while the performance of the Informed models is shown in blue.

We use this criterion to support our claim that *using an informed structural prior distribution results in higher quality models during search than using an uninformed distribution.* As can be seen from the plot, the Informed models consistently have higher (average) likelihoods than the Random models throughout the search. Thus, in this example, it is not the case that searching using Informed models quickly loses its advantage. In the later iterations, the two curves do converge, yet note that the curve over Informed searches ends at fewer iterations than the curve for Random searches. Since these curves represent averages over the search from all starting models, we present a third criterion that explores the distribution of the number of search iterations.

C3: Compare length of search

We introduce a third criterion to investigate the length of search using models sampled from Random and Informed structural priors. The bottom right plot of Figure 7.2 shows a density plot of the distribution of the number of search iterations. Treating the value for the number of iterations as a random variable, the curve represents a probability density function, such that the area under the curve is equal to 1.

We use this criterion to support our claim that *using an informed structural prior distribution during learning results in faster searches than using an uninformed distribution.* This claim is particularly important in problems with many variables and few samples since we would like to explore the search space as fully as possible, which with greedy hill-climbing means more search restarts. Faster individual searches mean more models can be explored for a given amount of computation time. In the example of Figure 7.2, we see a clear separation between the Informed distribution, with a mode of 115, and the Random distribution, with a mode of 132. Combined with the results from the previous two criterion, we can conclude in this example that Informed searches are shorter since we begin closer to better solutions.

7.1.3 Relative Evaluation Criteria for Structural Priors

The three criterion listed in the last section investigate the performance of a single consensus likelihood function as a structural prior. To demonstrate the relative performance of all methods, we develop several other graphics derived from criterion C1 and C2 which were described in the previous

section. Using the following set of five *relative evaluation criteria*, we provide a simultaneous display for all the various consensus likelihood functions. This allows a quick comparison of the performance of those functions with respect to providing priors for Bayesian network learning.

Boxplots of Log-Likelihood of Starting Models

In the comparisons of the score components by criterion C1, the prior over structure term is computed merely for display purposes since during the actual searches the Random models are scored based solely on the likelihood term. Use of an informed prior clearly affects the score through the term $Pr(S)$. However, the informed prior can also affect the score through the likelihood term since the initial model is sampled from the informed prior distribution. The likelihood of the starting model dictates where the search begins on the objective function landscape, *i.e.,* how close or how far away is the starting point of learning from a (local) maximum of the objective function.

Viewing the distributions of the (log) likelihoods of the starting models as Boxplots allows a qualitative assessment of the advantage conferred by using informed structural priors. An example is shown in Figure 7.1.3(a) for distributions of log-likelihoods using each consensus likelihood function, as labelled along the vertical axis. A box contains lines at the lower quartile, the median, and the upper quartile of the distribution. Lines extending from the box show the remaining range of the data with outliers marked. In this example, the distribution of likelihood values for the Random models appears at the top. The median value of all methods is greater than the median value for Random. Highest performance is demonstrated by the method labelled STRING, showing little overlap with the Random boxplot. This indicates that the likelihood term for nearly all starting models from the STRING prior distribution was greater than the likelihood term for starting models from a uniform prior.

Average Log-Likelihood Across First Ten Iterations

Our second relative evaluation criterion derives from the individual traces of average log-likelihood shown by criterion C2. Since typically the curve for Informed is greater than the curve for Random throughout all iterations, we can get an idea of the relative performance of the techniques by studying the first few iterations. Figure 7.1.3(b) shows an example of the first 10 iterations of greedy search for a few informed priors. The first point to note in this example is that Random is worse than all methods. Also the different classes of consensus likelihood are partitioned by color. For example, the LinOP versions are depicted in blue.

This criterion presents several other issues which motivate the next set of relative criteria. One question that arises is how statistically significant are the differences observed between the curves. The next criterion addresses this issue. The criterion following that explores how sloped are the curves, in other words, how quickly is the difference lost between an Informed curve and the Random curve.

Significance of Difference between Log-Likelihood Distributions per Iteration

Search is performed from multiple restarts, using models sampled from both informed priors and uniform priors. The previous criterion showed the *average* likelihood of those models per iteration for the first 10 iterations. We can instead compare the *distribution* of log-likelihoods for those models (of which the previous graph showed the average) at each iteration in both the Informed and the Random setting using a two-sample t-test, testing the one-tailed hypothesis that the mean log-likelihood for the Informed searches is greater than the mean log-likelihood for the Random searches.

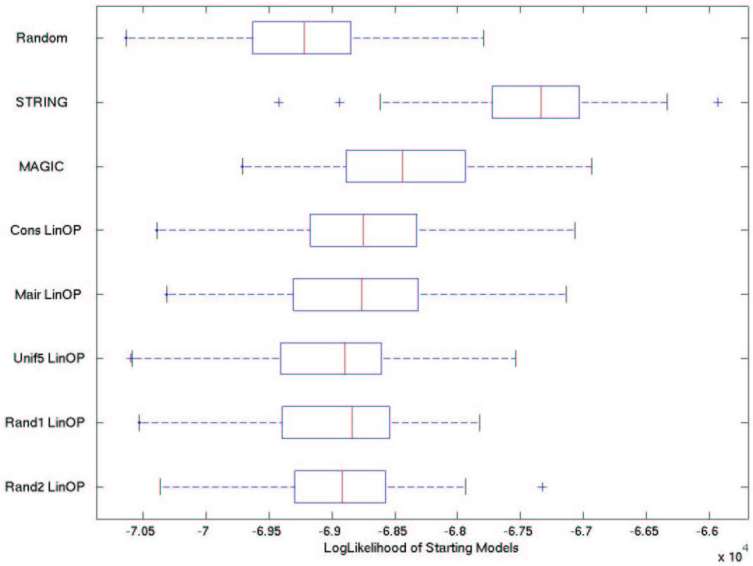
Figure 7.1.3(a) shows a plot of the p-value for a one-tailed t-test between the two distributions, per iteration. Lower p-values indicate that the Informed searches are considering models with higher log-likelihood than the Random searches at the same iteration. The vertical axis is on a log-scale to emphasize the region below a significance value of 0.05, shown by a horizontal red line. Note that in this example, the significance value of the t-test is well below the threshold of 0.05 until roughly iteration 50, then the Informed searches become worse than the Random searches, causing the significance value to cross the 0.05 threshold. The significance returns to values below the threshold up until roughly iteration 130, where the Random searches again outperform the Informed searches. The behavior on the tail end of the curve might be due in part to the fact that several of the searches have completed by that iteration so the t-test comparison is on fewer models. This example shows one Informed curve versus a Random curve; in general, we will display a panel of figures, one for each Informed prior.

Difference of Average Log-Likelihood per Iteration

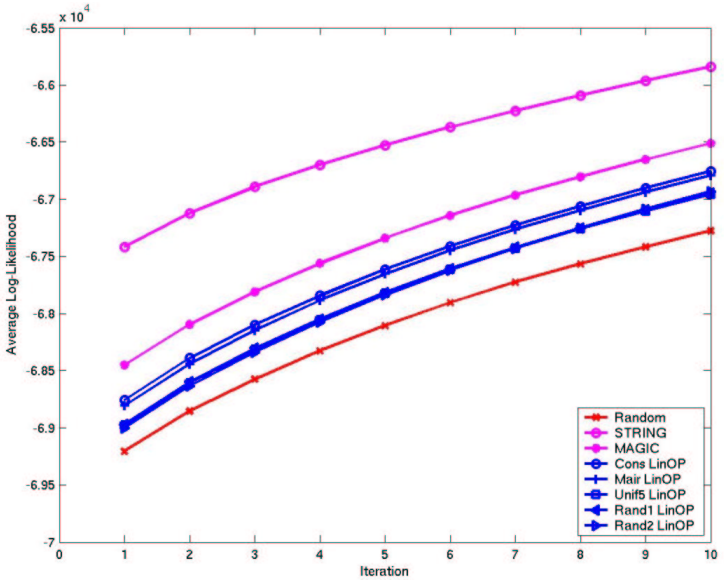
To examine the rapidity with which the Random curve approaches the Informed curve for a particular consensus likelihood prior, we can compute the difference between each trace of average log likelihood using an Informed prior from the trace using the Random prior. At each iteration, we subtract the average Random from the average Informed likelihood and plot this residual. The criterion thus becomes a scaled version of individual criterion C2. Figure 7.1.3(b) shows an example where the Random curve is shown at the value 0 across all iterations. The vertical axis shows the difference in average log-likelihood for each iteration. The point of interest is the slope of the remaining curves, illustrating the speed at which the Random searches recover from their deficit.

Difference in Average Log-Likelihood Averaged Across Iterations

Observing the trend of differences across iterations using the previous criterion suggests a way to rank the performance of the methods. Our fifth relative criterion averages the differences calculated above across all iterations. Figure 7.1.3(c) shows an example of the methods ranked by the average difference. The magnitude of the average appears along the horizontal axis while the methods are spaced along the vertical axis by rank, merely for visual convenience. In this example, clearly STRING and MAGIC outperform the others.

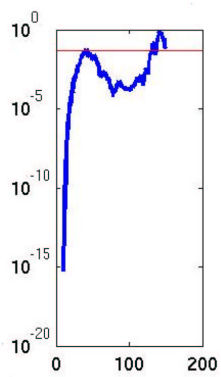


(a) Boxplots of Log-Likelihood of Starting Models

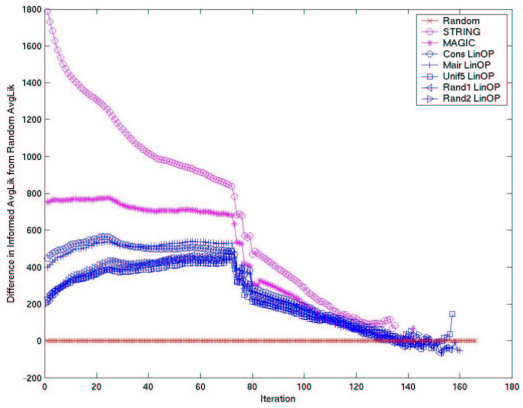


(b) Average Log-Likelihood for First Ten Iterations

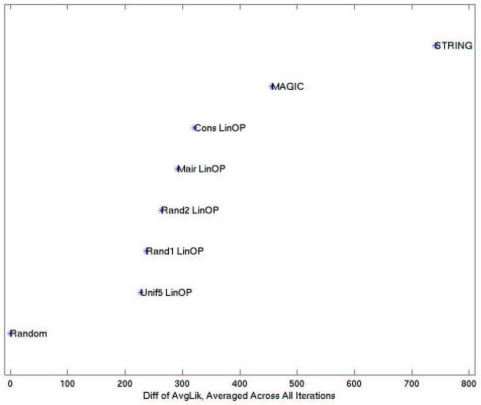
Figure 7.3: Example Illustration of Relative Evaluation Criteria



(a) Significance of Distribution of Log-Likelihood per Iteration



(b) Difference of Average Log-Likelihood per Iteration



(c) Difference of Average Log-Likelihood Averaged over Iterations

Figure 7.4: Example Illustration of Relative Evaluation Criteria, continued

7.2 ALARM Bayesian Network Learning

The ALARM model is a well-known benchmark for Bayesian Network learning, complete with structure and parameters from which a dataset can be sampled. We use the ALARM model as a testbed, as described in Section 6.1, since we know the behavior of Bayesian network learning from previous studies using that model, thus we can then investigate the effect of introducing structural priors with various types of error.

We sample the ALARM Bayesian network to generate a dataset of 2000 samples for the 37 variables. As described in Section 6.1, we create three types of experts (Positive, Negative, and Indirect) and five different sets of reliability assignments. Recall that each assignment is represented by a three integer string N.I.P where N, I, and P represent the reliability levels for the set of Negative, Indirect, and Positive experts, respectively. Low, Medium, or High reliability is represented in the integer string by a 0, 1, or 2, respectively. The meaning of each assignment is repeated below, appearing in rough order of decreasing correctness of reliability assigned:

- **0.0.2:** Positive experts correctly assigned High reliability
- **0.1.2:** Reliabilities reflect intuitive understanding of correctness
- **0.2.0:** Indirect experts given most influence
- **1.1.1:** All experts assigned uniform reliability
- **2.1.0:** All experts assigned incorrect reliability

For the comparisons, 100 models were sampled from an informed structural prior distribution, formulated using the LinOP or NoisyOR consensus belief function combined with one of the five reliability assignments, to create a total of 10 sets of Informed models. Additionally 100 models were sampled cocurrently from a uniform prior over structure, each set of which we denote as Random. Greedy hill-climbing search was applied to each of the starting models, recording the score components $Pr(S)$ and $Pr(D|S)$ at each iteration of the search. In the next section, results are shown first for select results, then for all methods simultaneously in a series of figures, one for each criterion; the complete set of results organized instead by individual combinations are included in Appendix E.

7.2.1 ALARM Results Using Individual Evaluation Criteria

We evaluate the five reliability assignments for NoisyOR and LinOP using the individual evaluation criteria described in Section 7.1.2. For comparison of each consensus likelihood prior, a different set of Random models was generated cocurrently with the Informed models. Figure 7.5 shows all three criteria for the strongest performer, namely NoisyOR together with the reliability assignment 0.0.2 which weights the Positive experts most heavily. From the top plot (C1) showing the score components for the starting and ending models, we can see the strong separation between the clouds of points for the Random and Informed starting models, demonstrating that the initial scores for the

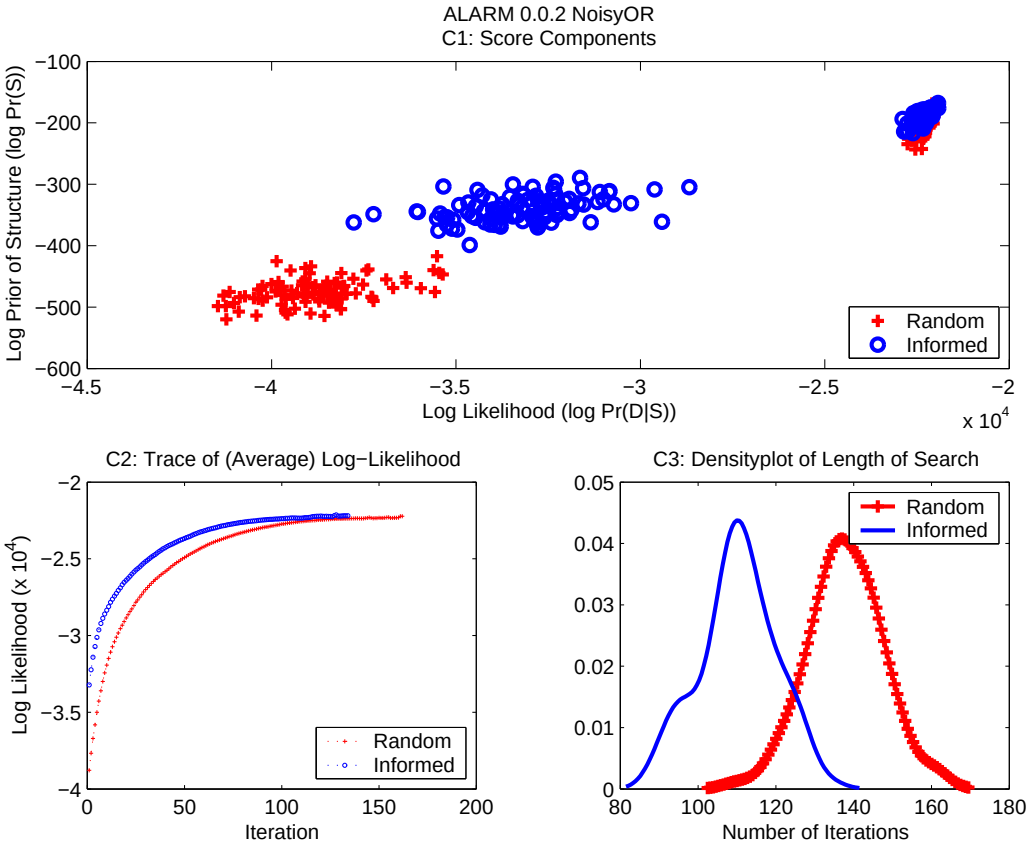


Figure 7.5: Example of Individual Criterion, using ALARM 0.0.2 NoisyOR

Informed models are higher. Also, the curves are well separated in the bottom left plot (C2) showing average log-likelihood per iteration. The distribution of number of search iterations (C3) shown in the lower right plot are also distinct, with the Informed searches having fewer iterations than the Random searches on average. All three plots show results for 100 Informed and 100 Random.

Given the large search space of graphs, one question might be whether sampling from only 100 models is sufficient. Figure 7.6 shows the component scores graph for five models in the left panel and 100 models in the right panel. As can be seen from this graph, using five models provides the same qualitative results as using the full set of 100 models in ALARM. It is reasonable to assume this result will extrapolate.

Viewing similar plots of the three criteria for all method, the overall conclusion is that as the quality of the reliability assignments decreases, the disparity between the length and the quality of the search lessens. Figure 7.7 demonstrates this conclusion using LinOP and NoisyOR for the least correct reliability assignment, 2.1.0. Note that in both the C2 and C3 sets of graphs, Informed search has lost the advantage. In fact, for the NoisyOR plots, the Informed results are in fact worse than Random. This reflects the observation made earlier that NoisyOR is very susceptible to incorrect reliability assignments.

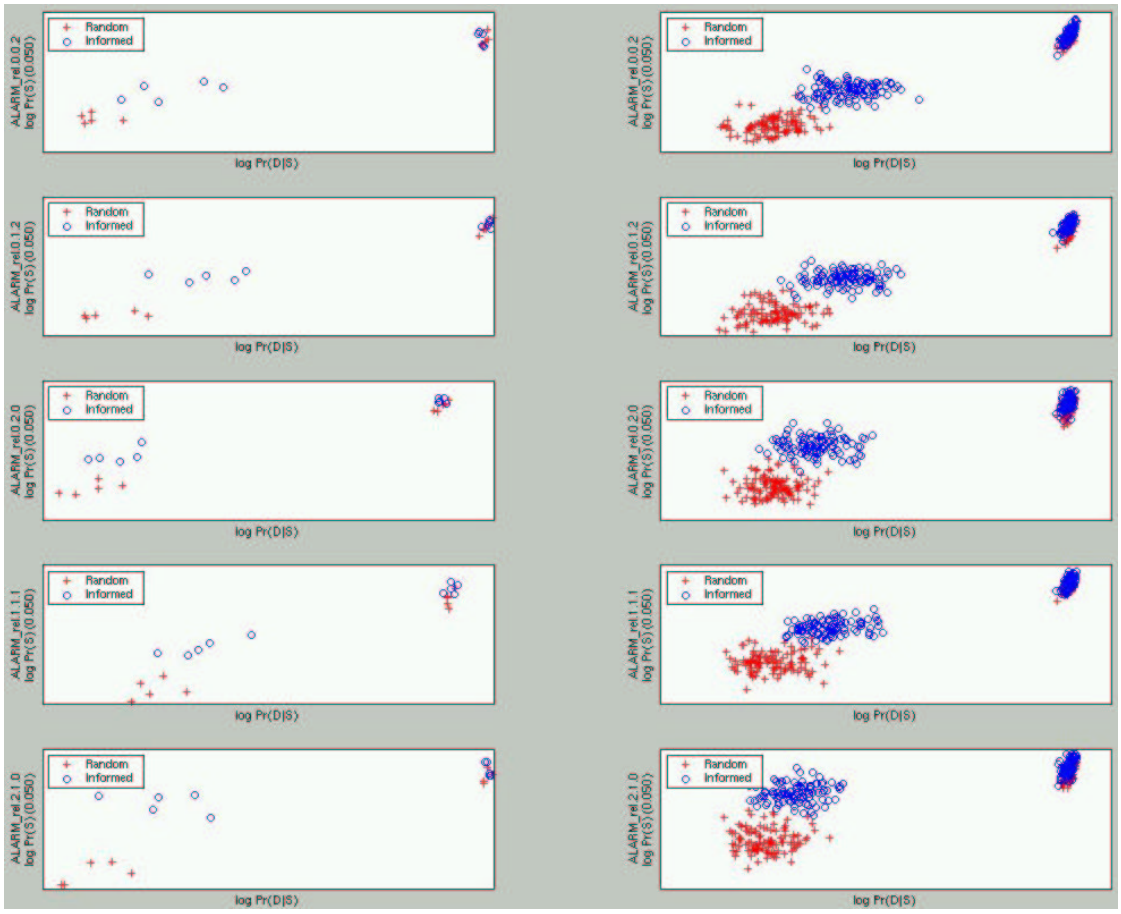


Figure 7.6: Using 5 versus 100 models for ALARM

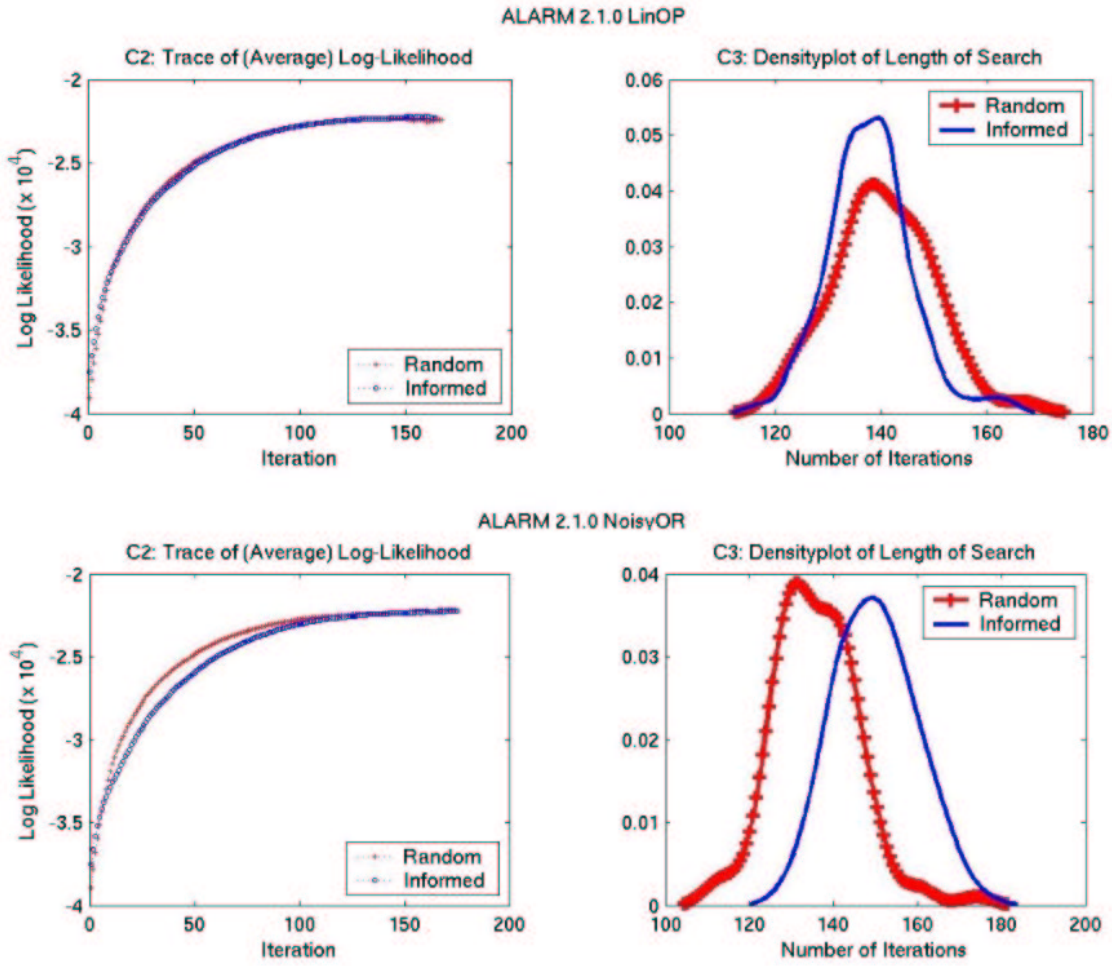


Figure 7.7: ALARM 2.1.0 LinOP versus NoisyOR

Simultaneous display of results for all methods using the three individual evaluation criteria are shown in Figure 7.8 through Figure 7.10, where the axis labels have been removed for visual clarity. Red markers indicate Random models while blue markers indicate Informed models. Figure 7.8 shows results for the components of the score. The horizontal axis is the log-likelihood $Pr(D|S)$ while the vertical axis is the log-prior over structures $Pr(S)$. Figure 7.9 shows the results for the average log-likelihood (vertical axis) per iteration (horizontal axis). Figure 7.10 shows the distribution of the total number of iterations per search. The scale of the horizontal axis denotes the number of iterations from 50 to 200. Moving from top to bottom in any of the figures, we see the curves in blue (Informed) steadily encroach upon (and sometimes overtake) the curves in red (Random). This illustrates that the performance of the Informed models versus the Random models degrades as the error in reliability assignments increases.

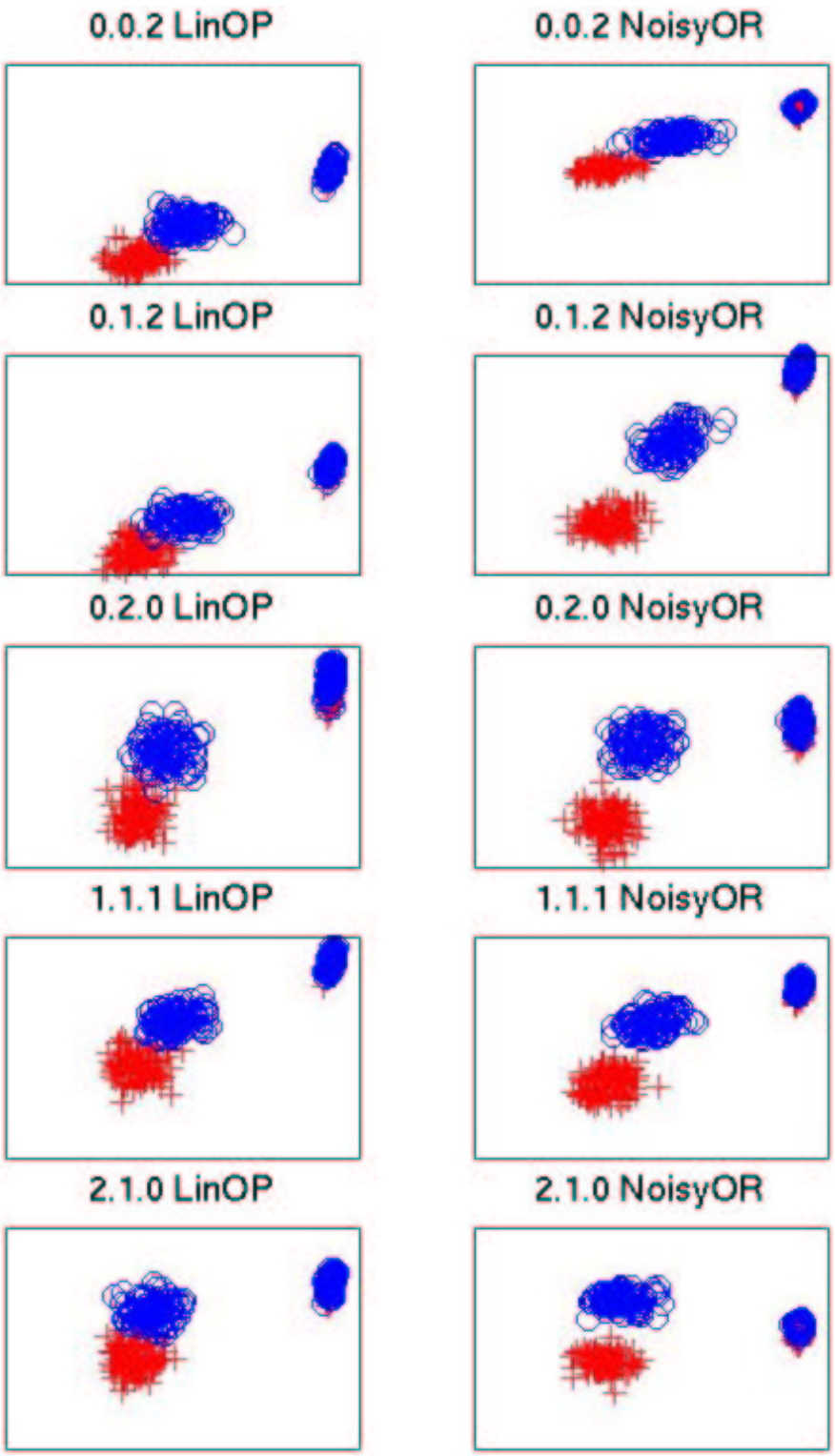


Figure 7.8: ALARM Criterion 1 (Score Components for Starting and Ending Models) for All Methods

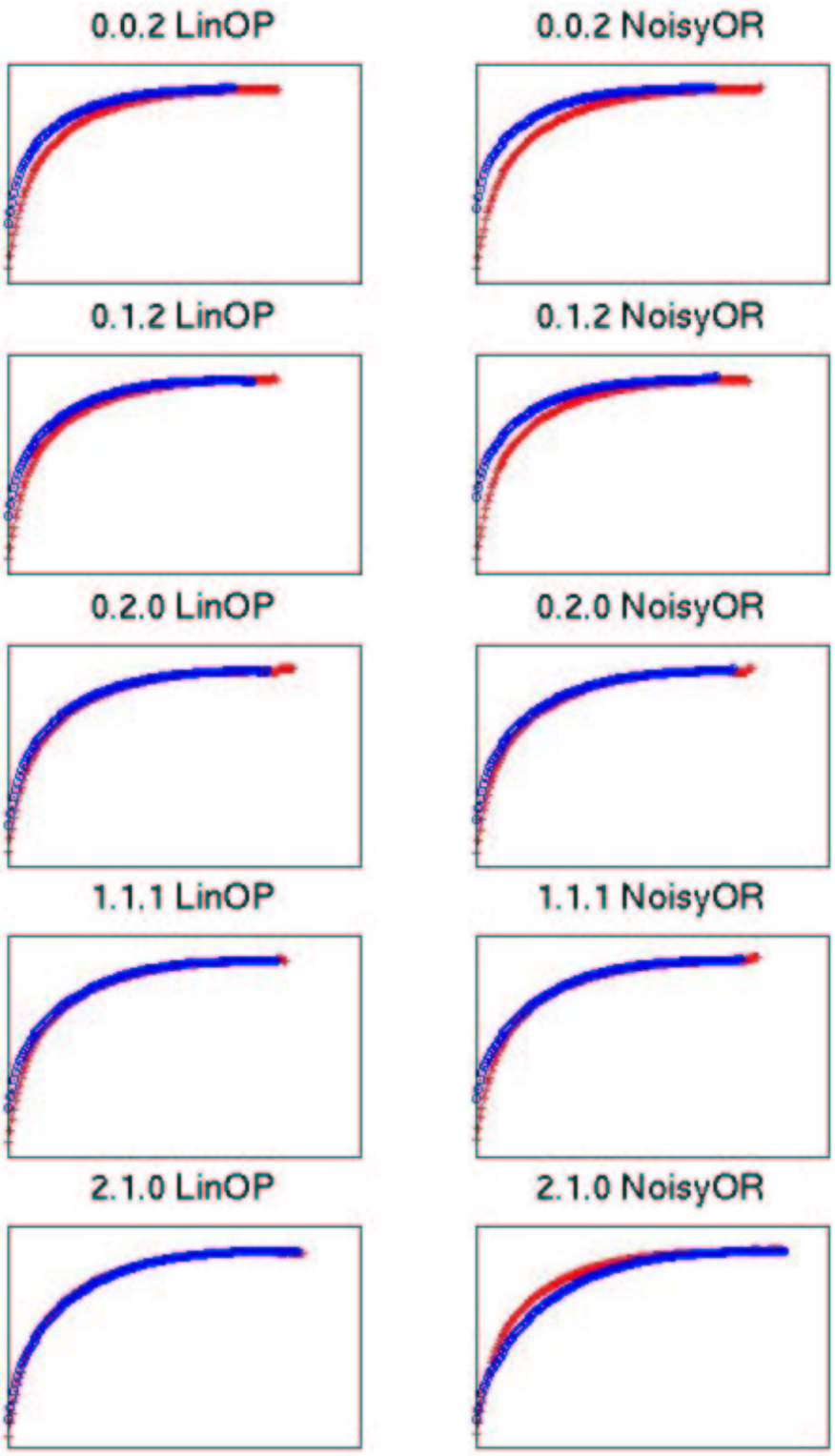


Figure 7.9: ALARM Criterion 2 (Average Log-Likelihood Per Iteration) for All Methods

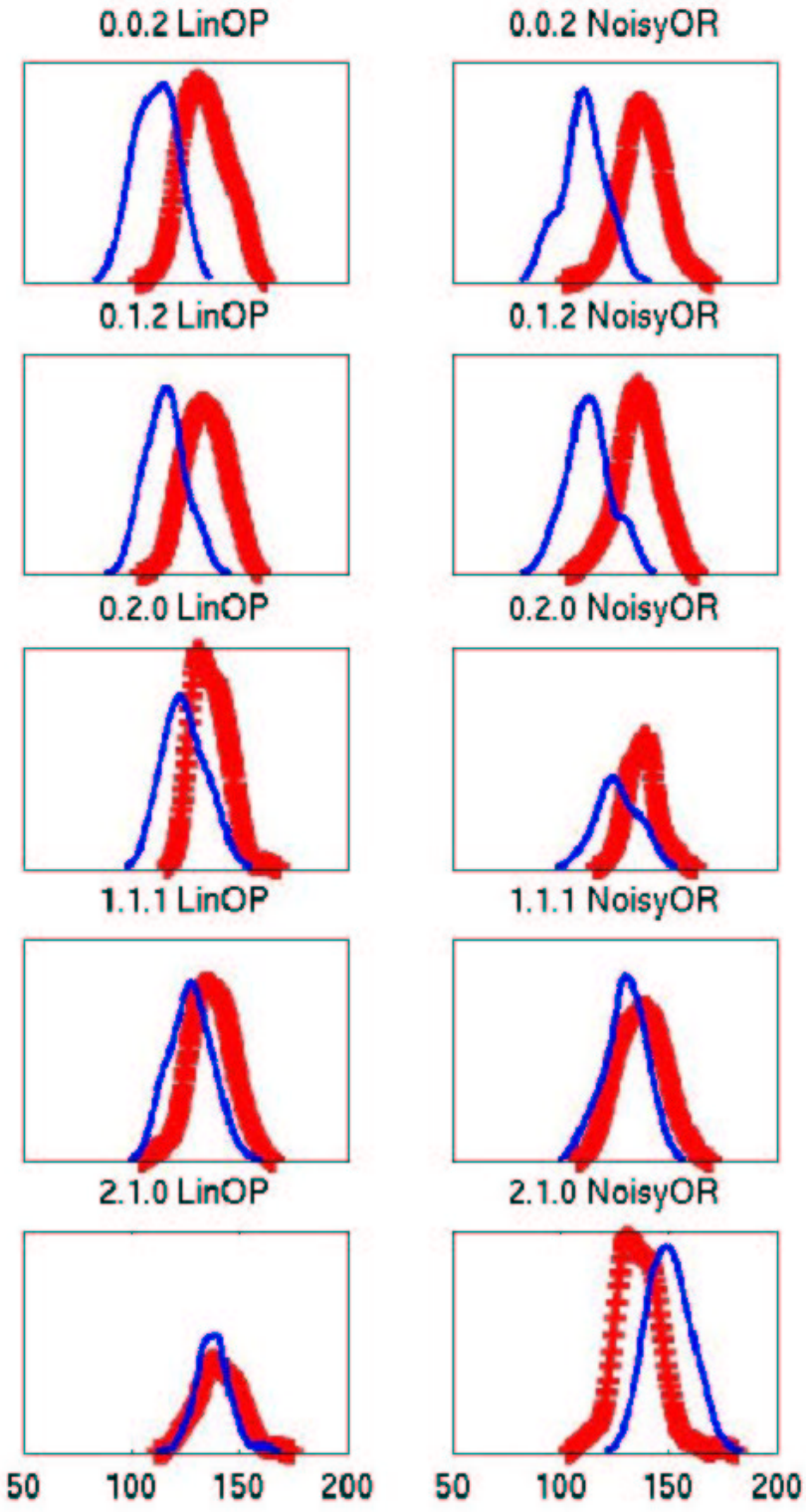


Figure 7.10: ALARM Criterion 3 (Distribution of Number of Search Iterations) for All Methods

7.2.2 ALARM Results Using Relative Evaluation Criteria

To assess comparative performance of the various informed priors, we use the relative evaluation criteria of Section 7.1.3. For ALARM, a different set of Random models was generated cocurrently with each of the 10 sets of Informed models for a total of 1000 Random models. Thus, the Random results for the relative criteria represent statistics on the full set of 1000.

The Boxplots of the log-likelihood of the starting models is shown in Figure 7.11. The Random model distribution consists of all 1000 models. We can see that the median of all Informed model distributions is greater than the median of the Random model distribution. From this plot, 0.1.2 NoisyOR appears to be the leading performer with respect to median yet the spread of model log-likelihood for the 0.0.2 NoisyOR distribution extends further into the higher log-likelihood range. The 2.1.0 versions (completely incorrect reliability assignments) have the weakest results. These observations reiterate those of Figure 7.7 which compared 2.1.0 LinOP to 2.1.0 NoisyOR, yet here it is made in the context of all sets of models.

The average log-likelihood for the first 10 iterations is shown in Figure 7.12. The average at each iteration for the Random curve, shown as a red x line, is taken over 1000 models. The LinOP versions are shown in blue while the NoisyOR versions are shown in green. Note that the spread of the curves for NoisyOR is wider than for LinOP, reflecting the former's susceptibility to incorrect reliability assignments. For the first few iterations, 0.1.2 NoisyOR appears to have higher average log-likelihood but is quickly overtaken by 0.0.2 NoisyOR.

From the figure of just the first few iterations, we see that both 2.1.0 versions begin to perform more poorly than the Random searches. This is a significant observation since earlier in Chapter 4, we made the comment that the emphasis on providing informative priors for Bayesian network learning has been on *parameter* priors since in typical applications, the likelihood term dominates the score. However, in a data-limited application such as the ones we explore here, we claim that the structural prior deserves greater attention. The results of Figure 7.12 speak directly to this claim, demonstrating that using an informed structural prior can have a significant impact on the search path taken. In particular, the badly informed prior created from the incorrect reliability assignment 2.1.0 causes the search to explore grossly suboptimal solutions, not only when compared to the other informed prior combinations, but even when compared to the uniform prior.

The significance of difference between log-likelihood distributions per iteration is shown in Figure 7.13. With the exception of the 1.1.1 and 2.1.0, the curves remain below the 0.05 significance threshold, shown as a red horizontal line, until roughly the 100th iteration. The 1.1.1 versions dip above the threshold twice, illustrating that the searches begin to perform poorly relative to the Random searches, then recover for most of the rest of the search, before returning to poor performance. The 2.1.0 versions almost immediately begin to perform worse than the Random searches, causing a rapid jump to values above the 0.05 threshold. This observation confirms the results of the previous criterion and is seen again in the next criterion.

The difference of average log-likelihood per iteration is shown in Figure 7.14. Again, the NoisyOR versions, in green, are spread more widely than the LinOP versions, in blue. From this graph, 0.0.2

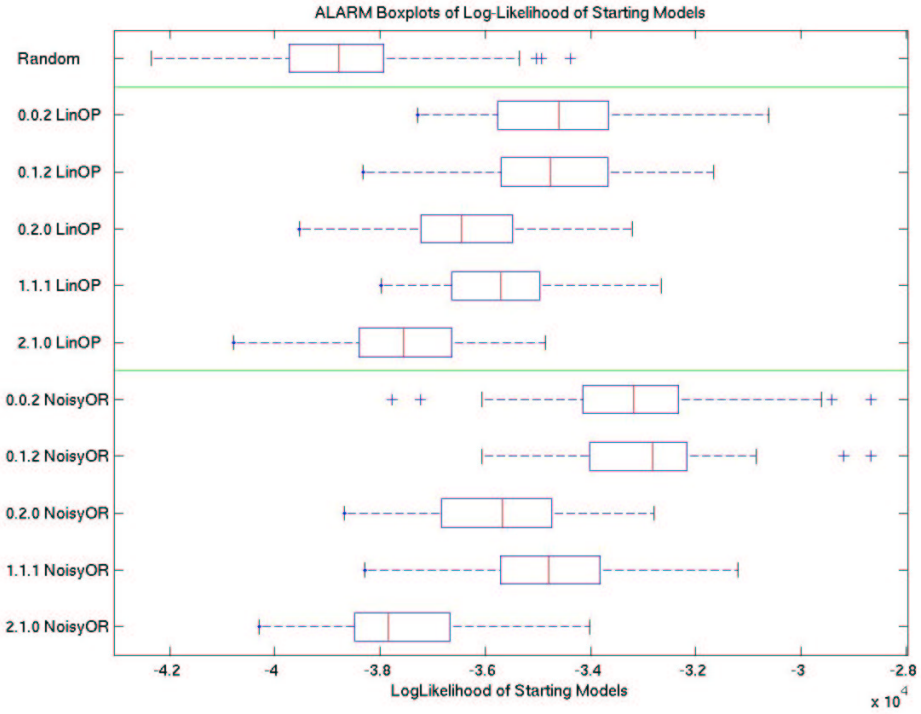


Figure 7.11: ALARM Boxplots of Log-Likelihood of Starting Models

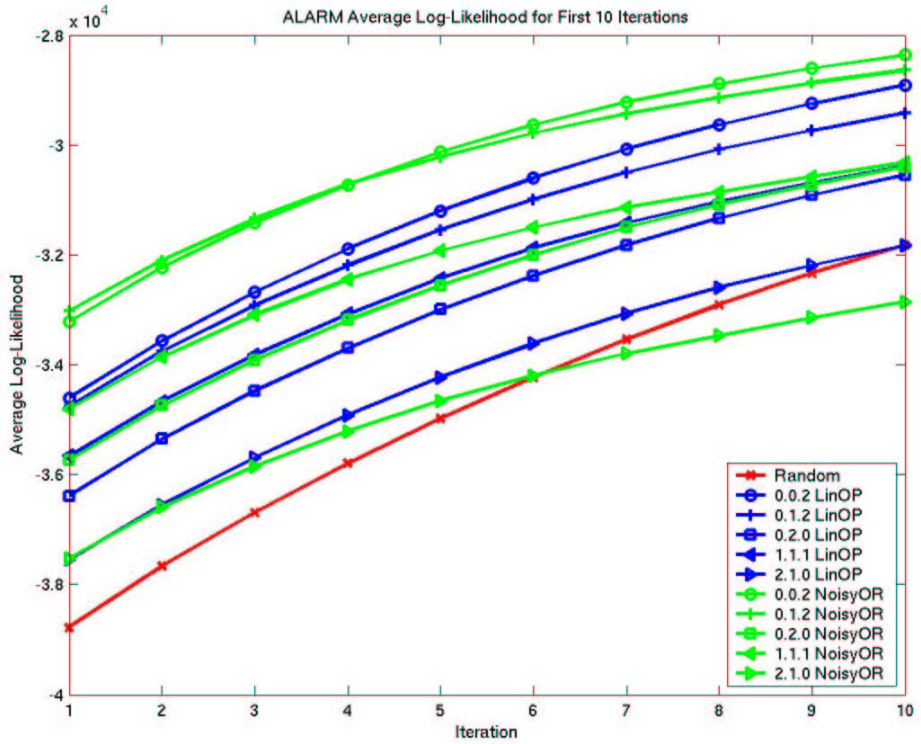


Figure 7.12: ALARM Average Log-Likelihood for First 10 Iterations

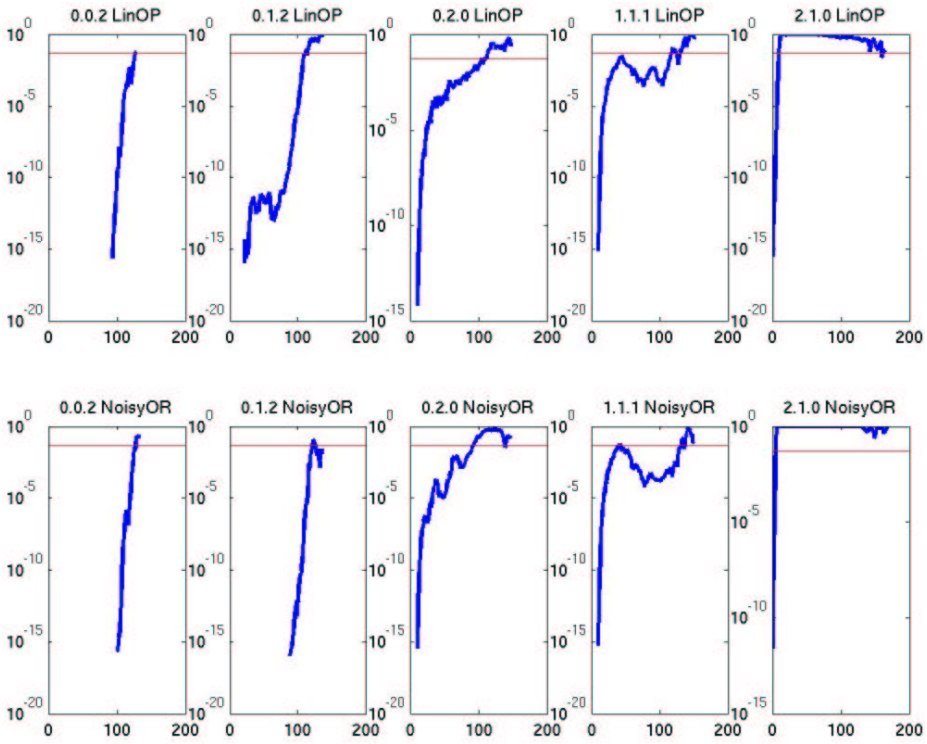


Figure 7.13: ALARM Significance of Difference between Log-Likelihood Distributions, per Iteration

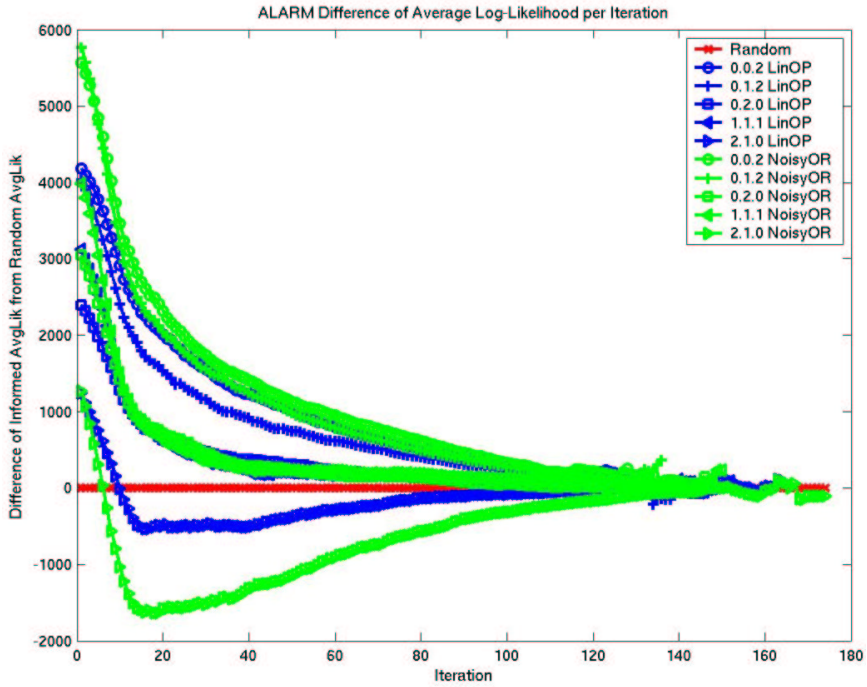


Figure 7.14: ALARM Difference in Average Log-Likelihood per Iteration

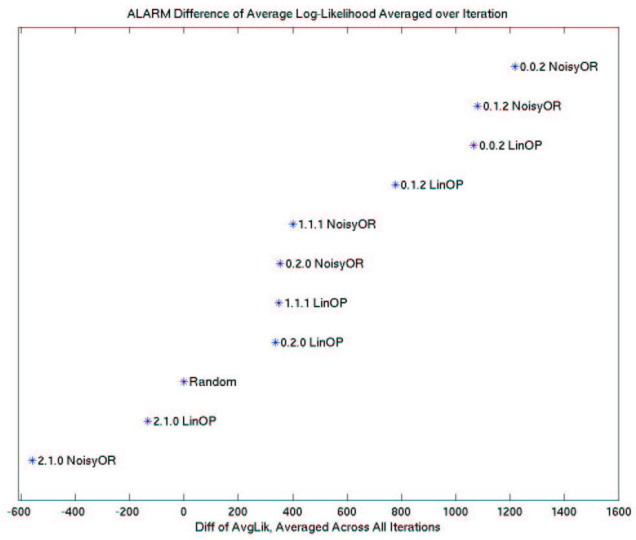


Figure 7.15: ALARM Difference in Average Log-Likelihood, Averaged over Iteration

NoisyOR (green circle) appears to perform the best over the complete set of iterations though 0.1.2 NoisyOR began as the best in Figure 7.12. This emphasizes the importance of viewing the performance of the algorithms during the entire search, not just the starting and ending models. Adding even greater weight to the point is the results for the 2.1.0 assignment. Both versions quickly drop below the Random curve (red x) and spend the rest of the search attempting to recover. As argued in a previous paragraph, this behavior is important in terms of the evaluation of the effect of informed structural priors in data-limited applications. Examining only the starting models, or as is typically done in comparative studies, only the ending models of the search, the use of informed structural priors may appear to have no effect on the final results. It is important to observe the behavior at the intermediate iterations in the search since some applications may require search to be limited to a fixed number of iterations rather than be allowed to continue until convergence. It is therefore useful to characterize the effects of informed structural priors on the search performance throughout.

The difference of average log-likelihood, averaged over all iterations is shown in Figure 7.15. In this figure, 0.0.2 NoisyOR emerges as the best performer while the 2.1.0 versions are the worst. Note that the assignments 1.1.1 and 0.2.0 perform roughly the same for both LinOP and NoisyOR. Also, on average, the NoisyOR versions outperform the LinOP versions, with the exception of the incorrect reliability assignment 2.1.0.

Overall, the results from the relative evaluation criteria reinforce the views provided by the individual evaluation criteria. The best combination is 0.0.2 NoisyOR, while 2.1.0 NoisyOR is the worst. Using the ALARM model allows us to explore the effect of errors in reliability assignments. The results allow us to conclude that while clear benefits are demonstrated with favorably informed structural priors, the use of incorrectly informed structural priors can have a significant effect on search performance.

7.3 Yeast Bayesian Network Learning

Data for the yeast genome is used to demonstrate the utility of including structural prior information in a real-world example. To make the dimensions of the problem comparable to those in the ALARM dataset, we learn Bayesian Networks on the 50 genes in the IDEKER50 set described in Section 6.2.3. Note however, that the consensus likelihood functions are learned using the full set of 331 genes (IDEKER331) and the relevant entries in those matrices are then simply extracted for the IDEKER50 genes for use as a structural prior.

In order to apply the BDeu metric within Bayesian network learning, the data must be over discrete variables and contain no missing values. In the following two sections, we describe how we obtain a gene expression dataset of 1738 samples and then impute the missing values. We then discuss the two strategies we used to discretized the real-valued expression values, namely STD and MAG1–IDRR.

For the learning algorithm applied to the resulting dataset of complete samples on discrete variables, two different settings are used for the equivalent sample size: $N' = 1$ to indicate a weak bias in the parameter prior, and $N' = 10$ to indicate a reasonably strong bias. Bayesian learning is applied to the combination of discrete dataset and BDeu parameter setting, using 110 random restarts of greedy hill-climbing for the STD $N' = 1$ combination and 50 random restarts for all other combinations. Each restart model is sampled from either an informed structural prior distribution using one of our consensus likelihood functions (24 in all) or a uniform structural prior. We refer to the former set of models as the Informed models and the latter as the Random models. Unlike ALARM where a set of Random models is generated cocurrently with each set of Informed models, the same set of Random models is used in all comparisons in yeast.

During learning, we set the max_fan-in=3 to control the number of parents of a given variable and we restrict the number of edges per graph to between 50 and 100, where a graph is generated by first selecting a target number of edges uniformly from that range and then including that number of edges by repeatedly sampling from the appropriate structural prior distribution.

We conclude the chapter by presenting the results of learning using the individual and relative evaluation criteria in Section 7.3.2 and Section 7.3.3 for all 24 consensus likelihood functions and all four combinations of discretization technique and equivalent sample size setting. A summary of the individual criterion is shown in panels simultaneously for all methods; the complete set of results organized instead by individual consensus function are included in Appendix F.

7.3.1 Gene Expression Dataset for Learning in Yeast

For the dataset used to learn the Bayesian networks, we obtain a collection of publically available data from Huang *et al.* [HMA02], Yvert *et al.* [YBW+03] as well as from the references used by Tanay *et al.* [SKS05]. [2] Each of the datasets from these resources represents an individual study of yeast gene expression, thus the measurements are taken over a large variety of experimental conditions

[2] Tanay *et al.* [SKS05] provide a list of the 52 references at `http://www.cs.tau.ac.il/ amos/qubic/database.html`.

and stimuli, using different technologies. To make the data more comparable across studies, we transform the data from cDNA microarrays to $\log_2$-ratios if not already done so by the original data source. Note that in cases where the data was previously log-transformed, the base may be either $\log_2$ or $\log_{10}$ since it is not always apparent which was used by the authors of the original study. We transform data from oligonucleotide microarrays to $\log_2$-ratios using a reference sample contained within the same dataset in the denominator of the ratio, or the average across all samples within the same dataset when no reference sample seems appropriate. Duplicate measurements for the same ORF in a single sample are averaged. Overall, the combined dataset has gene expression data for 7102 yeast ORFs in 1738 samples, where individual experimental studies contribute anywhere from 1 to 300 samples.

Imputation of Missing Values in the Yeast Dataset

In order to apply the BDeu metric during learning, the dataset must be complete. For our dataset of 7102 ORFs by 1738 samples, 15% of the data is missing. Typical strategies for imputation include setting the missing values to zero (or some target value) or replacing the missing values with the mean across all samples. In most applications where the data is missing at random, these imputation strategies are sufficient. However, in our application, doing so can introduce correlations between genes since data is often missing due to the fact that a whole subset of genes may not be included on a particular version of the microarray. This means that the same set of genes will have missing data for the same set of samples and setting all missing values to a target value results in superficial correlation.

To overcome this problem, we have developed a strategy for imputation which is sensitive to the patterns of missing values evident in gene expression data. We first remove the ORFs that have more than 50% missing data across samples. This prevents any analysis of genes whose data would otherwise have a majority of imputed values. Of the remaining 6161 ORFs, 99% of them have less than 15% missing data. Imputation is done by taking the average expression value from the top five nearest neighbors. In gene expression data, we must be careful in defining what it means to be a nearest neighbor.

Computing the (Euclidean) distance to a neighbor is sensitive to the number of missing values for both the gene and its potential neighbor. Euclidean distance between two vectors with missing values sums the difference in values over each commonly non-missing dimension of the vectors. Suppose we have $M = 500$ samples, then the distance of a target gene G to a neighbor with 498 non-missing values will necessarily be greater than the distance of G to a gene with only 301 non-missing values. Carrying this further, the closest neighbor to G would therefore be the neighbor with the greatest number of missing values where G has values.

To avoid this situation, we could require that nearest neighbors be calculated only among genes with the same, or greater, number of non-missing values as the target gene. However, for genes with only a few missing values, we must ensure that this policy does not also result in arbitrarily imputed values. For example, suppose $M = 500$ and G has 498 valid columns but there are only six genes with

more than 497 valid columns. We wish to compute averages over five nearest neighbors yet if our set from which to choose neighbors is restricted to those six genes, the imputed values will be almost arbitrary; we are unlikely to find a true close neighbor from among those six. We instead introduce a parameter ATLEAST which specifies the minimum number of non-missing dimensions that a gene must have in common to be considered as a potential neighbor of another gene. This value is set to the minimum of $2/3 * M$ and the number of valid datapoints in the target gene. However, we can again arrive at the situation where a neighbor seems to be the closest merely because it barely meets the minimum requirement for the number of valid datapoints.

Our solution is to compute the distance using only ATLEAST number of columns, by sorting the difference between the vector values for each dimension and summing over only the top ATLEAST maximum differences. Since the value of ATLEAST can depend on the number of valid datapoints for a given gene, if that gene has $m <$ ATLEAST valid datapoints in the first place, all distance computations are therefore computed using the maximum m common columns, which in this case is exactly the number of common columns the gene could have anyway.

Using the strategy described above, we impute values for the dataset of 6161 genes which now has only 3.9% missing values (versus 15% for the 7102 genes). As an aside, the expression data for the set of 50 genes in IDEKER50 has only 2.5% missing data. In the next section, we describe how the completed dataset is discretized.

Discretizing the Yeast Dataset

In Appendix A, we evaluate 31 different strategies for discretization on a variety of types of datasets, including gene expression data. In fact, we present three classes of discretization methods that are motivated by the types of analyses required for gene expression datasets. However, that evaluation is made on individual datasets generated for a single experimental study, where each sample is measured under laboratory conditions similar to the other samples in the same study. The dataset of 6161 ORFs and 1738 samples we wish to use for Bayesian network learning is instead comprised of multiple experiments from different labs, on different microarray platforms. Careful attention must be paid to normalizing the data before discretization such that values in samples from one experiment are comparable to values in samples from other experiments. Our use of the log transformation described earlier also speaks to this objective.

Based on the results reported in Appendix A, we choose two strategies for discretization. For ease of explanation, consider the dataset as a matrix of 6161 rows and 1738 columns. The first discretization technique we use, denoted STD, is applied to each column (sample), where we compute the mean and standard deviation of expression values in each column. The values in the column are then assigned to one of three bins where the bin boundaries are drawn at the mean plus or minus one standard deviation. Thus, values that fall within one standard deviation of the mean correspond to expression changes that do not deviate far from the mean expression level in the individual sample. Values outside one standard deviation can be considered to represent highly over- or under-expressed genes relative to that sample. In this way, the discrete values assigned to

genes in different samples have a similar interpretation of meaning. Note that STD is more accurate in terms of determining the meaning of over- or under-expression if as many genes as possible are used in computing the mean and standard deviation. Even though the STD method did not show strong performance in the evaluations of Appendix A, the use of standard deviation thresholds is familiar to biologists when interpreting differential expression of a gene. Moreover, the evaluations in Appendix A considered STD applied to each row so that values could be interpreted as expression changes within an experimental study, rather than as used here, for normalization across studies.

The second discretization technique, denoted MAG1–IDRR, uses one of the novel techniques presented in Appendix A that was shown to have superior performance across most evaluation metrics we considered. The method is called *Individual correlation of Discrete to Real (IDR)*, and here we use the particular variant that employs random restarts, thus named IDRR. The IDR method is applied to each gene such that bin boundaries are chosen to maximize the Pearson correlation between the vector of real-valued expression levels in all samples and the resulting vector of discrete values given the set of bin boundaries. Intuitively, IDR attempts to preserve the shape of the vector profile as much as possible when discretizing. Since the search over bin-boundaries is exponential in the number of bins, we use a greedy local search which begins with an initial assignment of boundaries and explores small changes to that assignment in order to maximize the objective function. Though we show in Appendix A that the objective function is nearly monotonic, numerical imprecision can lead to local maxima. Thus, we use random restarts to avoid suboptimal solutions, thereby developing the discretization method we refer to as IDRR.

In order to apply IDRR to individual genes whose expression levels are a concatenation of datasets from different experiments, we must first normalize the data to make it comparable across samples. Similar to the discussion of STD, we would like a real-valued expression level of 0 to have the same interpretation in each sample. Thus, we normalize the data in each column of the datamatrix to have mean 0 and magnitude 1, where the magnitude $|v|$ of a vector v with n dimensions is defined as $|v| = \sqrt{\sum_{i=1}^{n} v_i}$.

Applying this normalization, referred to as MAG1, the (log) expression values in each column generally fall between -1 and 1 so that genes with values near 1 for a given sample indicate that those genes were highly overexpressed in the particular experimental condition interrogated by the sample. After applying the MAG1 normalization to the datamatrix columns, IDRR using three bins is applied to the rows. Therefore, supposing the bin identities are -1, 0, and 1 respectively, a gene assigned discrete values of 1 across different samples indicates that the gene was highly expressed in each of those samples. After the initial MAG1 normalization step, using IDRR discretization is independent for each gene, thus we need only discretize the genes of the IDEKER50 dataset.

In the following two sections, we show the results of applying Bayesian network learning to each of the two discretized datasets, STD and MAG1–IDRR. We report the performance using the individual evaluation criteria described in Section 7.1.2 and the relative evaluation criteria described in Section 7.1.3.

7.3.2 Yeast Results Using Individual Evaluation Criteria

Figure 7.16 through Figure 7.19 show the results for Criterion C1 where the individual components of the score are plotted along each axis for the starting and ending models using informed or uniform priors. In each figure, there are examples where the Informed starting models have higher scores than the corresponding Random models, notably the STRING and MAGIC results. There are also examples where the scores for the starting Informed models show little or no improvement over the Random models. As with the ALARM results, the **RAND1** versions consistently show poor scores relative to the Random models. Also the Informed models in the **No Ratio** versions of BAYES and PRM, shown in the third column of graphs in each figure, do not appear to perform better than the Random models. In general, the **No Ratio** versions for BAYES and PRM perform much worse than their **Ratio** counterparts. The most dramatic example of this is Figure 7.19 for the PRM results. The Informed starting scores in the **No Ratio** versions is heavily concentrated in a very narrow range of log-likelihood values along the horizontal axis, barely distinguishable from the Random scores. In the **Ratio** versions, the Informed PRM starting scores are clearly separated from the Random scores. As in ALARM, the scores for the final models in all Informed searches converge to the same general values as the Random searches.

Figure 7.20 through Figure 7.23 show the results for Criterion 2, the average log-likelihood per iteration. Generally, the Informed searches perform better than the Random searches in terms of higher average log-likelihood per iteration. The exceptions, once again, are the **No Ratio** versions of the BAYES and PRM. The **Unif5, RAND1** and **RAND2** show slight improvements while the STRING, MAGIC, **CONS** and PRM **Ratio** versions most strongly exhibit differences from Random.

Figure 7.24 through Figure 7.27 show the results for Criterion 3, the distribution of number of iterations per type of prior. Nearly all Informed priors yield distributions whose average is distinctly lower than Random. The PRM **No Ratio** versions in Figure 7.27 show the poorest performance relative to Random.

For Criterion 1 on both STD and MAG1–IDRR, using $N' = 10$ appears to result in more spherical clusters of scores than when using $N' = 10$. For Criterion 2, visual inspection suggests there is slightly less distance between the Informed curves than the Random curves for $N' = 10$ versus $N' = 1$ for STD, while the opposite is true in the MAG1–IDRR results. For Criterion 3, the use of $N' = 10$ often results in marginally more peaked distributions for the Informed curves relative to their $N' = 1$ counterparts. However, despite these small differences for all three criteria, there is little change in the qualitative results using STD versus MAG1–IDRR or $N' = 1$ versus $N' = 10$.

Overall, the set of individual criteria for Bayesian network learning in the yeast IDEKER50 gene set offer the same conclusions as for the ALARM dataset. In cases where expert reliability assignment is reasonably accurate, using informed structural priors formed from consensus likelihood functions results in better scoring starting models, better scoring models during search and shorter searches when compared to uninformed structural priors. These results are very encouraging since they hold true in the yeast domain despite having significantly more complex experts than in the ALARM example.

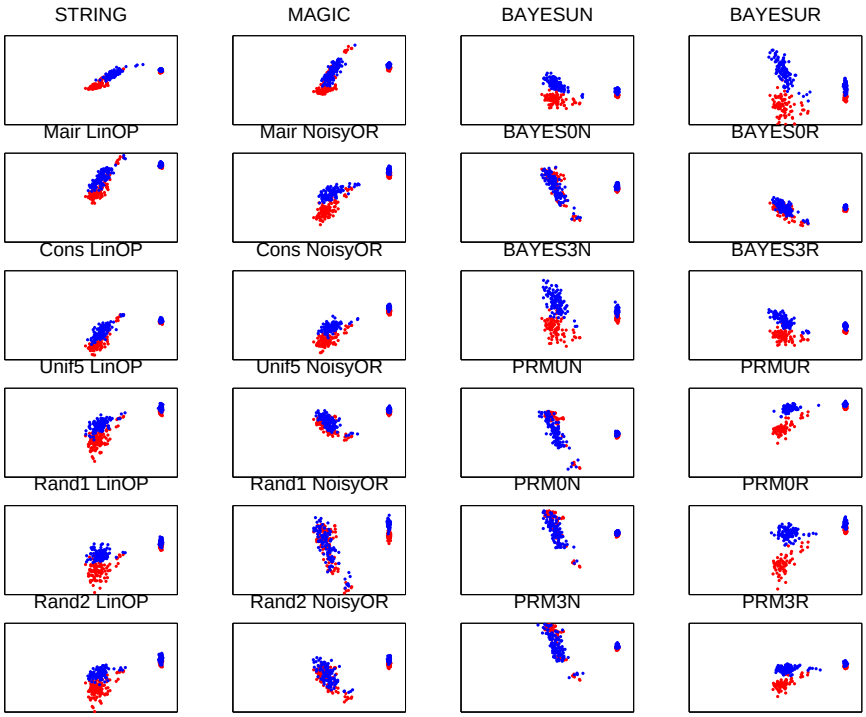


Figure 7.16: IDEKER50 STD $N' = 1$ Criterion1: Score Components

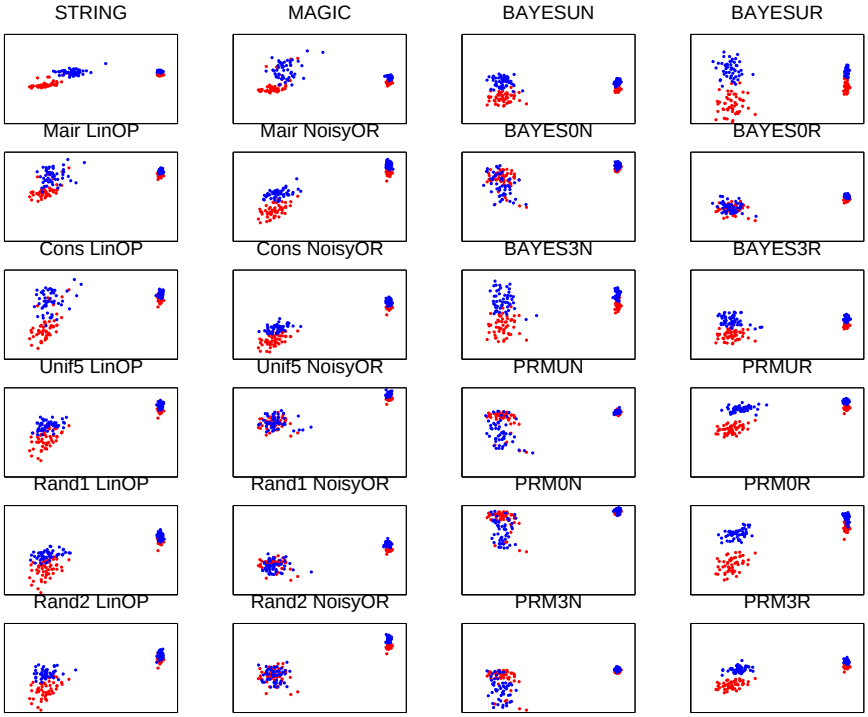


Figure 7.17: IDEKER50 STD $N' = 10$ Criterion1: Score Components

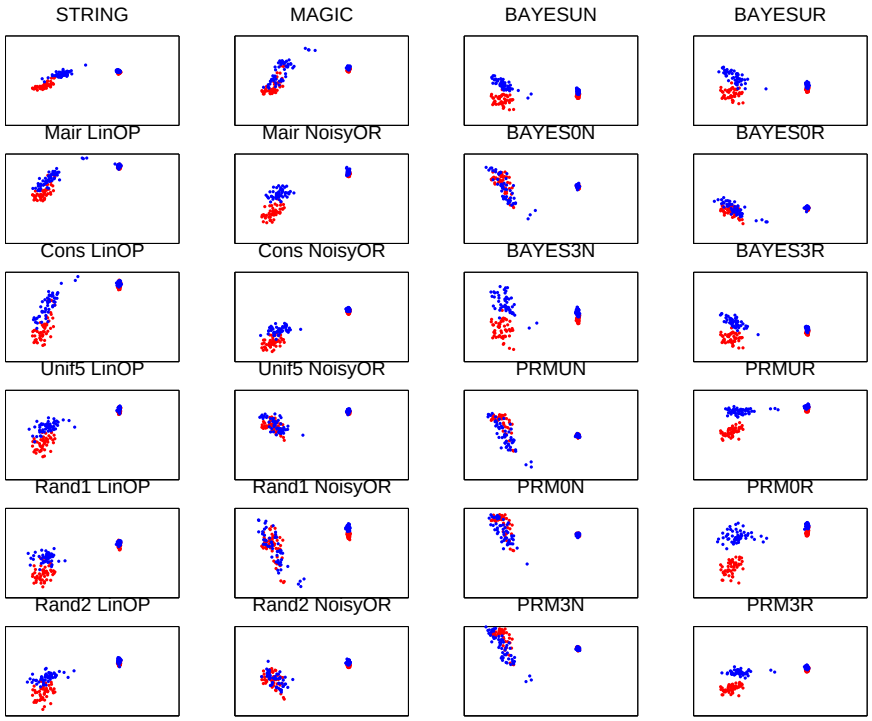


Figure 7.18: IDEKER50 MAG1–IDRR $N' = 1$ Criterion1: Score Components

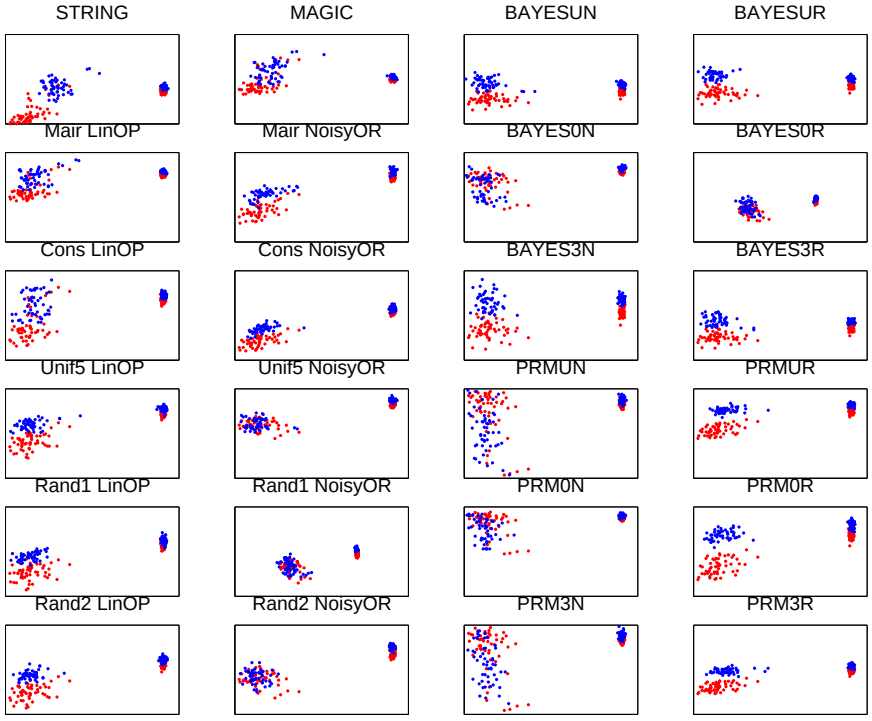


Figure 7.19: IDEKER50 MAG1–IDRR $N' = 10$ Criterion1: Score Components

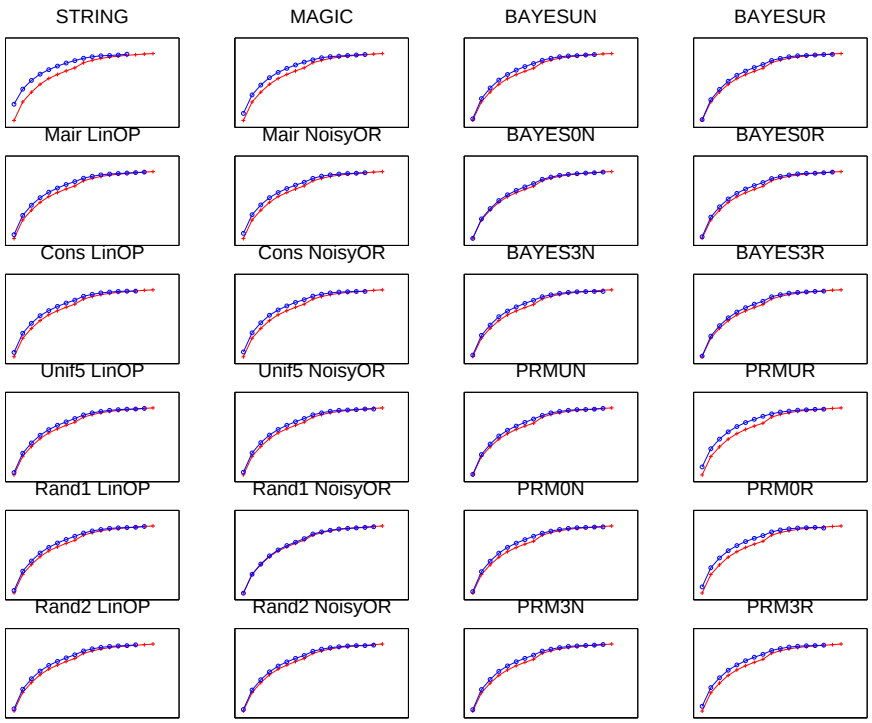


Figure 7.20: IDEKER50 STD $N' = 1$ Criterion2: Average Log-Likelihood per Iteration

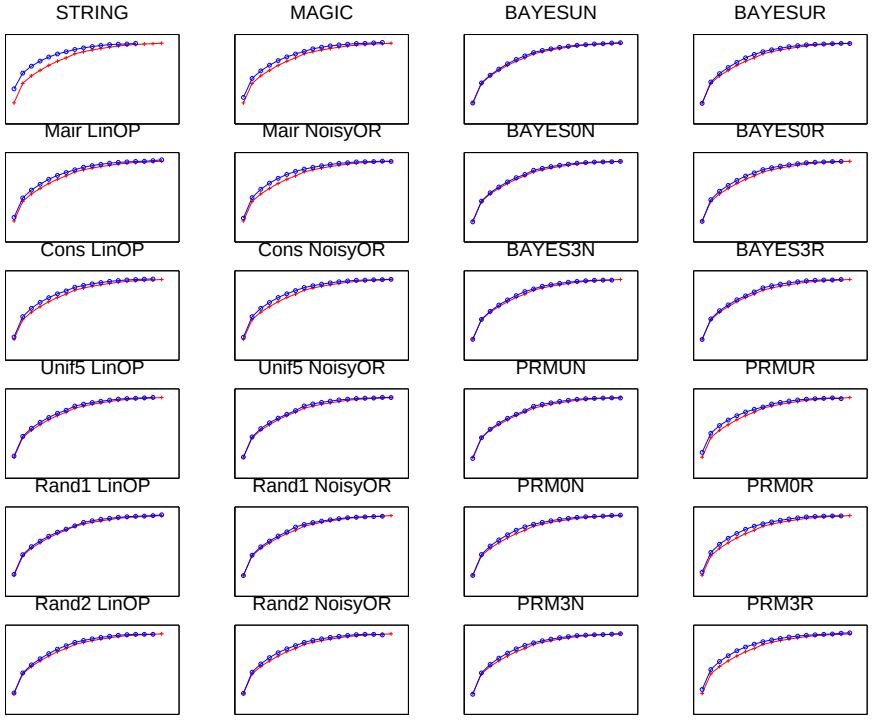


Figure 7.21: IDEKER50 STD $N' = 10$ Criterion2: Average Log-Likelihood per Iteration

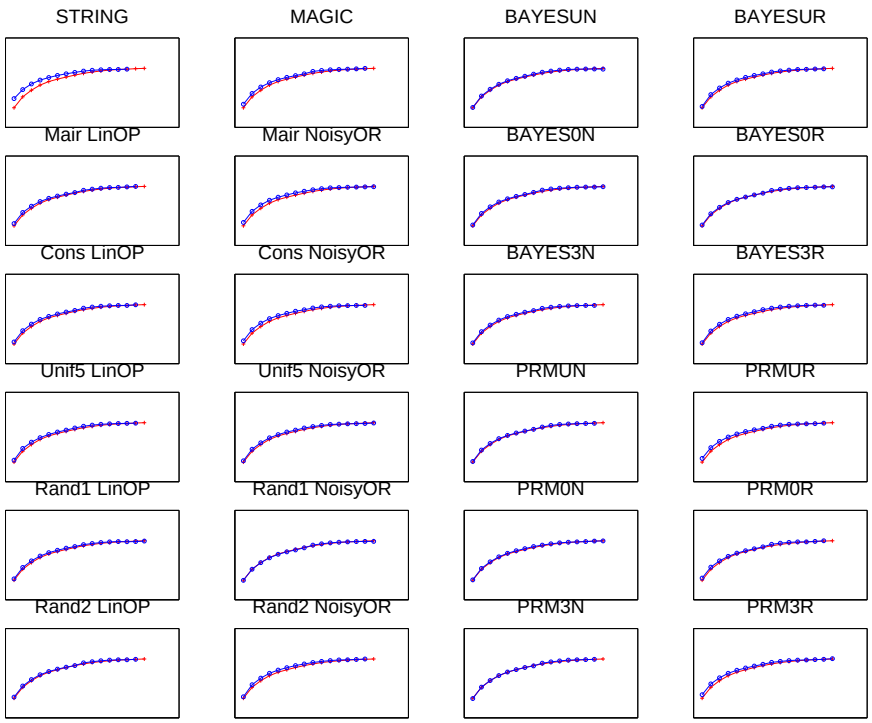


Figure 7.22: IDEKER50 MAG1–IDRR $N' = 1$ Criterion2: Average Log-Likelihood per Iteration

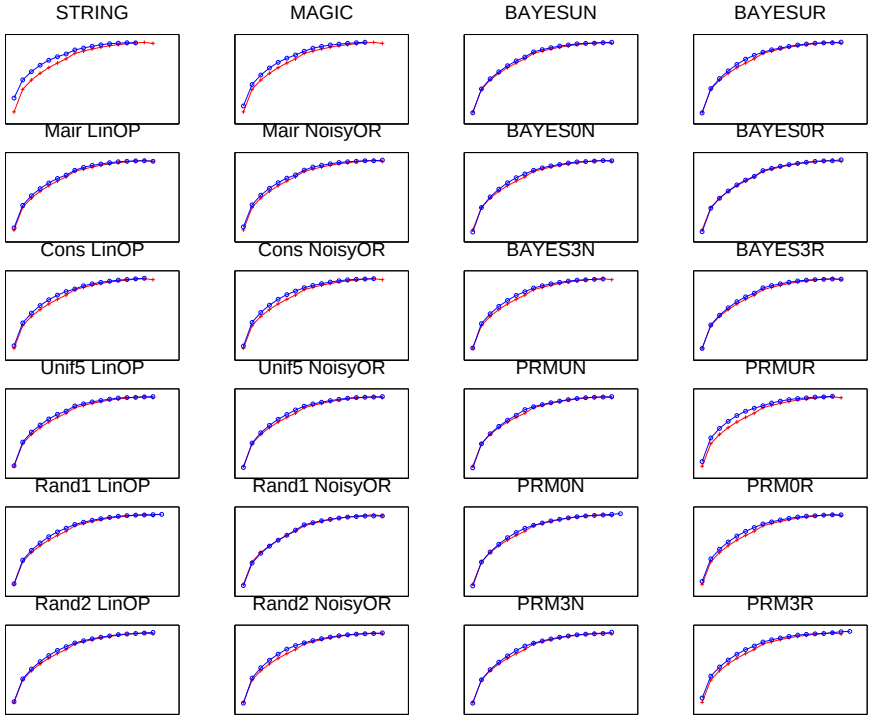


Figure 7.23: IDEKER50 MAG1–IDRR $N' = 10$ Criterion2: Average Log-Likelihood per Iteration

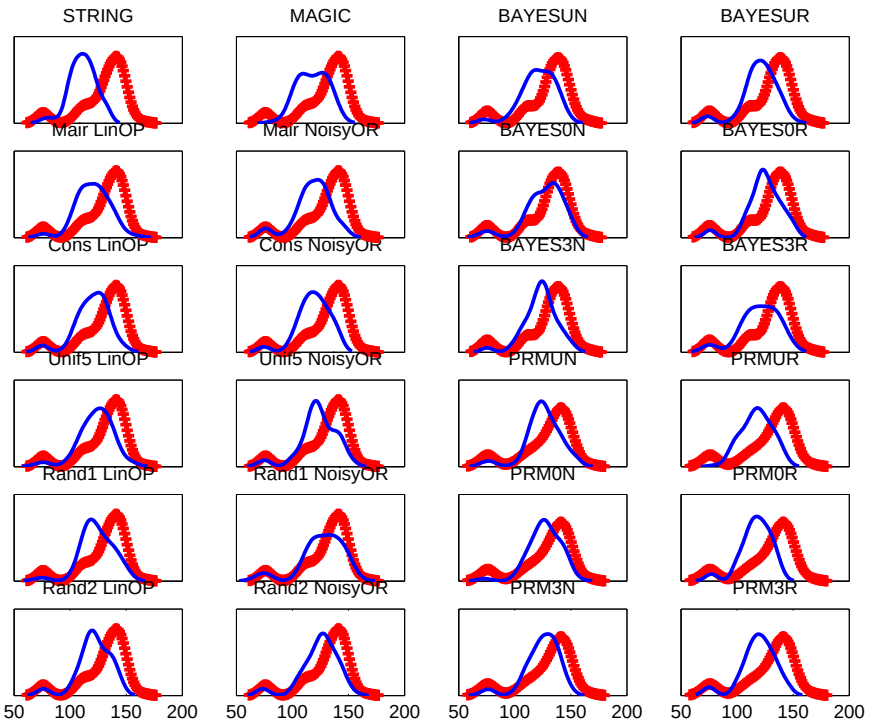


Figure 7.24: IDEKER50 STD $N' = 1$ Criterion3: Density Plot of Number of Iterations

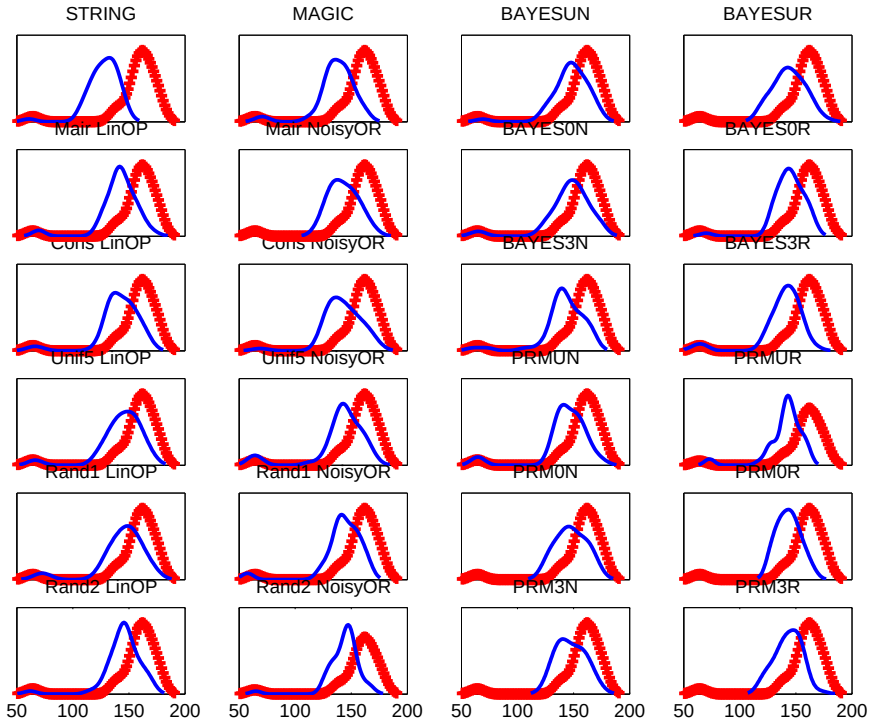


Figure 7.25: IDEKER50 STD $N' = 10$ Criterion3: Density Plot of Number of Iterations

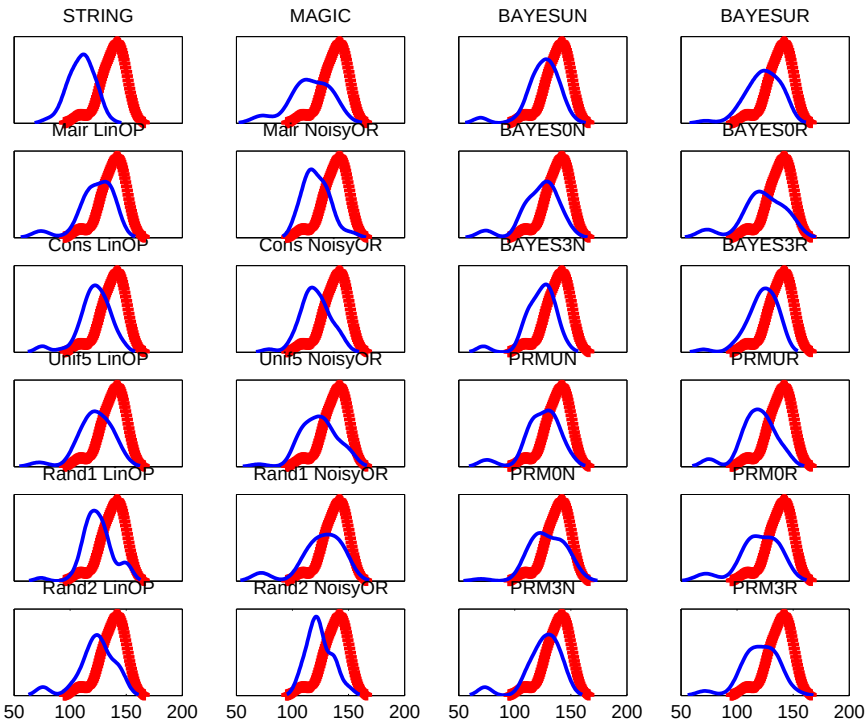


Figure 7.26: IDEKER50 MAG1–IDRR $N' = 1$ Criterion3: Density Plot of Number of Iterations

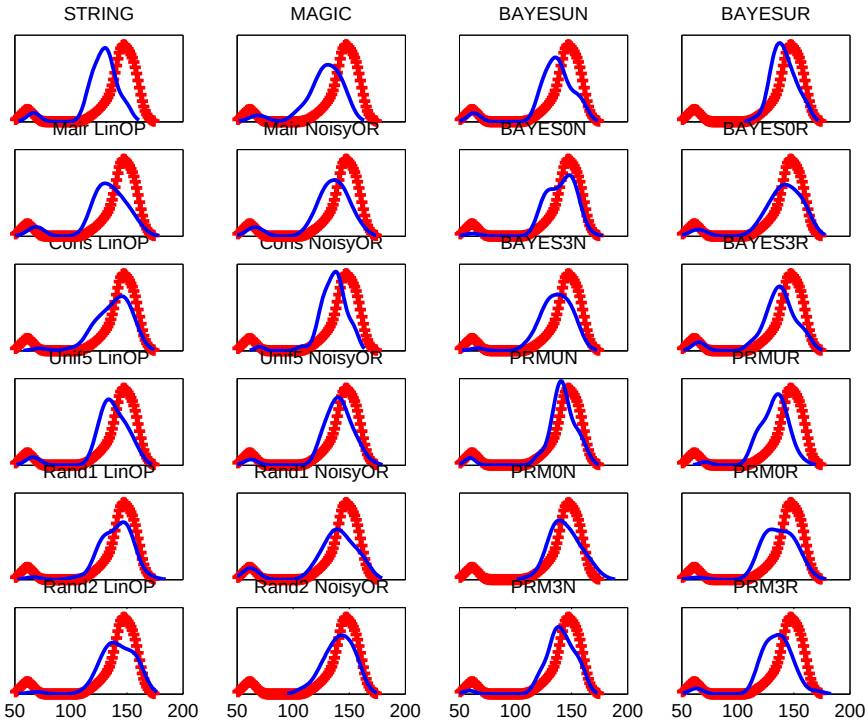


Figure 7.27: IDEKER50 MAG1–IDRR $N' = 10$ Criterion3: Density Plot of Number of Iterations

7.3.3 Yeast Results Using Relative Evaluation Criteria

Figure 7.28 shows Boxplots of the log-likelihood distribution for starting models. For ease of comparison, a vertical black line is placed at the median of the Random distribution. Those Informed boxes where the red median line lies to the left of the black vertical line indicate poor performance relative to Random. For STD $N' = 1$, all Informed priors generate distributions whose median is greater than Random, except **RAND1** NoisyOR whose median lies almost directly at the black vertical line. In contrast, STD with $N' = 10$ results in a majority of **Informed** distributions performing worse than the Random distribution. For the MAG1–IDRR comparisons, there is also an increase in the number of poorly performing Informed distributions using $N' = 10$ relative to $N' = 1$, though there are fewer high performers for MAG1–IDRR than for STD with $N' = 1$. In both $N' = 10$ plots, all BAYES variants show poor results relative to the other Informed distributions. In general, the **No Ratio** versions of BAYES and PRM exhibit lower medians than their **Ratio** counterparts. The STRING, MAGIC, **MAIR**, **CONS** and PRM **Ratio** priors show consistently higher performance than Random under this criterion.

Figure 7.29 and Figure 7.30 show the average log-likelihood for the first 10 iterations for STD and MAG1–IDRR, respectively. In all plots, several methods show poor performance relative to Random since their curves fall lower than the red curve. As with the ALARM results, the spread of the curves for NoisyOR versus LinOP is greater. The STRING, MAGIC and several PRM variants are consistently the best performers according to this criterion. There appears to be a larger separation of STRING from the next best performer for MAG1–IDRR when $N' = 1$ than when $N' = 10$. However the gap between STRING and Random is closer at $N' = 10$ than $N' = 1$, reinforcing our earlier observations about the effect of varying N' on Criterion 2.

Figure 7.31 and Figure 7.32 show the significance of the difference between the log-likelihood distributions of Informed and Random per iteration. Recall that the red horizontal line indicates a significance threshold of 0.05. Values above this line mean poor performance of Informed relative to Random. For STD $N' = 1$, most methods have log-likelihood distributions consistently below that of Random, with the exception of **RAND1** NoisyOR and BAYES0N. At $N' = 10$ for STD, the **UNIF5, RAND1, RAND2**, BAYES and PRM versions show poor performance relative to Random. For MAG1–IDRR, the **No Ratio** versions of BAYES and PRM often stray above the red significance line, with severe effects for $N' = 10$. In that same plot, **UNIF5, RAND1**, and **RAND2** also show degraded performance with the curve above the red line for the majority of iterations.

Figure 7.33 and Figure 7.34 show the difference of average log-likelihood per iteration for STD and MAG1–IDRR respectively. The same oscillatory behavior seen in the significance plots is seen for these curves. Most methods show an increase in the difference up to iteration 12, where the difference drops and rises again dramatically before continuing with a steady decline. STRING, MAGIC and PRMUR consistently perform well in these plots, regardless of discretization strategy or N' setting.

Figure 7.35 shows the difference in average log-likelihood as displayed in Figure 7.33 and Figure 7.34, averaged over all iterations. This provides a summary of the performance of each method throughout the search. Evident in each of the plots is the dramatic performance of STRING relative to the other methods. Other high performers are MAGIC, **CONS** NoisyOR and several of the PRM **Ratio** variants. For the MAG1–IDRR dataset, **RAND1** NoisyOR actually performs worse than average according to this criterion. Generally the **No Ratio** and all BAYES variants show poor performance.

7.3.4 Conclusions from Yeast Results

The results using either STD or MAG1–IDRR for discretization are quite similar. Differences are usually observed between individual methods but the overall conclusions remain the same. Generally more of the informed structural priors performed better than uniformed with the STD dataset than with MAG1–IDRR. The MAG1–IDRR appears to be slightly more sensitive to the setting of N' than STD. As N' is increased for the MAG1-IDRR dataset, the margins of relative performance between the top Informed performers and Rand results becomes narrower than seen using the STD dataset.

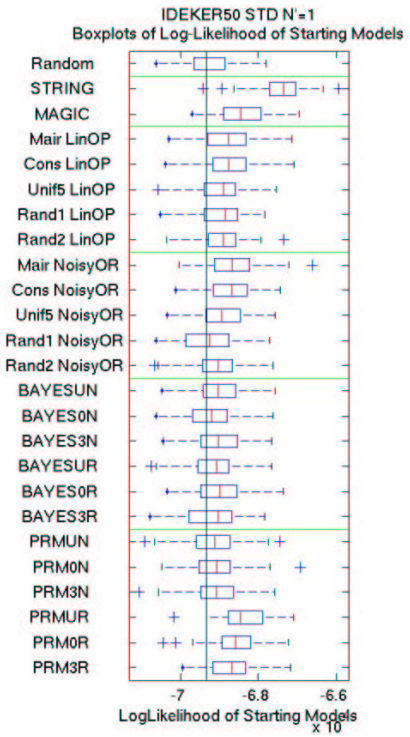
Regardless of discretization technique or equivalent sample size, the strongest performance in all individual and relative evaluation metrics is demonstrated by the STRING matrix. In fact, there was a large differential between the results for STRING and the next best method under all criteria. Earlier in the Introduction, we raised the question of whether the types on information we have available for forming structural priors would indeed be helpful for learning from expression data since the assertions of interactions are made on very different molecule types and are often indirect. These results suggest that the highly evolutionarily-conserved interactions uncovered by STRING provide very effective prior information for learning from expression data.

The MAGIC matrix also demonstrated considerable merit under all evaluation metrics. Recall that, like STRING, the consensus likelihood matrix for MAGIC contains fewer high-probability edges (red areas in the matrix) compared to the other matrices, which argues on the side of conservatism when creating consensus likelihood matrices. This fact is also shown by the observed difference in performance between the **SEMRELG0** and **SEMRELG3** variants of BAYES and PRM, where generally the more conservative assignment policy of **SEMRELG3** which required at least three experts before setting the SEMREL or SEMREL.Exists variable yields better results for Bayesian network learning. Also, the PRM variants generally performed better than the BAYES variants, arguing the case that learning interactions simultaneously for all pairs of genes has an advantage over considering information independently for each pair.

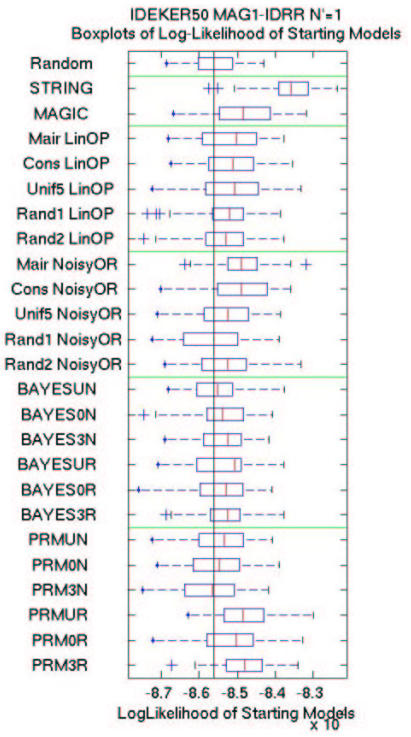
The **MAIR** and **CONS** reliability assignments, especially in conjunction with the NoisyOR consensus likelihood function, also show favorable results and were often close to one another in performance. As discussed earlier, this suggests that **CONS** may be a viable automated alternative to assigning reliabilities. Generally the **UNIF5** reliability assignment did not have strong results which provides evidence to the claim that it is important to take into account the relative reliability

of the expert sources which comprise the consensus likelihood function.

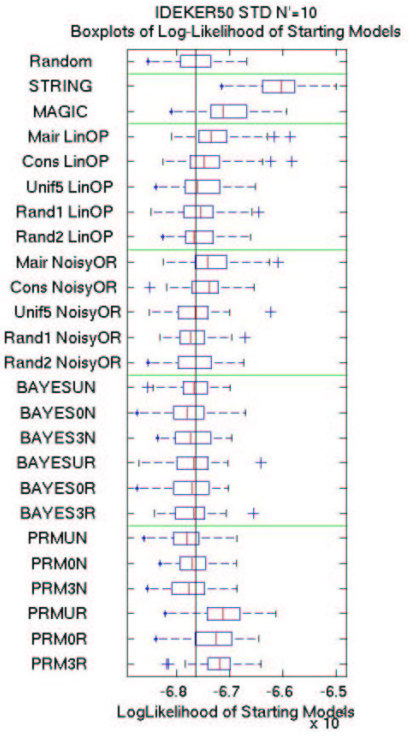
Overall, we have demonstrated in this chapter that the use of informed structural priors can greatly improve the Bayesian network learning by speeding the search to find high quality models, compared to learning from uninformed priors. We showed that structural priors can have a profound effect on search in data-limited applications by exploring the consequences of errors in judgements of reliabilities for our experts. This result is an important contribution to the field of Bayesian network learning since research up to this point has concentrated primarily on formulating parameter priors. Our research represents an important first step in the exploration of conditions under which we might elevate the importance of structural priors.



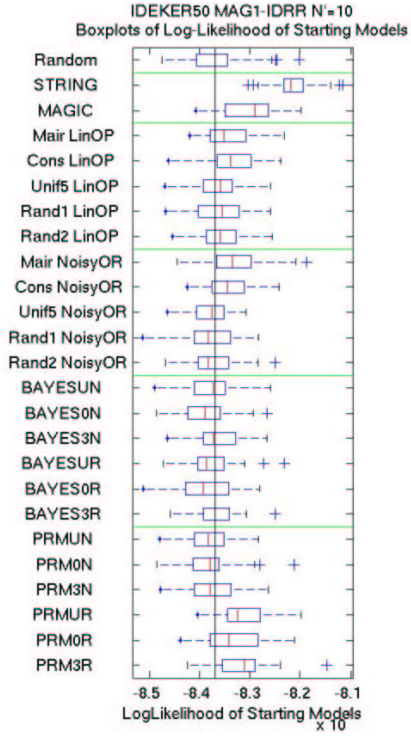
STD $N' = 1$



MAG1–IDRR $N' = 1$

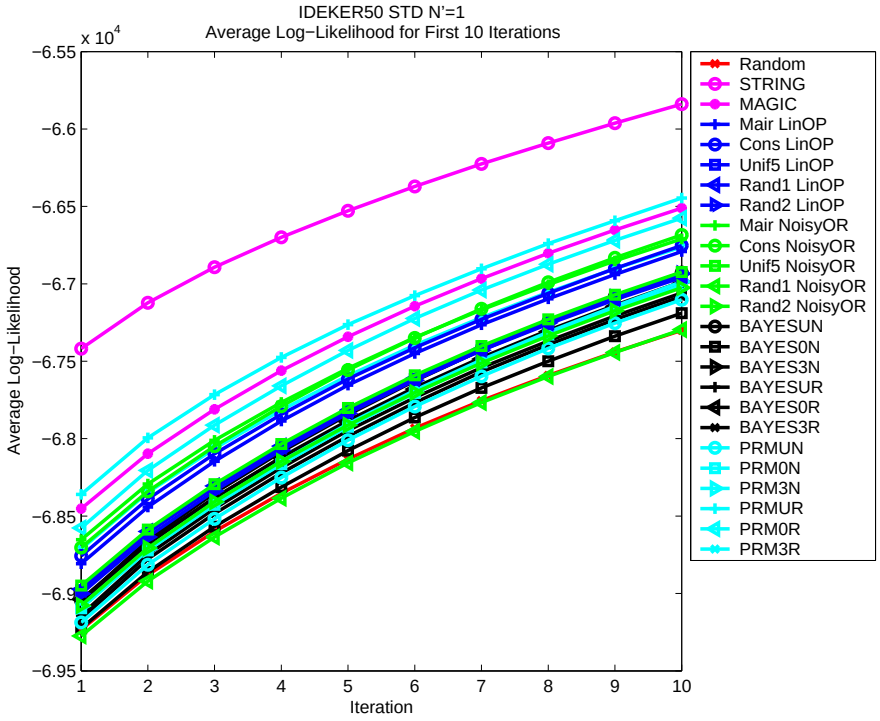


STD $N' = 10$

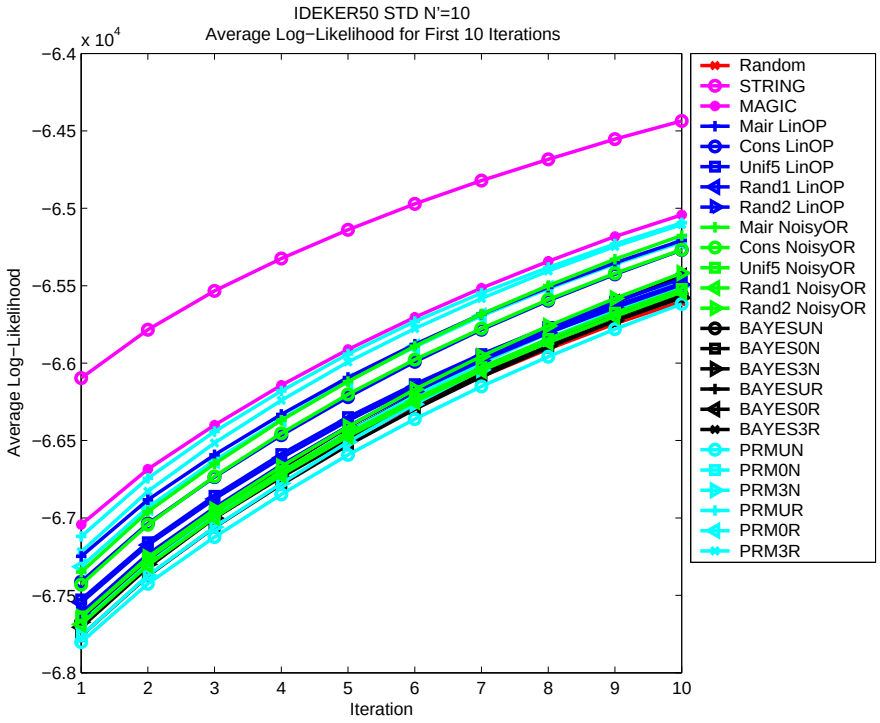


MAG1–IDRR $N' = 10$

Figure 7.28: IDEKER50 Boxplots of Log-Likelihood of Starting Models

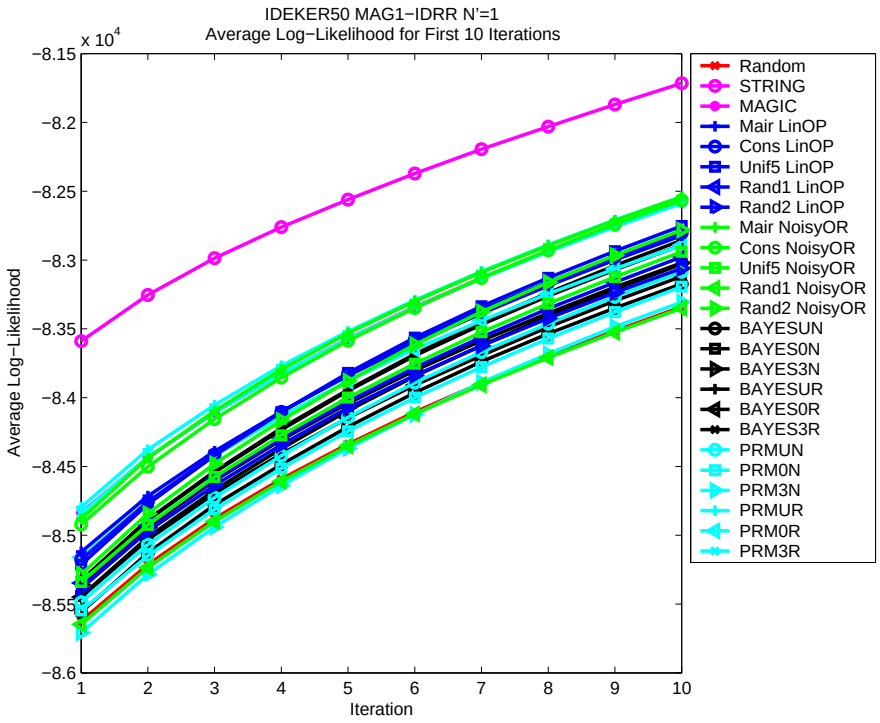


STD $N' = 1$

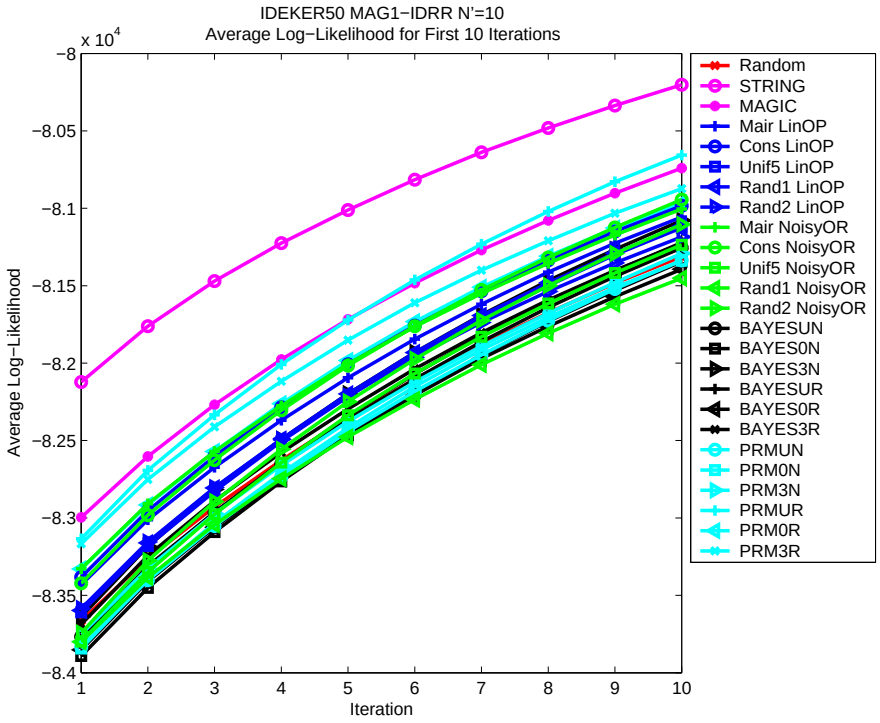


STD $N' = 10$

Figure 7.29: IDEKER50 STD Average Log-Likelihood for First 10 Iterations



MAG1–IDRR $N' = 1$



MAG1–IDRR $N' = 10$

Figure 7.30: IDEKER50 MAG1–IDRR Average Log-Likelihood for First 10 Iterations

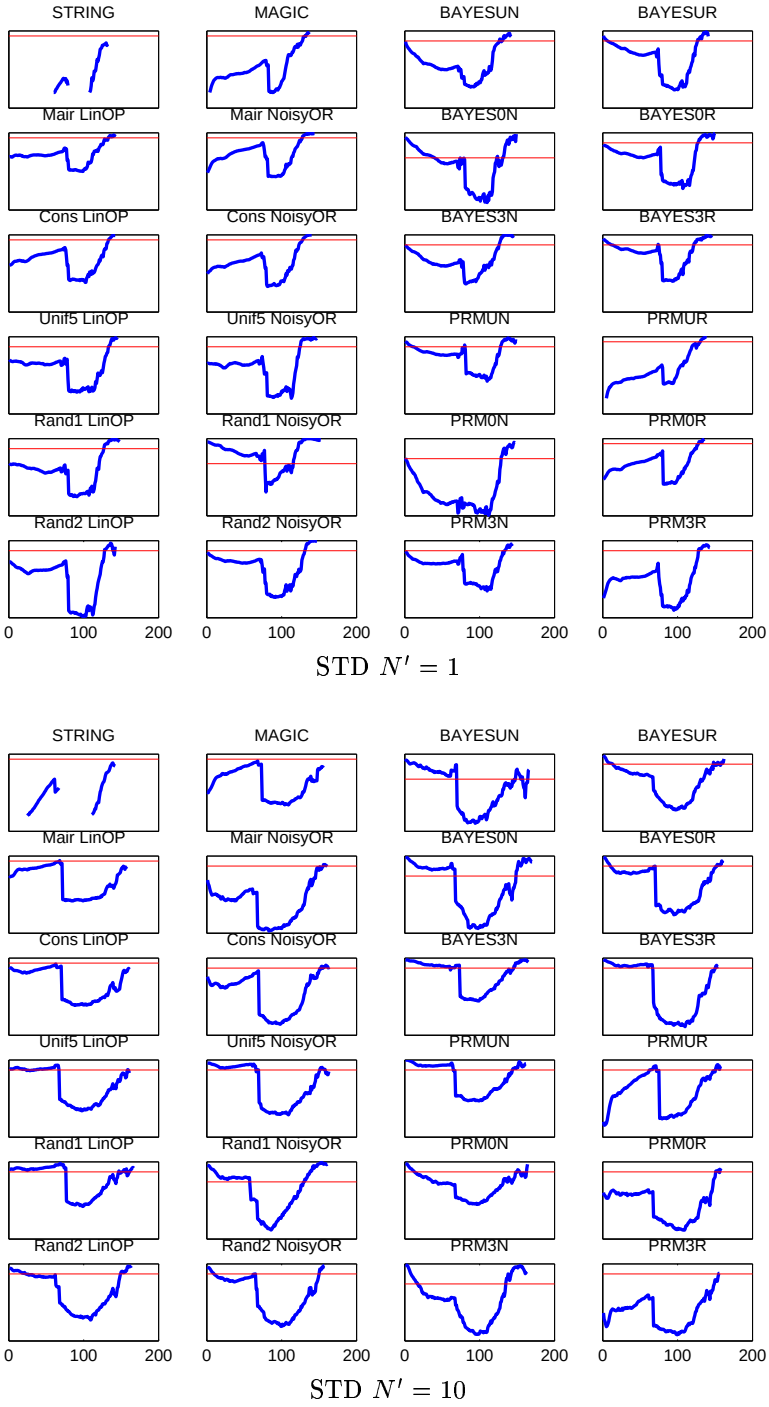


Figure 7.31: IDEKER50 STD Significance of Difference between Log-Likelihood Distributions, per Iteration

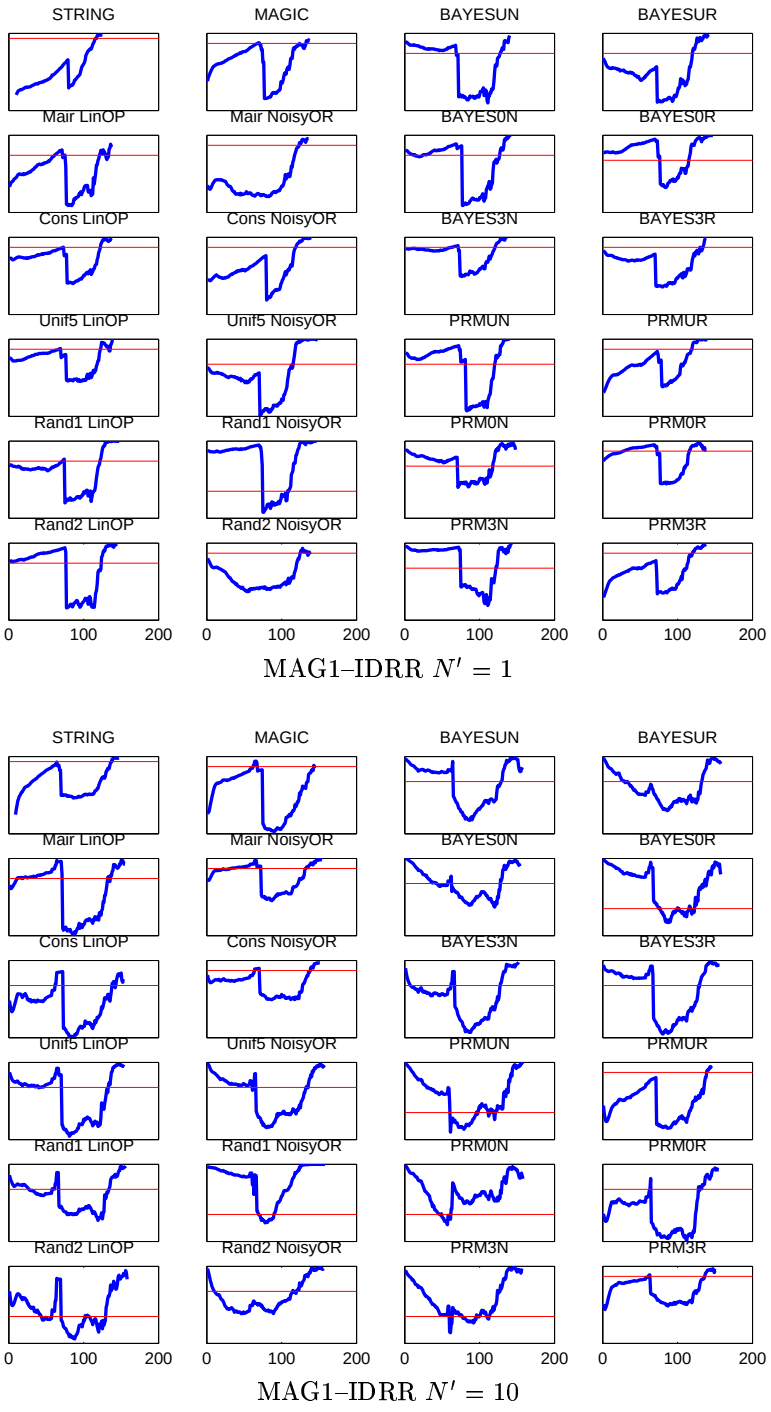
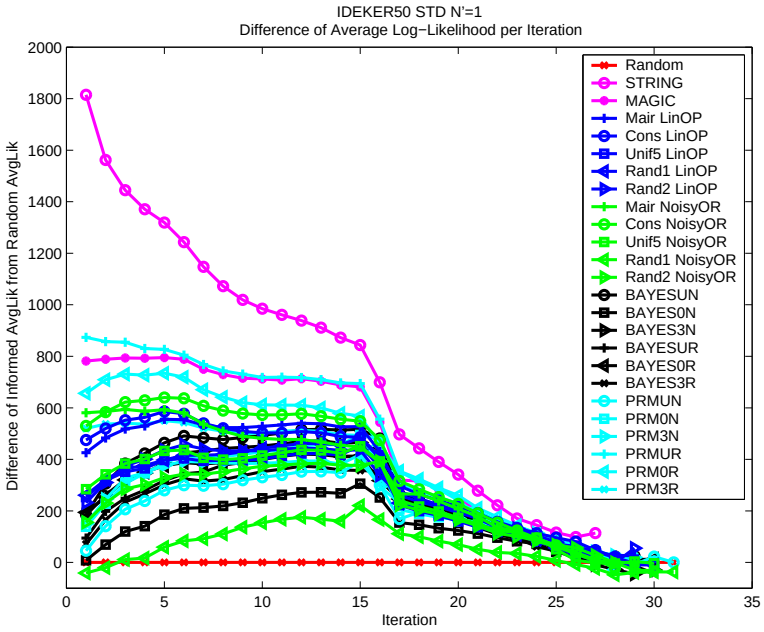
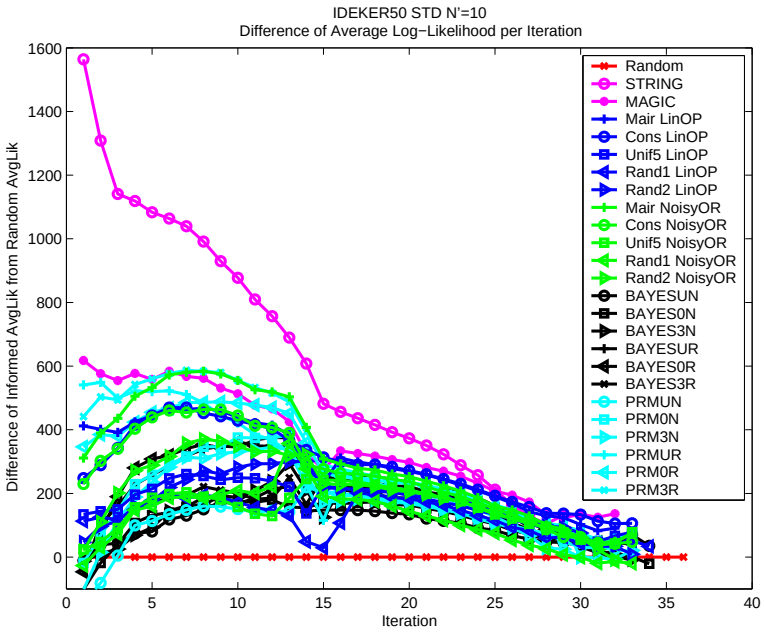


Figure 7.32: IDEKER50 MAG1–IDRR Significance of Difference between Log-Likelihood Distributions, per Iteration

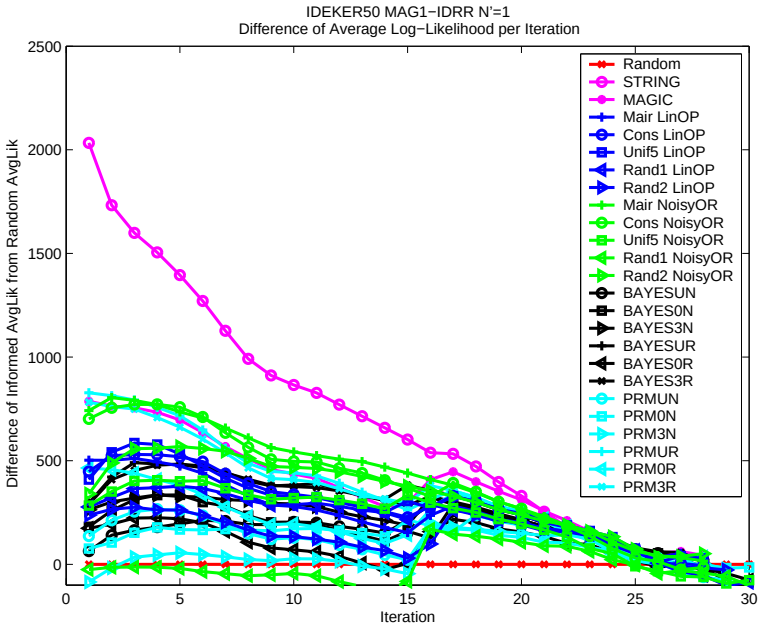


STD $N' = 1$

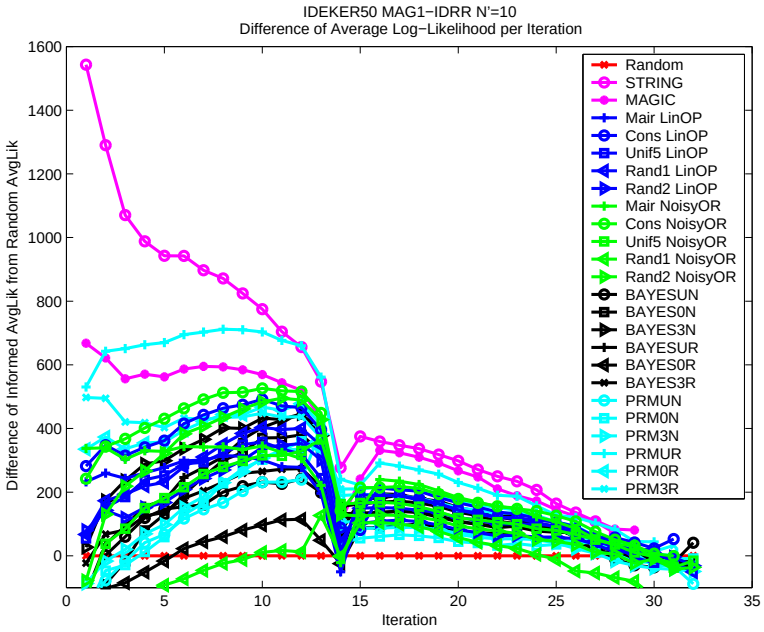


STD $N' = 10$

Figure 7.33: IDEKER50 STD Difference in Average Log-Likelihood per Iteration

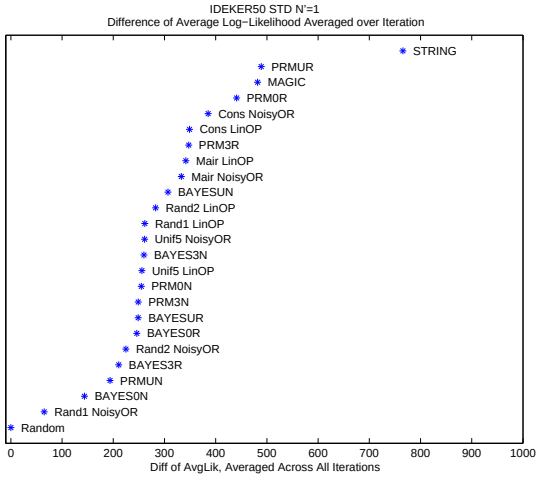


MAG1–IDRR $N' = 1$

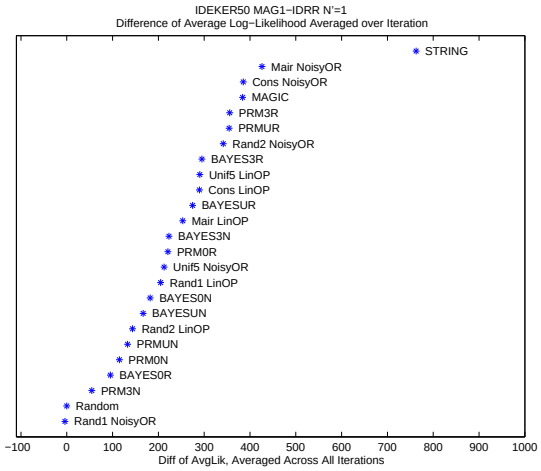


MAG1–IDRR $N' = 10$

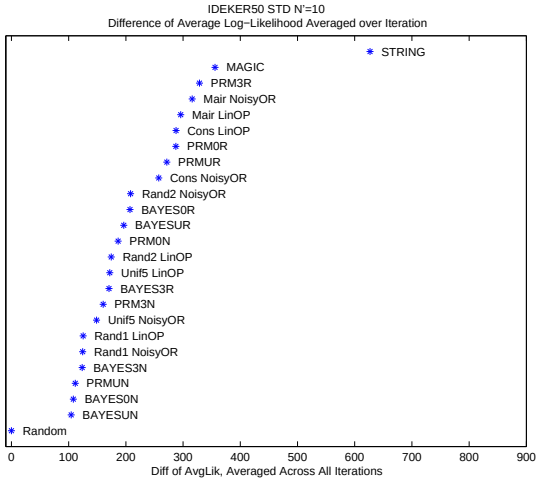
Figure 7.34: IDEKER50 MAG1–IDRR Difference in Average Log-Likelihood per Iteration



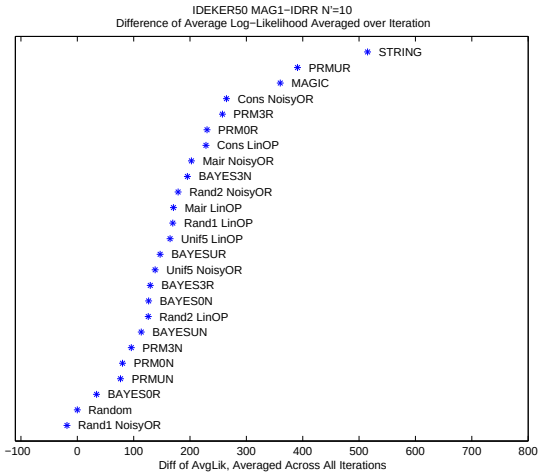
STD $N' = 1$



MAG1–IDRR $N' = 1$



STD $N' = 10$



MAG1–IDRR $N' = 10$

Figure 7.35: IDEKER50 Difference in Average Log-Likelihood, Averaged over Iteration

Chapter 8

Visualizing Gene Interaction Networks

In this chapter, we describe how the consensus likelihood functions of Chapter 5 can be used to develop visualization tools for large-scale datasets. In recent years, biology has become a data-rich science as advances in experimental techniques allow the interrogation of genetic entities on a genomic scale. The rate of data generation has significantly outpaced the biologists' ability to assimilate and explain the observed phenomenon. Before, biologists had designed the experiments for the quantities measured so explanation came naturally, but the hypothesis-free nature of high-throughput data allows mostly description of activity, not explanation.

Gene expression data, as measured by the microarrays described in Section 2.4, represents one example of a large-scale technique applied simultaneously to thousands of genes across numerous experimental conditions. Early approaches to analyzing expression data attempt to garner explanation by reducing the complexity of the problem, primarily through clustering-based techniques which group together genes whose activity suggests correlation between the genes. These groupings were then compared against the existing knowledge available from a collection of low-throughput sources in the hope that some coherent explanation emerged. Even with these computational strategies to reduce the complexity, the burden of explanation still resides with the biologists to review the groupings and recognize plausible explanations.

Figure 8.1 shows such an analysis pipeline for gene expression data, illustrated using the mouse pregnancy and lactation study of Rudolph and Neville introduced in Section 6.3.3. Expression data is measured for over 12000 mouse genes in mammary tissue samples using oligonucleotide microarrays for 11 timepoints spanning from virgin mouse to post-lactation (*i.e.,* involution) [RMH$^+$03, RMP$^+$]. Our colleagues are particularly interested in the set of genes differentially expressed between the timepoints of Pregnancy Day 17 and Lactation Day 2, when the mammary gland necessarily undergoes an enormous change in gene regulation to induce lactation. The mechanisms of the pregnancy/lactation switch are currently not well understood.

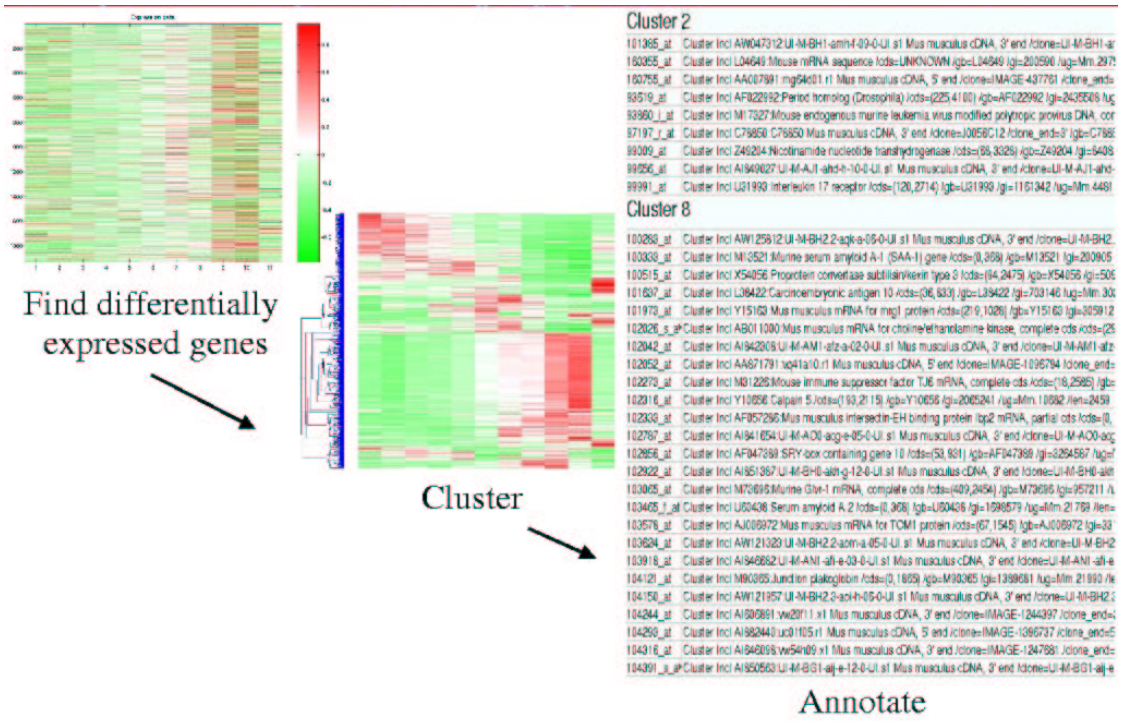


Figure 8.1: Typical Gene Expression Data Analysis Pipeline

Since the majority of the thousands of genes measured by a microarray are not typically affected by any given experimental condition, the first step of analysis is to find the genes with statistically significant differential expression between the two experimental conditions. Significance is typically determined using a test such as the (parametric) t-test or the (non-parametric) Wilcoxon-Mann-Whitney Rank Sum test [DeG70] on replicate samples per condition.

On the left in Figure 8.1, we see a heatmap display of expression data, as described in Section 2.4, for the 2009 genes found to have significant differential expression between the timepoints of Pregnancy Day 17 (P17) and Lactation Day 2 (Lac2). The rows of the heatmap matrix represent the 2009 genes while the columns represent the 11 samples, ordered by time point (Virgin Mouse, P1, P3, P7, P12, P17, P19, Lac1, Lac2, Lac9, Inv2, respectively). For the purposes of display, the expression data for each gene has been transformed by taking the $\log_2$ ratio of the measured expression level in a sample relative to the mean expression level across all samples, then normalized such the resulting vector of expression ratios per gene has mean zero and standard deviation of one across the vector. The colorbar to the right of the heatmap indicates over-expression (red) or under-expression (green) relative to the average expression across samples.

From the analysis so far, we have reduced the complexity from 12000 genes to 2009 and the heatmap display of expression values for those genes clearly indicates stronger differential expression of all genes in the Lac2 and Lac9 samples (ninth and tenth columns), relative to the other samples. However, little further explanation of gene activity emerges. The second component of the analysis

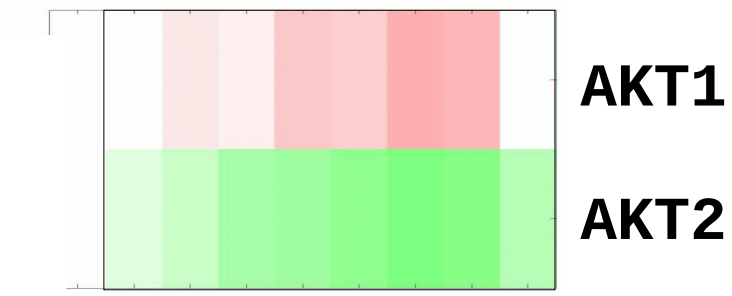
clusters genes based on correlation between expression profiles, shown in the center of Figure 8.1. Hierarchical agglomerative clustering [Har75] using Pearson correlation as the similarity metric reorders the genes according to the tree-structured dendrogram shown in blue on the left of the center plot. The columns remain ordered by timepoint. From the display, the genes along the top of the plot generally begin with over-expression in the virgin mouse (column 1) and become down-regulated in later timepoints, while the opposite behavior is observed for genes along the bottom of the plot. Studying clusters of genes with coordinated expression profiles can potentially reveal the factors affecting or affected by the pregnancy/lactation switch.

Clusters are defined by cutting the dendrogram at specific points and gathering the genes underneath each subtree. The resulting gene groups can then be annotated by gene identity and other additional gene information, as shown in the right of Figure 8.1. Such a list is then given to the biologists who examine the cluster information in order to generate plausible explanations for the cluster behavior.

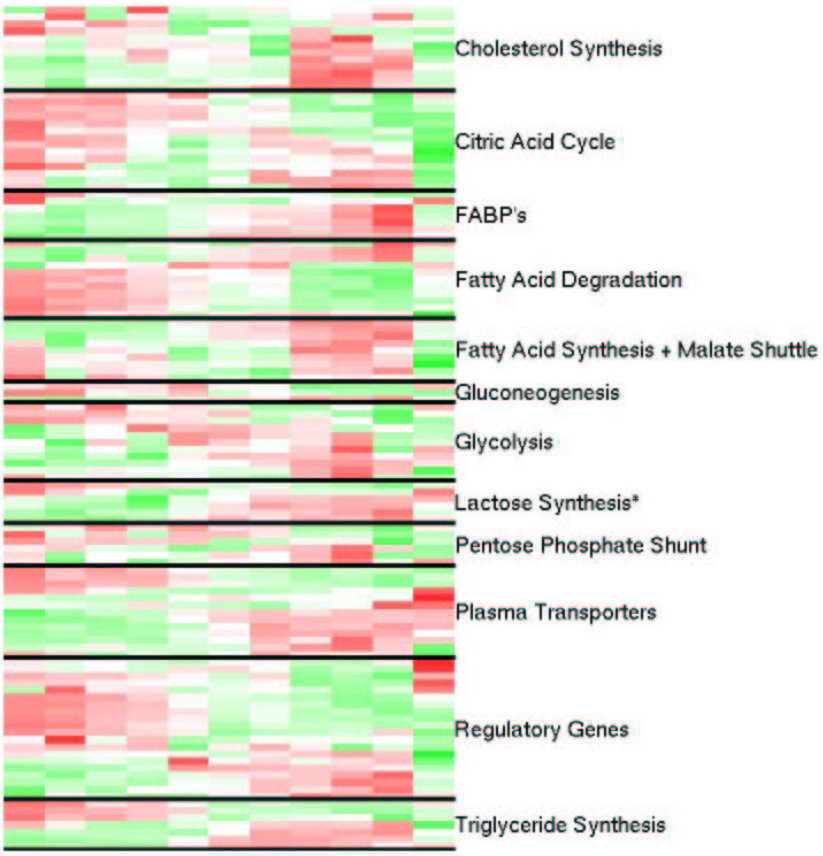
The pipeline outlined above is typical for the majority of analysis. Though the combination of steps has significantly reduced the complexity, the biologists are still left with the burden of making inference from a context-free listing of genes, albeit smaller and better organized. Also, the clustering algorithms typically measure positive correlations between genes, which would miss important relationships, like the one shown in Figure 8.2(a) between AKT1 and AKT2 for our data. Despite the fact that these gene isoforms are known to often display unique phenotypic roles [CTC+01, CA04], they are both protein kinases and important regulatory genes, particularly for the insulin signalling pathway [DSV+02]. This suggests that a display of gene expression data organized by functional category might be a useful alternative to the previous dendrogram display by correlated expression. Figure 8.2(b) illustrates this approach using 119 mouse genes along the vertical axis for the 11 timepoints in the mouse dataset of Neville and Rudolph. The functional categories are listed to the right. However, examining expression data by function is not the answer either since there is no sense of context or connection between the pathways or functions. For all the reasons listed above, we are instead motivated in this thesis to use *interaction networks* as the basis for visualization tools.

In the Introduction, we presented a gene interaction network from Ideker *et al.* [ITR+01], repeated in Figure 8.3 for convenience. Now knowing the limitations of most analysis approaches, we can better appreciate the impact of this example and how it came to motivate the work of this thesis. Genes are represented as nodes in the graph, spatially organized by functional category, where interactions between genes are represented by arcs in the graph. Note that there is generally higher connectivity within a functional category than between categories, though the inter-category links provide context, indicating which functions may be related to other functions. Together, the arcs and spatial organization provide a global view of the gene relationships.

The real power of the visualization is demonstrated when additional information is overlaid on the graph structure, allowing an enormous amount of information to be viewed simultaneously and within a global context. Figure 8.3 studies the global response of deleting the key regulatory gene



(a) AKT Gene Expression



(b) Expression Data Organized by Function

Figure 8.2: Disadvantages of Typical Analysis Pipeline

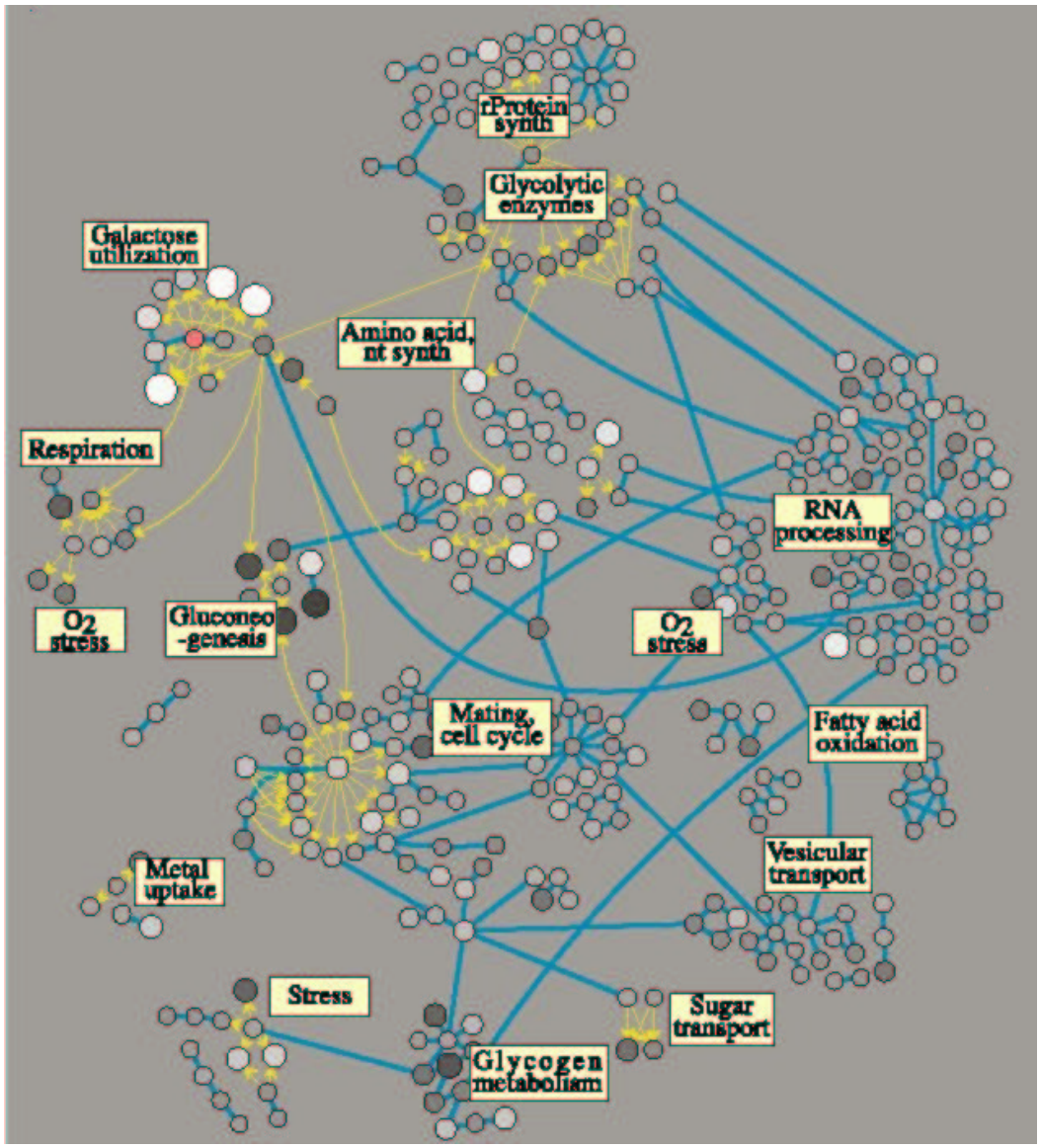


Figure 8.3: Example Gene Interaction Network from Ideker *et al.* [ITR+01]

.

GAL4 (shown as a red circle in the figure) in the Galactose utilization pathway. Gene expression data measuring the resulting change in expression in the deletion strain versus the wildtype response is indicated by varying the size and color of the node. The size of the node indicates the magnitude of the response, where larger nodes correspond to more dramatic changes. The color of the node indicate the direction of the response, where white nodes indicate under-expression or down-regulation, and black nodes indicate over-expression or up-regulation.

Readily apparent from the figure is the area of large white nodes labelled as the Galactose utilization pathway and concentrations of large dark nodes labelled as the Gluconeogenesis and Glycogen metabolism pathways. The visual context provided by Figure 8.3 immediately exposes the underlying biological response of the system: deleting GAL4 turns off the Galactose utilization pathway, depriving the cell of galactose as a source of glucose, which in turn forces the cell to increase production of glucose from raw products (Gluconeogenesis) or obtain glucose from the storage form, glycogen (Glycogen metabolism). This is in stark contrast to the flat functional display of expression data in Figure 8.2(b).

Our goal is to automatically create a spatial organization of genes as in Figure 8.3 using the consensus likelihood functions of Chapter 5. Ideally we would like the graphical structure to be that resulting from the Bayesian network learning of Chapter 7 since the original graph suggested by the combination of interaction information sources has been refined using the additional source of genome-wide expression data during learning. However, since the current amount of expression data available prohibits learning networks for a large number of genes, the consensus likelihood itself provides a valuable solution. We can specify highly probable graph structures over a much larger set of nodes while also incorporating the probability of each arc during the graph layout process to provide additional visual clues.

The intuition behind the use of the consensus likelihood function is to arrange information into *semantic* clusters based on 'relatedness.' For example, if two genes are known to have a physical interaction, they should be considered more related and should be closer in the graph. Or, if two genes have a shared or related function, they should be closer in the graph. Semantic relationships are precisely the quantity captured by the consensus likelihood functions which provide a probability of that relationship for each pair of genes. Thus, we can use the functions to suggest arcs in the graph, allowing the strength of the probability to control layout parameters such as edge thickness or spring constants in the case of the Force-Directed Placement graph layout algorithm [FR91].

In the remainder of the chapter, we present two case studies in mouse using data from our collaborators, Michael Rudolph and Dr. Peggy Neville, mentioned earlier in this chapter as well as in Section 6.3.3. The first case study considers a set of genes extensively investigated by our collaborators, thereby allowing us to validate the efficacy of our approach to visualization using the consensus likelihood functions against a well-studied example. The second case study involves a larger set of interesting genes not yet examined thoroughly by our collaborators, allowing us to explore the usefulness of our approach in de novo analysis. Before presenting these case studies, we review existing approaches to visualization of gene expression data using networks.

8.1 Previous Approaches to Network Visualization

While many researchers have explored the properties of interaction networks, there is currently limited availability of tools which integrate interaction sources with gene expression data. The Cytoscape graphical package [SMO+03] is open source software specifically designed for visualizing molecular interaction networks and integrating those interactions with gene expression profiles and other state data. Though the burden of providing the network structure lies with the user, Cytoscape provides a variety of layout algorithms and facilities for associating additional information with the nodes and edges of the graph. Ideker *et al.* [ITR+01] cite the use of Cytoscape in creating Figure 8.3 and we use this tool in several of the figures in subsequent sections.

Ingenuity Systems©(`http://www.ingenuity.com`) provides a commercial solution which allows a user to upload expression data for genes deemed interesting by the user's prior statistical analysis. The system then identifies relevant pathways and biological functions from a proprietary database using Fisher's exact test to calculate a p-value determining the probability that the association between the genes in the dataset and the canonical pathway is explained by chance alone. The expression and pathway/function information is then overlaid on the graph. Figure 8.4 shows example output of Ingenuity Pathways Analysis©for one sample of expression data using a subset of the mouse genes displayed in Figure 8.2(b). Differential gene expression is indicated by varying the color of the node on a red-green scale, similar to the scale of Figure 8.2(b) or the white-black scale of Figure 8.3. The shape of the node indicates the functional class of the gene. The edges in the graph are annotated by the nature of the relationship between the nodes (*e.g.,* P for phosphorylation, T for transcription). Though the tool integrates a number of sources, there is no probabilistic component associated with the edges and the commercialness of the product limits its availability.

The STRING database [vMHJ+03, vMJS+05] mentioned earlier in Section 2.2.1 and Section 6.2.3 is another public resource for displaying gene networks given a list of genes. Edges between nodes represent interactions based on co-occurrence in protein names in PubMED [1] literature abstracts, on databases of experimental interactions (GRID [SBR+06], MINT [ZMQ+02], BIND [BDW+01], and DIP [XSD+02]), on databases of functional interactions (MIPS [MFG+02] and KEGG [KGH+06]), on co-expression and on computationally predictive methods based on gene neighborhood, gene fusion, and gene co-occurrence (*i.e.,* phylogenetic profiles). A reliability score for each method is estimated based on the percentage of predicted pairs known to be assigned to the same KEGG pathway. The reliability scores are then combined into a measure of confidence via a Noisy-OR function (see Section 5.2). Different versions of a network can be created by varying the threshold for the probability of an interaction. Figure 8.5 shows the network for the set of 331 genes appearing in the galactose example of Figure 8.3 for two different confidence thresholds, 0.2 and 0.9 respectively. An edge is displayed as a collection of colored lines, one for each interaction source, giving a visual indication of the strength of the confidence in each edge. However, currently there is no facility for associating gene expression data information with the STRING networks.

[1] `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?DB=pubmed`

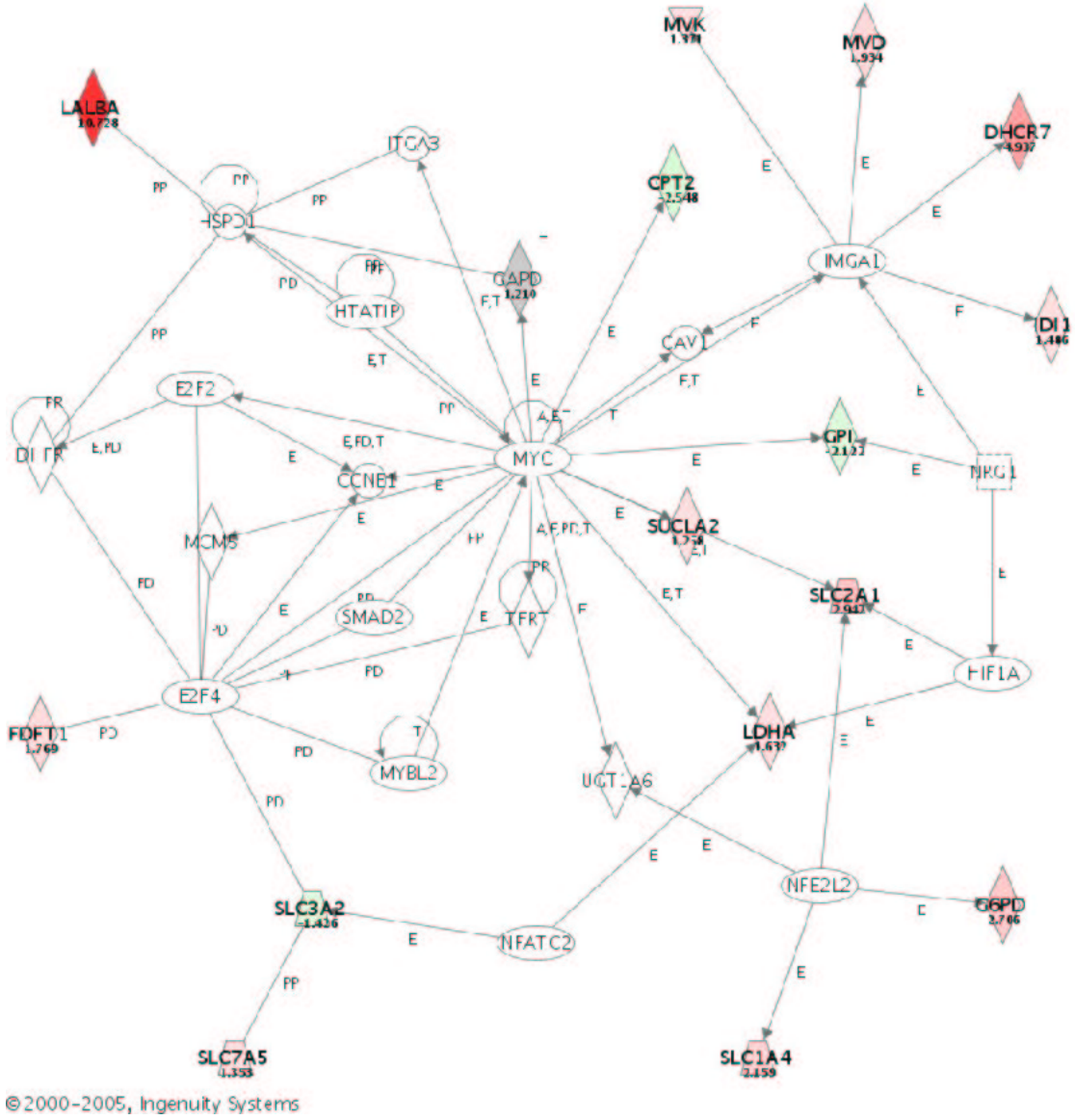
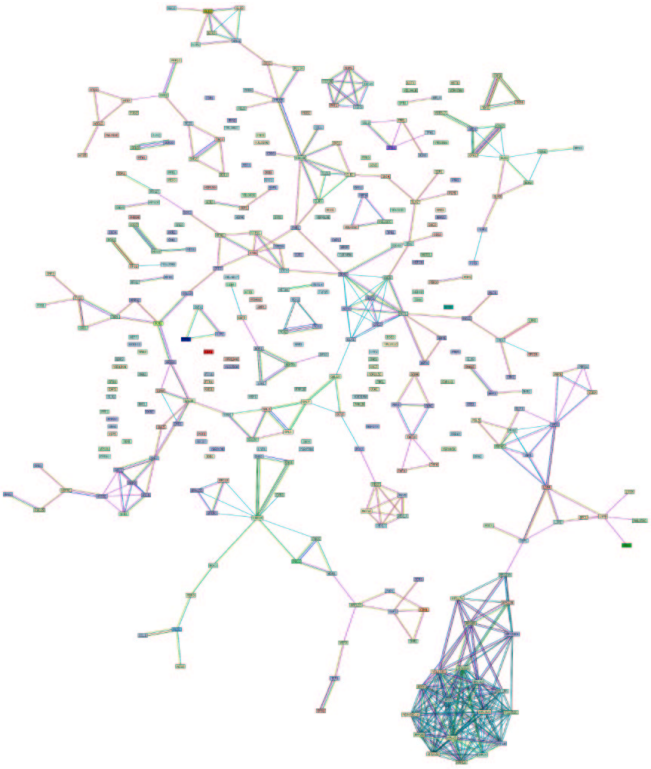


Figure 8.4: Example Network Visualization using Ingenuity Pathways Analysis (Ingenuity©Systems)

(a) Confidence ≥ 0.2



(b) Confidence ≥ 0.9

Figure 8.5: Example Network Visualization using STRING Database

8.2 Datasets

In this section, we describe the results of displaying expression data superimposed on networks created from the high probability edges suggested by a consensus likelihood function. Both examples stem from the study by our collaborators, Michael Rudolph and Dr. Peggy Neville, of gene expression profiles in mouse mammary tissue during the switch from pregnancy to lactation [RMH+03, RMP+]. Providing a framework in which the biologists can investigate their expression data is essential for this application since currently, the mechanisms controlling the transition from milk synthesis to secretion are not well understood.

Oligonucleotide microarrays containing markers for over 12000 mouse genes are applied to 11 mammary tissue samples staged from virgin mouse to post-lactation (involution), using four replicate samples per timepoint. Timepoints include Virgin Mouse (at four days post-ovariectomy), Pregnancy (Day 1, 3, 7, 12, 17, and 19), Lactation (Day 1, 2, and 9) and Involution (Day 2). All displays of expression data in the following sections show data per gene transformed by taking the average of the four sample replicates, calculating $\log_2$ ratios of the (replicate average) expression value in the sample for a given timepoint relative to the average across all timepoints, then normalizing to have zero mean and variance one across the timepoints. Thus a value of -1 represents a two-fold down-regulation of the gene with respect to the average while a value of 2 represents a four-fold up-regulation of the gene with respect to the average.

8.2.1 Well-Studied Mouse Example (MG122)

Of particular interest to our colleagues is the set of genes with significant differential expression between the timepoints of Pregnancy Day 17 and Lactation Day 2 when all the mechanisms must be in place for the astonishing achievements of the next stage: in a single 20-day lactation cycle, the mammary gland synthesizes and secretes milk lipid equivalent to its entire body weight. Figure 8.6 shows a pictorial representation of several significantly over-expressed or under-expressed metabolic genes in the context of a mammary gland cell. For example, along the left of the figure, genes such as Glut1, AldoC and TKT involved in the Glycolysis and Pentose Phosphate Shunt pathways are shown as colored shapes outlined in bold. The blue and turquoise colored shapes denote decreased expression (down regulation) from Pregnancy Day 17 (P17) to Lactation Day 2 (Lac2) while the red and amber colored shapes denote increased expression (up regulation) for the same timepoints.

The figure illustrates our colleagues' current understanding of the metabolic changes undergone by the cell to situate itself for later milk secretion. Synthesis of such an enormous volume of lipid requires energy in the form of NADPH (label shown in green text near the left side of the figure). In order to generate NADPH, the figure shows a diverting of activity to alternative pathways which produce NADPH as a bi-product. For example, on the top left, activity of the Glucose pathway is circumvented through the Pentose-P Shunt in order to liberate two NADPH molecules, evidenced by a significant down-regulation (blue shapes) of the key genes HK1 and GPI and up-regulation (red shapes) of the endpoints, G6PD2 and AldoC, of the Pentose-P Shunt. Additional NADPH is created

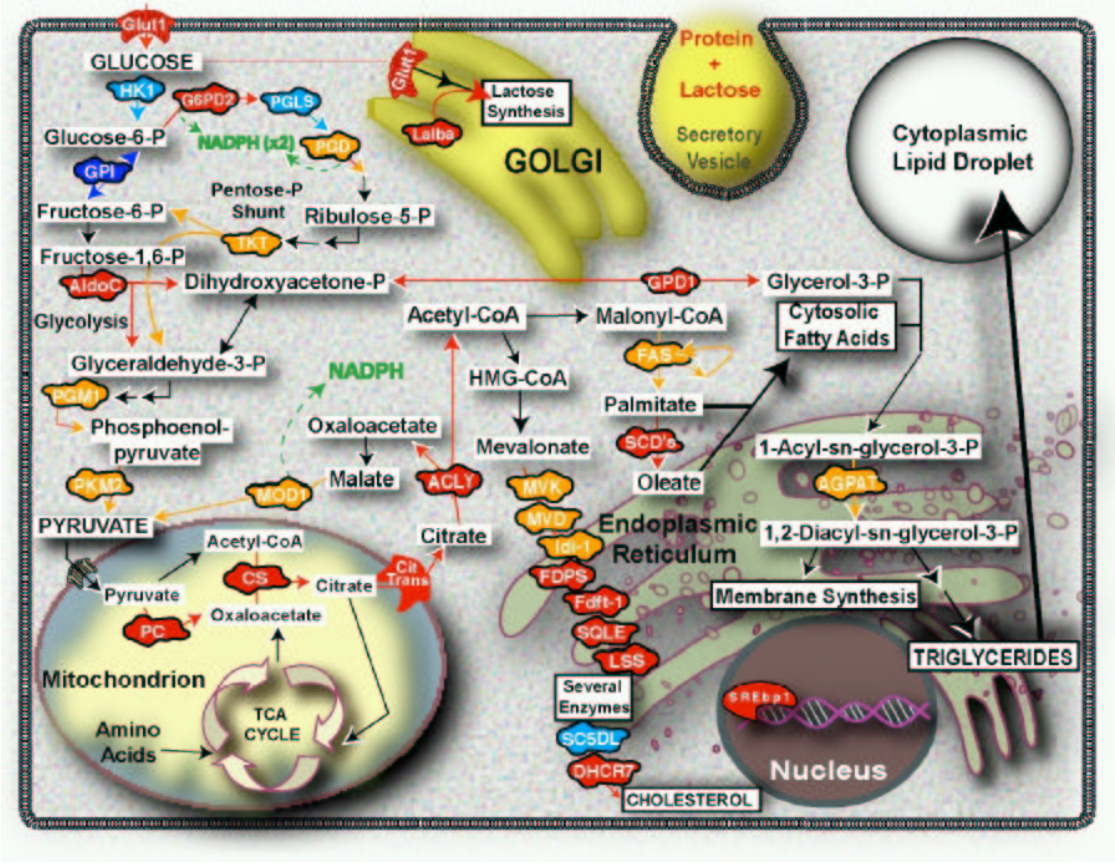


Figure 8.6: Differential Metabolic Gene Expression in Mouse Mammary Tissue between Pregnancy Day 17 and Lactation Day 2 (image courtesy of Michael Rudolph)

from cycling activity through the Malate Shuttle, which uses the mitochondria to continually recycle Pyruvate and produce NADPH.

In addition to producing NADPH, there is massive up-regulation of pathways synthesizing the components of milk, namely Lactose, Triglyceride, and Lipid Synthesis, shown on the right in the figure. Glycolysis plays a key role since 6-carbon molecules, like glucose, are split into 3-carbon molecules which become either phospholipids or triglycerides. Cholesterol Synthesis is also up-regulated since cholesterols maintain membrane fluidity.

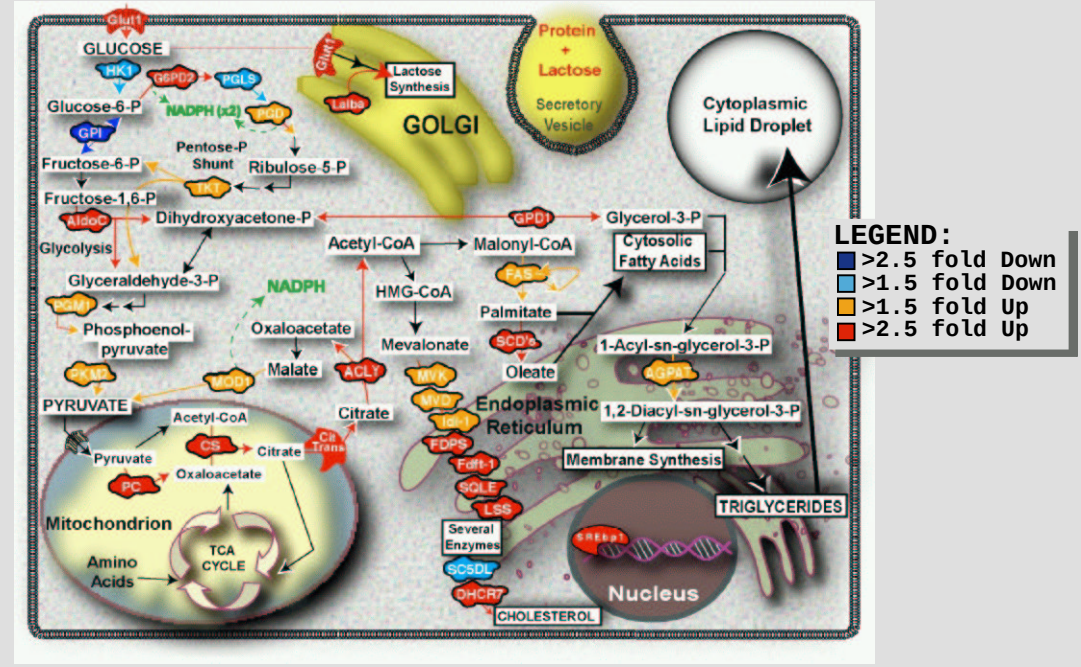
Consensus Likelihood Graph for MG122

The importance of a figure like Figure 8.6 is that it provides an example of the way biologists spatially organize information. The figure represents over two years of careful study by our colleagues using traditional analysis tools. Our goal is to recreate such a display *automatically* using the consensus likelihood functions in conjunction with available graph visualization tools.

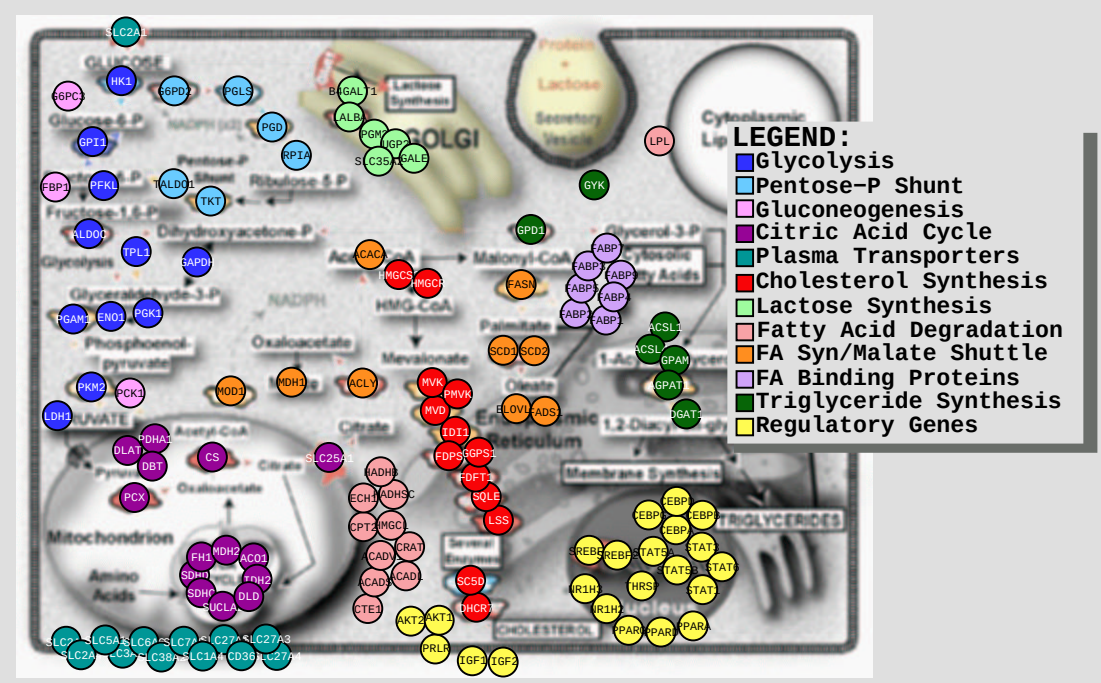
We create a list of 122 genes, the set MG122 discussed previously in Section 6.3.3, by applying a process depicted in Figure 8.2.1. The original illustration is repeated in Figure 8.2.1(a) where 31 genes with significant differential expression between P17 and Lac2 are indicated by colored shapes. The legend for the expression level changes is given on the right. We expand the set of genes to 122 by including genes along each of the pathways shown in Figure 8.2.1(a) regardless of the significance of their expression level. Figure 8.2.1(b) shows the additional genes in the spatial context provided by the original figure. The genes are now represented by circles, colored by the functional category assignment provided by our colleagues. The legend to the right shows the correspondence of color to category.

We apply the consensus likelihood functions of Chapter 5 to the MG122 Gene Set as described in Section 6.3.3. The resulting consensus likelihood matrices were given in Figure 6.17. We create a graph for use in our visualizations by including all edges with a probability $Pr(e_{ij}) > 0.52$ according to the **CONS** NoisyOR consensus function (ignoring edge orientation). Any of the other consensus likelihood functions could be used; we found **CONS** NoisyOR to be reasonable in terms of graph connectivity. Using the threshold of $Pr(e_{ij}) > 0.52$, a total of 467 edges are added to the graph. However, several nodes are completely disconnected from the graph and prior experience with biologists suggests that it is better to provide some context, however vague, for the genes. Thus, we connect each orphan node to its maximum probability neighbor, where that probability must be larger than the default edge probability $\mathcal{R}_0$ provided by our default expert and by definition is less than 0.52. For the MG122 dataset, this adds an additional 20 edges.

We then apply the Organic Layout option in the Cytoscape drawing program [SMO+03] to the graph to create a substrate for further visualization. The result is shown in Figure 8.2.1(b), where the same coloration scheme for functional categories is applied as a node attribute. Figure 8.2.1(a) repeats the original spatial display for convenience of comparison. Most notable is that large clusters of nodes remain together by functional category and the relative order of the categories is similar. For example, along the left, the relative placement of each functional class is preserved for the blue

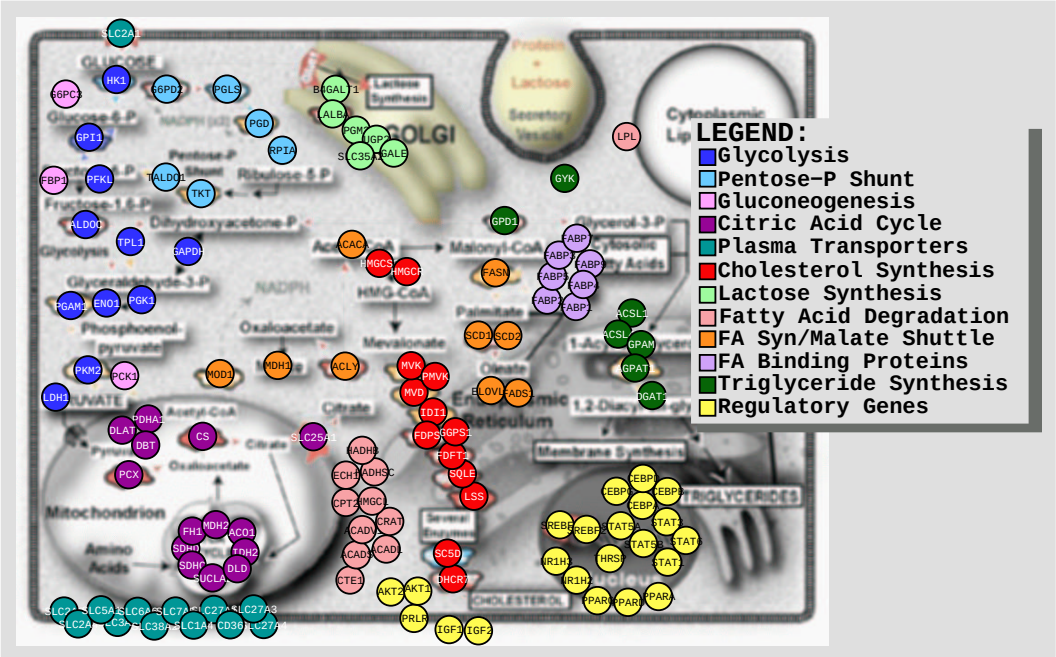


(a) Differential Gene Expression (P17 vs Lac2)
in Mouse Mammary Tissue for 31 Highly Differential Genes
(image courtesy of Michael Rudolph [RMH+03])

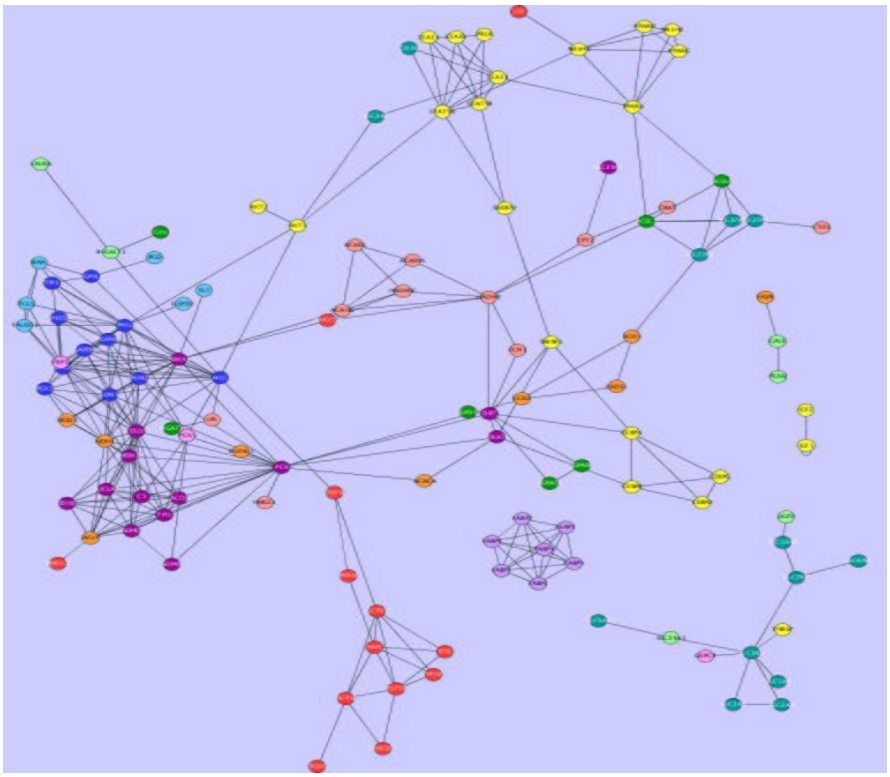


(b) Full Set of 122 Mouse Genes from Relevant Functional Categories

Figure 8.7: MG122 Gene Set for MOUSE

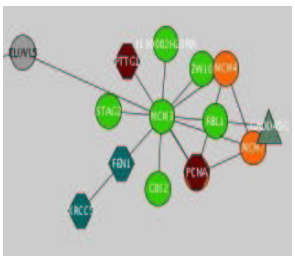


(a) Full Set of 122 Mouse Genes from Relevant Functional Categories

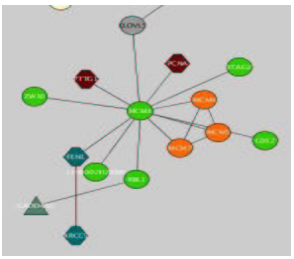


(b) Network formed from **CONS** NoisyOR where $Pr(e_{ij}) > 0.52$ (shown using Cytoscape with Organic Layout [SMO+03])

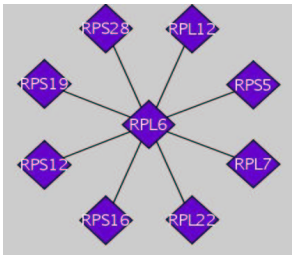
Figure 8.8: Visual Displays of MG122 Gene Set for MOUSE



Cytoscape Organic Layout [SMO+03]



Graphviz Force-Directed Placement [GN00]



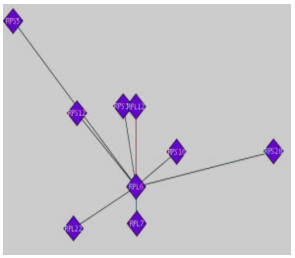


Figure 8.9: Visualization Examples using Force-Directed Placement

Glycolysis, the light blue Pentose-P Shunt, and the dark purple Citric Acid Cycle genes. In terms of our objective to recreate a graph with semantic spatial organization like the galactose example of Figure 8.3, we have achieved success using our consensus likelihood function.

Ideally, we would like to augment the display by incorporating the edge probabilities actively into the layout algorithm. For example, force-directed placement algorithms allow the weights to be associated with each edge so that higher weights, or probabilities of interaction in our case, force the layout to place the endpoints node closer together spatially. Hence, clusters of genes interconnected by higher probability arcs would appear as a compact subgraph in the visualization. Currently, Cytoscape implements a force-directed placement algorithm but does not provide a facility to input the edge weights. The general purpose open-source graphing package Graphviz [GN00] does incorporate supplied weights; a comparison of the two on example subgraphs is shown in Figure 8.2.1. Note that the essentially spherical placement of nodes around a central node in the Cytoscape graphs on the left is distorted according to probability edge weights in the Graphviz versions on the right. Unfortunately, we can only show subgraphs since Graphviz does not optimize the global coordinates during placement so that disconnected subgraphs of the larger graph overlap one another. This leaves us to use the Organic Layout of Cytoscape instead for our larger visualizations. A second option for incorporating the edge information is to vary the width of the arc according to the strength of the edge, akin to the visual weight added by additional sources in the STRING networks of Figure 8.5. However, the current version of Cytoscape provides only discrete edge thicknesses rather than a linear mapping and initial experimentation did not yield pleasing results. Therefore, we do not make use of this facility in the graphs shown later in this section.

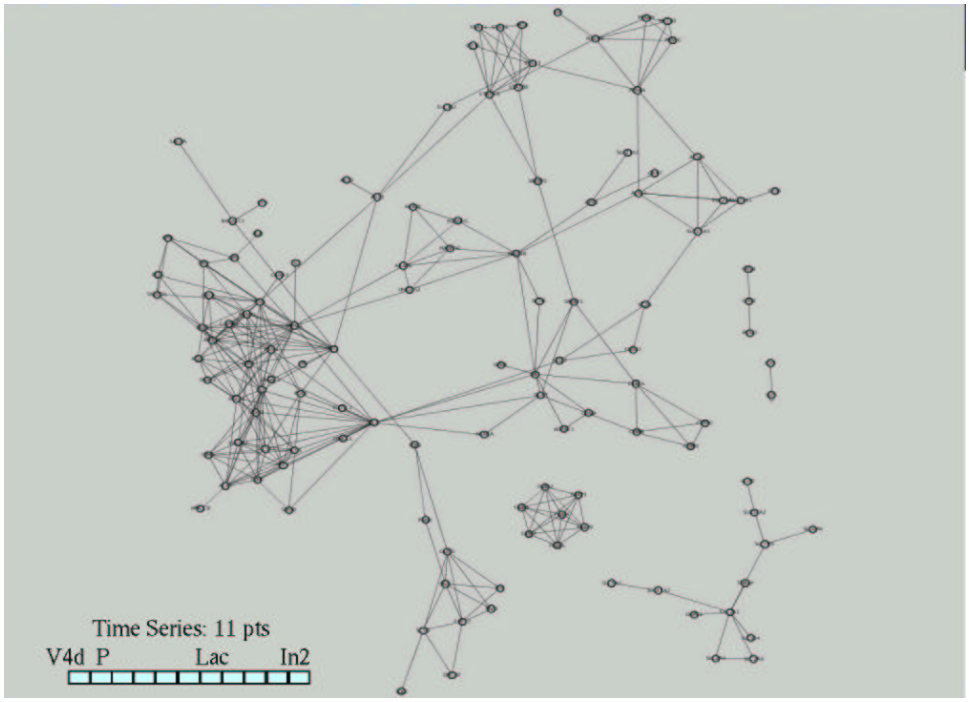
Visualization of Differential Expression for MG122

As illustrated with the galactose example, the real power of visualization comes from animating nodes as gene expression changes. We use the **CONS** NoisyOR graph and superimpose the expression data using variations of the node attributes to indicate magnitude and direction of a difference in gene expression between two conditions. Similar to the galactose example, nodes are colored based on the degree of differential expression for a gene in a sample at a given timepoint to a reference, which for the MG122 data is the average for the gene across all 11 samples. We use red to indicate a gene is expressed above the mean (up-regulation) whereas green indicates the gene is expressed below the mean (down-regulation). Nodes are sized based on the degree of differential expression such that larger nodes mean a greater magnitude of over- or under-expression.

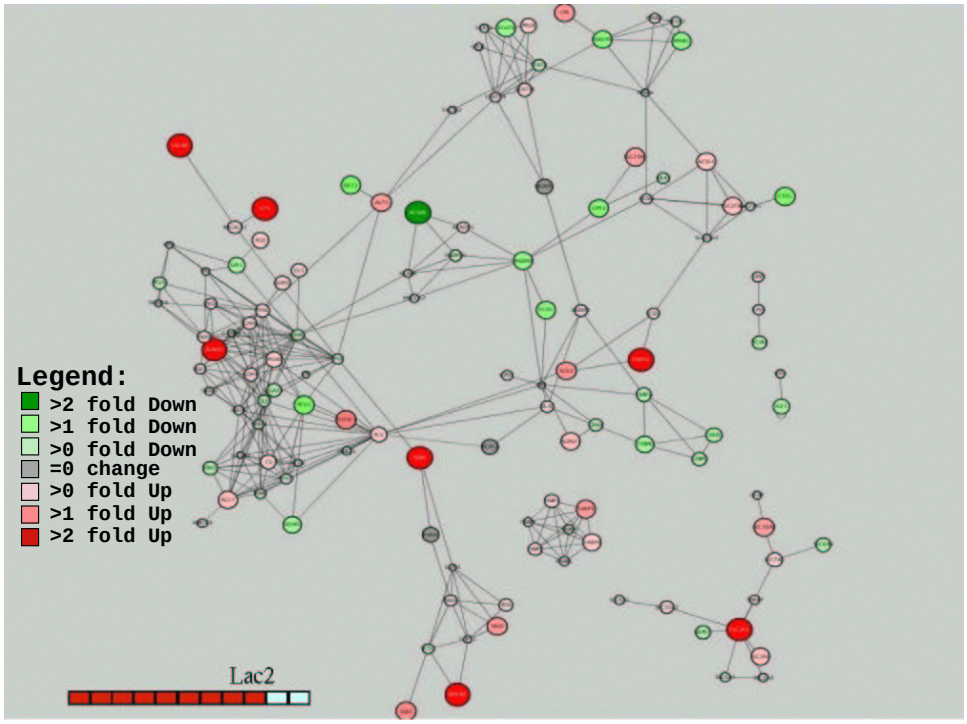
Figure 8.2.1(a) shows the graphical structure of the **CONS** NoisyOR Graph ($Pr(e_{ij}) > 0.52$ and orphans connected) while Figure 8.2.1(b) shows the overlay of expression data for the Lactation Day 2 timepoint from earlier figures. The legend indicates the scale of the coloration where larger circles are generally also darker colors. The differential expression figure can be compared to the functional category overlay in Figure 8.2.1(b) to suggest biological explanations for the expression level changes, akin to the GAL4 response in the galactose example. For example, as would be expected for a lactation timepoint, the large red circles in the upper left corner of the graph indicate up-regulation of the Lactose Synthesis genes.

The fact that our expression data forms a timeseries presents an even more exciting opportunity. By viewing graphs of expression data overlays for successive timepoints, we can observe how the system responds by identifying patterns of changes in gene expression data through time. Figure 8.11 visualizes the expression data for all 11 timepoints. Full size versions are available in Appendix G. The general impression from such a display shows the cluster of predominantly down-regulated (green) genes on the left of the figure eventually become up-regulated (red) through lactation. Creating an animation of the graph frames shows the ebb of expression as groups of genes become turned on or off through time. Our colleague Michael Rudolph used such an animation to identify a subset of genes which could be used to demonstrate more clearly the amenability of our approach for elucidating immediate biological explanation. The panels for the subgraph are shown in Figure 8.12 while the full scale versions appear in Appendix G.

In Figure 8.12, the node to the extreme top left tip of the v-like structure in each subgraph is alpha-lactalbumin (LALBA), the principal protein for the synthesis of the milk sugar lactose. LALBA forms a heterodimer with beta 1,4-galactosyltransferase (B4GALT1) shown just below LALBA in the plots. Hexokinase-1 (HK1) is the node at the bottom of the v-like structure which phosphorylates glucose and its interaction with mitochondria is mediated through AKT [MNB$^+$04], appearing as AKT1 and AKT2 in the center of the graphs. In particular, AKT1 is connected to HK1. We previously discussed the anti-correlation in gene expression between AKT2 and AKT1. Though there are no significant differences between the AKT variants in terms of upstream regulators or downstream targets, there is evidence that their biological and physiological functions are different [PKK$^+$05]. In particular, AKT2 is required for insulin-regulated glucose metabolism, while AKT1 is not.

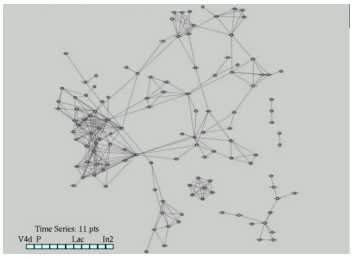


(a) **CONS** NoisyOR Graph ($Pr(e_{ij}) > 0.52$)

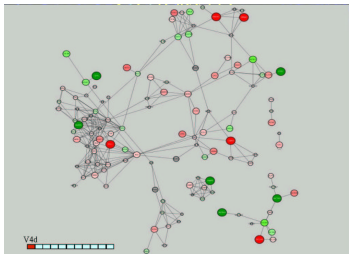


(b) Lactation Day 2

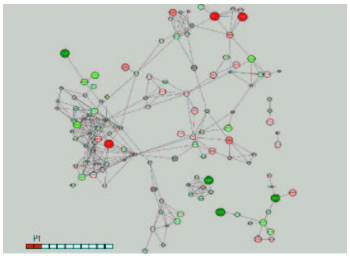
Figure 8.10: Visualization of Differential Expression for MG122 MOUSE Dataset



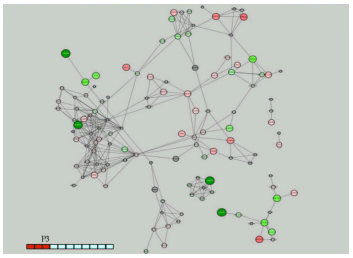
(a) **CONS** NoisyOR Graph



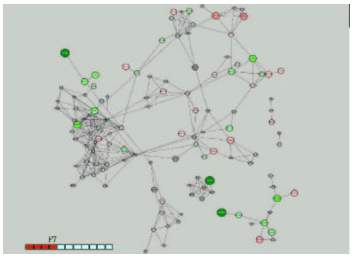
(b) Virgin Mouse



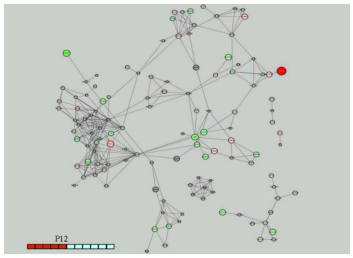
(c) Pregnancy Day 1



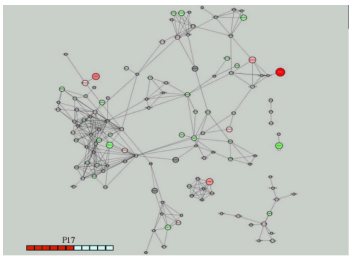
(d) Pregnancy Day 3



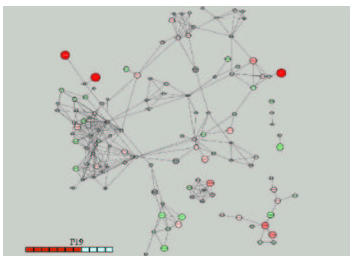
(e) Pregnancy Day 7



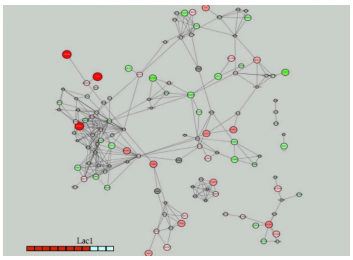
(f) Pregnancy Day 12



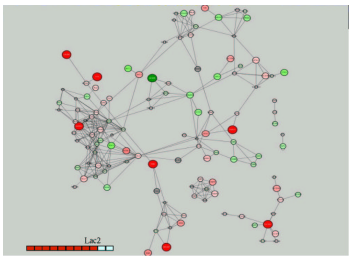
(g) Pregnancy Day 17



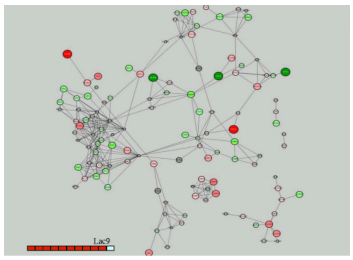
(h) Pregnancy Day 19



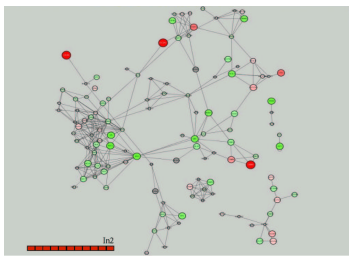
(i) Lactation Day 1



(j) Lactation Day 2

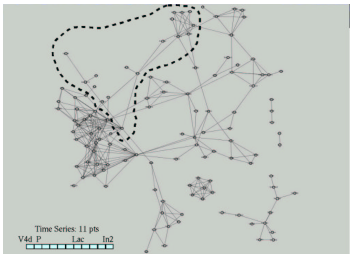


(k) Lactation Day 9

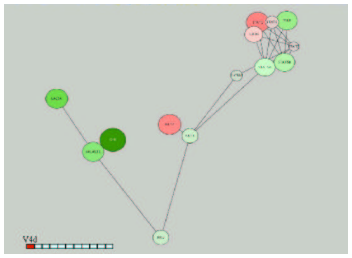


(l) Involution Day 2

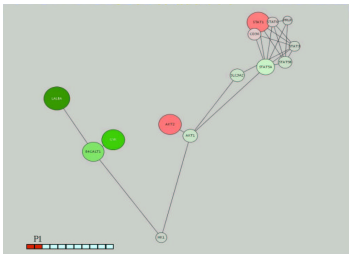
Figure 8.11: Visualization of Differential Expression Timeseries for MG122



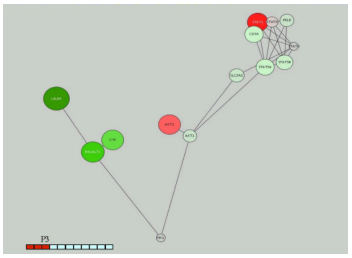
(a) Selected Subgraph



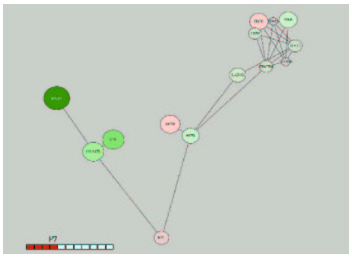
(b) Virgin Mouse



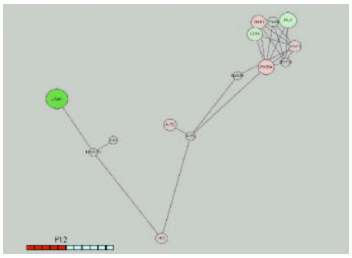
(c) Pregnancy Day 1



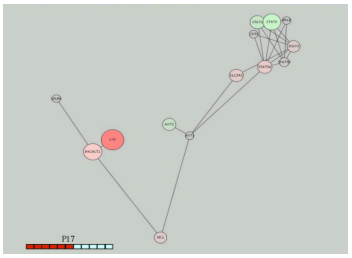
(d) Pregnancy Day 3



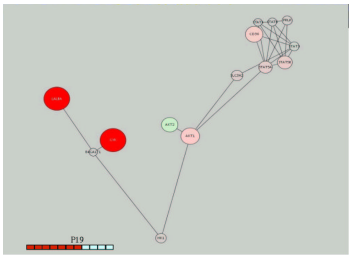
(e) Pregnancy Day 7



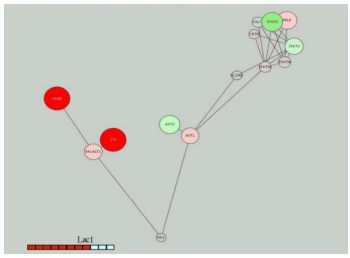
(f) Pregnancy Day 12



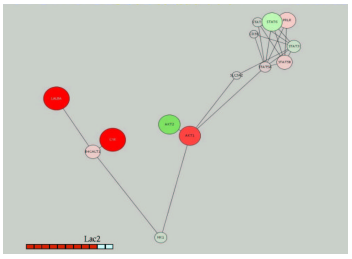
(g) Pregnancy Day 17



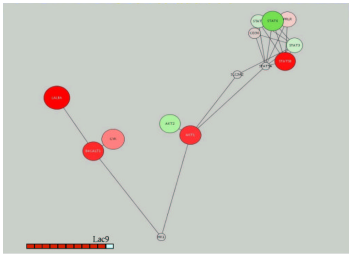
(h) Pregnancy Day 19



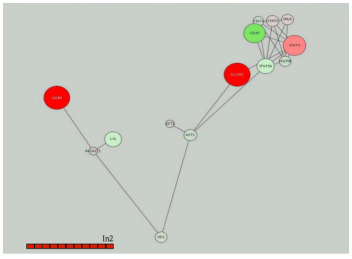
(i) Lactation Day 1



(j) Lactation Day 2



(k) Lactation Day 9



(l) Involution Day 2

Figure 8.12: Visualization of Differential Expression Timeseries for MG122 Subgraph

The AKT genes are connected to a cluster of Regulatory Genes including STAT1, STAT3, STAT5A, STAT5B and STAT6, shown together with the prolactin receptor PRLR to the upper right of each panel. The STAT5 and AKT genes are part of the JAK-STAT signalling pathway [KGH+06]. Each of the STAT regulatory genes can have distinct downstream targets [CW03] The STAT3 regulatory protein controls AKT1 expression [PKK+05]. Presence of the hormone prolactin signals the onset of lactation through the prolactin receptor protein PRLR, shown as the top-right most gene in the Regulatory Gene cluster. Prolactin-induced lactogenesis is mediated through the STAT5 genes [NAY+00].

Viewing the changes in gene expression shown in the panels in Figure 8.12, combined with the above context of known roles for each gene, reveals the following possible biological explanation of metabolic activity in the lactating mouse. Considering the top right cluster in Figure 8.12(b) for the Virgin Mouse, regulatory control shifts from the STAT1/STAT3 proteins to the STAT5 family by Pregnancy Day 2, shown by a transition from red to green for the genes near the top of the cluster and the opposite transition for genes near the bottom of the cluster. The large green circle to the top right of the cluster in Figure 8.12(b) is PRLR, whose expression also increases towards lactation. Increased STAT5 activity results in a shift from using AKT2 to using AKT1, demonstrated by the flipflop of red and green for the circles in the center of each graph as time progresses. Increased AKT1 activity is quickly followed by an increase in the Lactose Synthesis genes on the left arm of the v-like structure, where the large green circles transition to large red circles through lactation.

This example was generated by our colleague after only a short time of experimenting with our visualization tool. He later commented that he felt he gained a better understanding of his data using our network for a few minutes than he was able to gleam from the two years he spent using other tools. Such feedback provides heady and practical validation of the usefulness of creating visualizations from the consensus likelihood functions.

8.2.2 De Novo Mouse Example (MG449)

Since the set of genes considered in the previous example is a set previously studied intensely by our colleagues, we create another example where the relationships between the highly differentially expressed genes is not yet well-examined. We obtain a list of the 2009 most differentially expressed genes (either over-expressed or under-expressed) between the timepoints for P17 and Lac2. We choose the 74 among them with a $\log_2$ ratio of P17 to Lac2 expression greater than 2.1 (or -2.1), and expand that list by considering all genes on the pathways in common with the 74 genes also in the list of 2009 genes, where pathway assignments are given by the GenMAPP [DSV+02] resource. We refer to the resulting set of 449 genes as MG449. The consensus likelihood matrices for this gene set was shown in Section 6.3.3. We again use CONS NoisyOR, with a threshold of $Pr(e_{ij}) > 0.6$ and connect the orphans to their maximum probability neighbor. We used Cytoscape by apply the Organic Layout algorithm to the CONS NoisyOR structure. The result is shown in Figure 8.13, where the shape and color of the nodes represent the pathways assigned by GenMAPP. Since the genes could have multiple pathway assignments, we superimposed on each node the GenMAPP

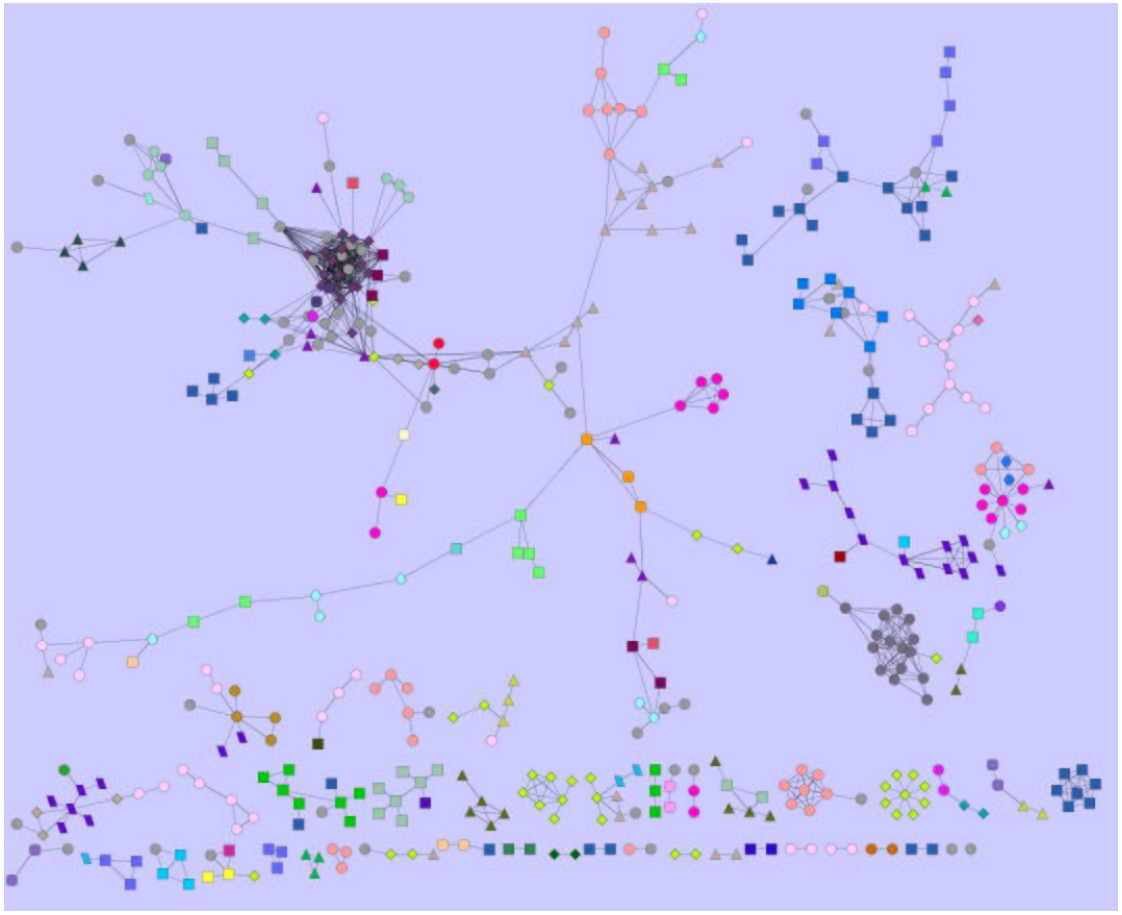


Figure 8.13: Network for MG449 formed from **CONS** NoisyOR ($Pr(e_{ij}) > 0.60$) with bi-connected components colored by shared GenMAPP Pathways [DSV+02] (shown using Cytoscape with Organic Layout [SMO+03])

pathway most commonly represented within each bi-connected component of the graph. Gray nodes indicate that no pathway could be assigned using this strategy. An interesting consequence of such a policy of assignment is that a colored node connected to a node of a different color indicates that each endpoint of that edge shares both pathways represented by the colors and differ only due to their membership in distinct bi-connected components. This allows us to determine which pathways are related to which other pathways and which genes provide the bridge between them. As can be seen in the graph, the same tendency for large clusters of similar colors witnessed for the MG122 example is evident for the MG449 example.

As remarked previously, the consensus likelihood matrices for the MG449 example were much more coherent with respect to their correspondence to biological function, in that the matrices had strong groupings along the diagonals in Figure 6.18. Thus we might expect that the visual display would reveal similar preservation of GenMAPP pathways. However, the results on both datasets are very encouraging since they suggest that viewing gene expression data in graphs exhibiting strong

functional cohesion allow quick interpretation by biologists, and our decision to include non-physical or non-functional interactions (such as the correspondence between AKT1 and AKT2) can provide key links in the global structure that would be ignored otherwise. Though we have not currently performed further analysis of the MG449 graph of Figure 8.13, our colleagues are very excited for the chance to explore the identity of each of the colored clusters and view their expression data throughout the timeseries in the context of in the interaction graph found by our consensus likelihood functions.

Chapter 9

Overall Conclusions and Future Work

In this thesis, we have demonstrated techniques for integrating a diversity of information sources, of varying scale, reliability, and density of coverage of a problem domain, for purposes of generating a probabilistic consensus belief of interaction between variables in the domain. Though our chosen application was one in Molecular Biology, namely to create networks of gene interactions, this work can also be applied to other domains. For example, multi-agent systems can benefit from creating such a consensus likelihood since each self-interested agent separately learns a model of the domain, yet the agent can influence the actions of other agents and must therefore have some knowledge of the other agents' beliefs [BDS00]. Computing consensus beliefs also has advantages in consolidating doctor expertise in the medical field [MC01]. There is work in solving crossword puzzles by computer where distinct expert modules specialize in solving specific types of clues and are then combined probabilistically to search for an optimal solution [KSL+99]. A special issue in the Journal of Machine Learning Research, entitled "Fusion of Domain Knowledge with Data for Decision Support" [DLMP03] further emphasizes the importance of the potential impact of our work.

For the problem domain of Molecular Biology, this thesis illustrates the utility of creating consensus likelihood functions over gene interactions in two distinct applications. The first is in providing structural priors for learning Bayesian networks from gene expression data, where using a prior distribution over interactions between genes can significantly increase the speed and quality of search for high scoring Bayesian networks. The experimental results of Chapter 7 demonstrate that the use of informed structural priors improves learning over an uninformed prior, even when no distinction is made between the reliability of the sources, or in some cases even when the sources are given random reliabilities. These results hold in the case where the experts generally assert plausible interactions. On the other hand, under a setting where we could control the maliciousness of the experts, we demonstrated that grossly incorrect assignments of reliability can results in badly informed priors

which in term can dramatically influence the search for models, causing the learner to perform worse than when searching from models with uninformed structural priors. These results show that using informed structural priors can have a significant impact on learning in data-limited applications, in contrast to the conclusions offered in previous work in Bayesian network learning where the majority of investigations have occurred in data-sufficient settings. As a consequence, the general impression in the Bayesian learning community up until now has been an emphasis on specifying parameter priors and ignoring the importance of structural priors.

The second application of consensus likelihood functions is their use in creating visualization tools for exploring other large scale biological datasets. In Chapter 8, we provided results for examples in the mouse genome of graph networks on genes which include arcs representing the highly-probable interactions according to a consensus likelihood function. These networks provide a powerful and useful working model of interactions so that later superposition of additional data on the graph structure allowed biologists to better understand an otherwise overwhelming amount of information. We showed that even the less computationally intensive methods for computing consensus likelihood were an effective choice for creating networks on a large number of genes. Moreover, we showed that including diverse types of prior information spanning across function, cellular location, DNA (or protein) sequence information, and physical and genetic interaction when forming the consensus likelihood function captures valuable relationships among genes that would not be apparent using one source alone.

In the results for Bayesian network learning of Chapter 7, the two strongest performers were the consensus likelihood functions provided by outside research groups, namely STRING and MAGIC. The crucial difference between these resources and the methods developed in this thesis was the conservativeness in estimation of interaction probabilities. Very few interactions had high probability under these consensus functions. This is due in part to the fact that for STRING, the emphasis was on highly evolutionarily conserved gene groups, which generally imply conserved biological function, and for MAGIC, the purpose was in functional relationship prediction. It makes sense that these matrices would perform well when used to learn models on gene expression data, since it is generally accepted that genes with a common function are often co-expressed [WKB05].

The techniques for computing consensus likelihood functions introduced in this thesis, however, were designed to be as inclusive as possible in terms of the different sources used to infer interaction, not just basing those inferences on functional information. The reasoning is two-fold. First of all, not all patterns of co-expression can be explained on the basis of functional information so our consensus likelihood functions are meant to offer alternative hypotheses. Secondly, our motivation for creating consensus likelihoods was not solely for the purposes of aiding induction of regulatory networks from gene expression data. The second part of the thesis demonstrates that the consensus functions have a larger scope of applicability, such as providing a basis for visualization tools.

However, we can use the lessons learned from the Bayesian network results to improve our techniques. In particular, we can impose further restrictions on the combination functions to refine the quality of the expert data. For example, we can integrate cellular location to identify possible

false positives in the interaction data sources, since genes are less likely to interact if they occupy distinct compartments of the cell. Also, from the previous topographical studies, it has been shown that interactions between one gene and neighbors with no further interactions are likely false positives [SSH03]. We could use these types of information within a combination function, rather than treating the sources as independent as in NoisyOR and LinOP, to weight individual interactions from the different sources.

The BAYES and PRM consensus operators attempted to capture such correlations among sources and in fact, the PRM **Ratio** variants performed well in yeast. Stronger performance might be found for BAYES (and PRM) if the set of genes upon which the model was learned had better coverage by our expert sources than the set we chose from the galactose study. In mouse, the consensus likelihood matrices for the BAYES versions, as well as the PRM variants, demonstrated remarkable correspondence with the functional categories, suggesting there are likely fewer false positives than in the yeast matrices. Enough overlap of the sources existed to enable BAYES (and PRM) to amplify the true interactions by capturing correlations between sources. As more mouse gene expression data becomes available, it would be interesting to repeat the study in mouse.

Further refinements can be made to the consensus likelihood functions by including more and different types of experts. Protein sequence characteristics, such as the hydrophobicity and polarity of individual amino acids comprising the protein, can be used as additional attributes for inferring interactions [DMHK95]. Also, our experts might include information about co-occurrence of gene names in literature abstracts, an information source which was recently shown to have a greater ability to predict protein function than using experimentally verified interactions (*i.e.,* our explicit experts) [GHG, SPR$^+$03].

Overall, this thesis represents a first look at the benefits and limitations of integrating heterogeneous data sources for important applications in Molecular Biology. The demonstrated success of our initial approach encourages further exploration of strategies which can more accurately predict true gene relationships from the collection of weakly suggested or sometimes unreliable relationships available from our prior information sources. Future research can only improve upon the conclusions found here.

Appendix A

Discretization

One of the strengths of Bayesian networks is the ability to combine both real-valued data and categorical data within a unified probabilistic framework. Learning algorithms exist for multinomial nodes as well as many well-behaved distributions, such as those from the exponential family, the logistic distribution, or noisy-OR nodes. Even though Bayesian networks can handle continuous data, there are many advantages to using networks with only discrete variables, the most important of which is perhaps that many implementations of Bayesian network learning are optimized for tabular nodes. Under certain reasonable assumptions, computing the posterior probability of a discrete model given data has a closed form solution.

Outside of Bayesian network learning, discretization of continuous data can have many advantages in general. For most popular machine learning algorithms, discrete feature spaces are required. Although as mentioned for Bayesian networks, even those that can handle continuous attributes often benefit from discretization. Some discretization methods can be viewed as a form of knowledge discovery, revealing critical values in the continuous domain [KS96]. Discretization might also reduce error in domains where the features are known to be inherently discrete yet become continuous as a result of sampling noise [Har01]. In data-limited applications, discretization can help reduce dimensionality because continuous distributions typically have more degrees of freedom [Har01]. While parametric assumptions can also address the problem of dimensionality, discretization can better approximate the true distribution for data where such assumptions are clearly violated [KS96].

Even when sufficient data is available for learning, discretization can increase the speed to induce a model by limiting the search space [DKS95, KS96]. The reduced complexity of discrete models may also allow easier interpretability and visualization [DKS95, LM98]. Discretization can provide a measure of robustness against error and reduce the likelihood of overfitting [FW99, Har01]. Overall, discretization methods can offer a reduction of complexity while preserving the major qualitative relationships between the attributes [Har01].

For our task of learning models of gene interaction from gene expression data, we choose to discretize not only because we wish to make use of optimized Bayesian network learning algorithms but because we have limited data available for potentially thousands of genes, and because it might

be reasonable to assume that gene expression levels are inherently discrete [Har01]. The underlying biochemical events of transcription involve individual molecules and it is because of the difficulty in measuring expression levels on that scale that we pool large numbers of cells; the noise in this process translates to a continuous variable. Also, if transcription levels can be captured by such a rough categorization as saying expression of a gene is off/low/medium/high, then the relationships of interest among genes might be sufficiently described in essentially qualitative terms, such as asserting that one gene turns on the expression level of another gene, provided a third gene is present. Discretization, in our context especially, provides a valuable compromise between reducing complexity and preserving qualitative relationships.

In this chapter, we first characterize the dimensions along which the discretization techniques differ. In Section A.2, we then introduce the 11 distinct discretization methods and their variants (31 in all) used for our comparisons. Two of the methods, Individual Correlation of Discrete to Real (IDR) and Pairwise Correlation of Discrete to Real (PDR), are completely novel while a third method, Nearest Neighbor (NN), represents a new implementation of a general discretization concept. Section A.3 describes our experimental protocol in terms of the evaluation metrics and datasets used. We present the experimental results in Section A.4. To our knowledge, this study represents the first, and indeed largest, study of its kind for unsupervised discretization techniques.

A.1 Characterization of Discretization Methods

Discretization methods can be characterized along six different axes [BV98]: *supervised* versus *unsupervised*, *error-based* versus *entropy-based*, *user-specified granularity* versus *automatically determined granularity*, *top-down* versus *bottom-up*, *global* versus *local*, and *univariate* versus *multivariate*. The first axis distinguishes between supervised and unsupervised techniques depending upon whether class information is used during discretization or not. In the supervised learning community where many learning algorithms require discrete attribute spaces, the discretization techniques typically use the classification variable while discretizing, so by analogy are referred to as *supervised* or *contextual* discretization methods [DKS95, RRS96]. Unsupervised discretization is relevant in domains where either no class variable exists or the class is not relevant to the particular learning task, such as learning a generative model versus a discriminative one. We concentrate here on unsupervised methods since our task is to learn relationships between all genes. A good review of supervised discretization methods can be found in Dougherty *et al* [DKS95].

The second axis of distinction, error-based versus entropy-based, applies only to supervised discretization methods. Most supervised techniques attempt to minimize either class information entropy or classification error when evaluating candidate thresholds for discretization. As mentioned, we consider only unsupervised discretization methods here, thus we refer the reader again to the review by Dougherty *et al* [DKS95] for more detail.

The third distinction is made based on whether the user specifies the target number of discretization levels or whether the number of bins is determined automatically by the discretization method. Since discretization results in a loss of information, the choice of granularity of the discretization is very important. Key relationships between the variables can be lost by using either too few or too many bins. While we acknowledge the importance of this issue, in order to make fair comparisons between the methods, we consider mainly techniques with a user-specified number of bins. The automatic-determination methods we do consider decide the number of bins by an iterative algorithm that terminates when reaching some optimization criterion. For our purposes, we simply iterate until reaching the target number of bins.

The fourth distinction of top-down versus bottom-up describes the type of granularity-determining iterative algorithm mentioned above. The target number of bins is achieved either top-down by starting with a single interval containing all values for a feature and partitioning this interval into smaller ones, or bottom-up by starting with an initial number of intervals and merging adjacent intervals. Examples of such iterative algorithms are described fully in Section A.1.1 and are included in the host of algorithms we consider for our comparison.

The fifth characterization by global versus local refers to whether the entire (global) value domain of a numeric attribute is used during discretization or only localized partitions of the instance space. A familiar example of local discretization from supervised learning is C4.5 [Qui93], which discretizes an attribute at each node in the decision tree by examining only those instances assigned to the current node. Most techniques, however, are global techniques.

The sixth distinction, univariate versus multivariate, depends on whether or not feature interactions are taken into account. In some references, this distinction is termed *static* versus *dynamic*. In univariate or static methods, one feature is discretized independently of the other features. Dynamic, or multivariate, methods consider the interdependencies between the attributes while discretizing. The importance of this distinction is illustrated in Figure A.1. On the left, the sorted values for two variables X and Y are shown. Individually, the choice for how to discretized the variables into three bins in unclear. However, on the right, plotting X versus Y shows a clear dependency between their values; the dashed lines show the optimal 3-bin boundaries. Figure A.2 shows another example involving three variables where the binary-valued third variable, depicted by dot size, is a function of the values of both other variables.

The distinction of univariate or multivariate becomes important in our context since we wish to preserve the relationships between gene expression levels, to preserve the joint correlational structure. Multivariate techniques will be more successful at this task than univariate techniques, but most multivariate techniques are computationally impractical for large numbers of variables. Thus, an important question is to identify those univariate techniques which are able to approximate well the performance of the multivariate ones.

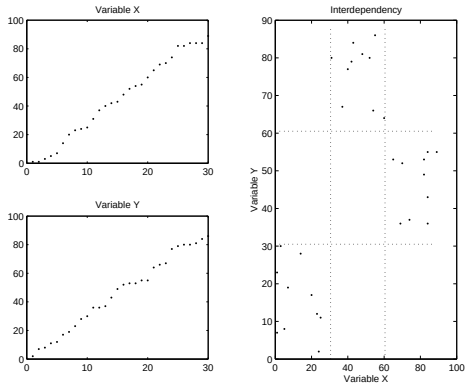


Figure A.1: Dependency between two features

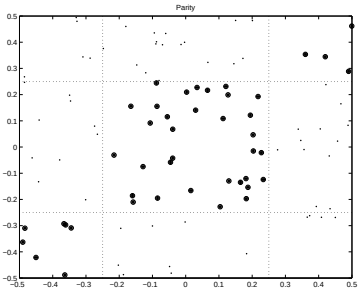


Figure A.2: Dependency among three features

A.1.1 Top-down versus Bottom-up Iteration

To facilitate discussion in the later sections, we first pause to describe two techniques of iteration which are used to achieve the user-defined number of discretization levels. Several of the discretization techniques detailed in Section A.2 will make use of these two alternatives, together with an optimization criterion specific to the discretization technique.

Top-Down: Binarization (BINARY)

Several of the methods considered are top-down procedures in that they begin with all data for a feature in a single bin and proceed by considering the best binary split within each bin implied by the previous iteration. The method then commits to the best among the candidate binary splits and the whole procedure iterates until the desired number of bins is reached. This method is greedy in that splits, once made, are never reconsidered. Best, both within and among bins, is determined by optimizing a particular criterion specified by the discretization method. Discretization methods considered in our comparisons which use this method are typically denoted by the suffix **Binary**.

Figure A.3 illustrates the process for a target of four bins. The data for the feature is sorted along the numberline and the solid rectangles represent the committed bin boundaries, hollow rectangles represent candidate binary splits. In Iteration 1, all data is assigned to a single bin. The best binary split of this interval is then found by optimizing the specified discretization criterion over all possible

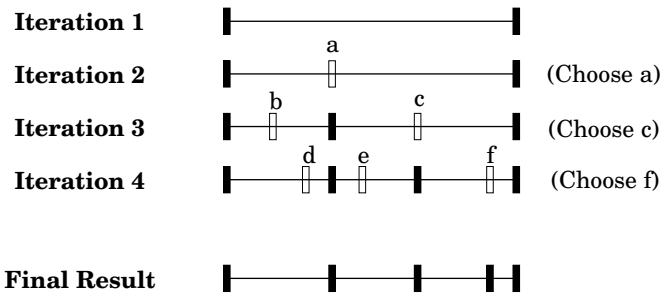


Figure A.3: Illustration of Binarization

binary splitpoints within the interval. With only one bin to consider in Iteration 2, the splitpoint a is chosen automatically. In the same manner, Iteration 3 considers each bin implied by the splitpoint a of the previous iteration, finding the optimal binary cutpoint within each bin, assuming data not assigned to the currently considered bin remains discretized as per the bin assignments of the previous iteration. Splitpoints b and c are found to optimize the criterion in their respective bins. Note that depending on the specific implementation of BINARY, the algorithm could commit to both b and c and proceed to refine each of their respective partitions. However, if the user is required to specify the granularity of the discretization, it is desirable to commit to a single split at a time to ensure that the best splits are made first. Thus, Iteration 4 proceeds by committing to the best among the binary candidates, namely cutpoint c, and considers each of the three resulting bins. The line labelled **Final Result** chooses f as the best among the three optimizing binary cutpoints d, e, and f offered by Iteration 4.

Bottom-Up: Discretization Level Coalescence (DLC)

Discretization Level Coalescence (DLC) [Har01] is a bottom-up procedure used for several of the methods whereby an initial fine-grained discretization of feature data is simplified by repeatedly coalescing two neighboring bins in each iteration while optimizing the criterion specific to the discretization technique. Discretization methods considered in our comparisons which use this method are typically denoted by the suffix **DLC**, and the user must specify both the target number of bins as well as the number of bins for the initial discretization.

The procedure is illustrated in Figure A.4 for ten initial discretization levels appearing as ordered leaves of the dendrogram. Starting from the singleton leaves, all neighboring pairwise combinations are considered for merging, a decision optimized according to a given discretization criterion. For example, the first merge considers the pairs (1,2), (2,3), (3,4), etc. and calculates the change in value of the discretization criterion. In the example, initial levels 1 and 2 are chosen first for coalescence; the resulting cluster is then merged with initial level 3, followed by a merge of level 9 with level 10, and so on. Discretization at any granularity can then be extracted from the resulting dendrogram. For example, for a three-bin discretization, the final discretization level coalescence would be {1-5},{6-7},{8-10}.

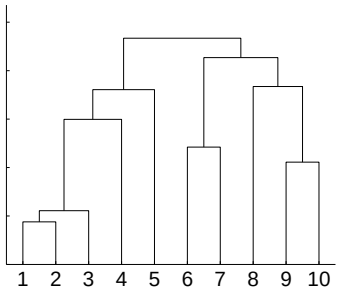


Figure A.4: Illustration of Discretization Level Coalescence

A.2 Discretization Methods

The topic of discretization is well studied in the supervised learning community though in contrast, the topic of discretization in unsupervised contexts remains largely unexplored. The paper by Bay [Bay01] details many approaches to unsupervised discretization but provides a comparison devised on classification tasks only between his proposed algorithm and an entropy-based supervised learning algorithm.

We provide a review of many unsupervised discretization previously referred to in the literature, as well as introduce several novel methods which were motivated by our task for learning models of gene interaction from expression data. To our knowledge, the comparison provided here represents the first and only large scale study of unsupervised discretization methods.

In all methods, we assign each of the original datapoints to a discrete value or *bin*, where the number of bins is defined by the user. We use the terms *splitpoint, threshold*, or *bin boundary* to refer to the largest real-valued number from the original data assigned to a particular bin. Thus, when the original datapoints are viewed as a sorted vector and labelled by their bin assignment, the bin boundary denotes the values up to and including the value that separates points assigned to one bin from those assigned to the adjacent bin. Often it is more convenient to speak of bin boundaries as indices into the vector of sorted values. The manner of usage as original data values or indices into the vector of sorted original values should be clear from context.

A.2.1 Method 1: Equal Frequency Interval (EqFreqInt)

The first discretization method, *Equal Frequency Interval (EqFreqInt)*, discretizes by quantile: each bin contains (roughly) the same number of data instances[Bay01, Har01, DKS95, RRS96]. Indices represent the bin boundaries, which are found by the formula:

$$\text{boundaryIndex}_i = \text{floor}(i * M)/N)$$

for M data instances and N bins, where $i = 1 \ldots N$. EqFreqInt is one of the most cited unsupervised discretization methods, owing in part to its ease of calculation. Also, according to [Bay01], "equal frequency partitioning maximizes the entropy for a fixed number of intervals and under Srikant and

Agarwal's *partial completeness measure* [SA96] minimizes the amount of information lost. Using the terms from Section A.1, we can characterize EqFreqInt as a a user-defined granularity, global, univariate discretization method.

A.2.2 Method 2: Equal Interval Width (EqIntWid)

The second discretization method, *Equal Interval Width (EqIntWid)*, discretizes by dividing the range of data into equal size intervals and assigning the data to bins according to the interval in which the data lie [Har01, LW00, DKS95, RRS96]. Unlike EqFreqInt where the bin assignment of a datapoint depends only on the ordering of the original values, the bin assignment under EqIntWid also relies on the relative spacing of the values. A potential disadvantage, however, is that some bins may be heavily populated while other bins may have few (or no) values. Like EqFreqInt, discretization by EqIntWid is easy to calculate, making it another popular choice for unsupervised discretization. The method is also a user-defined granularity, global, univariate discretization method.

A.2.3 Method 3: Standard Deviation (Std)

The third discretization method, *Standard Deviation (Std)*, discretizes by computing the mean and standard deviation of the original data and assigning the data to bins depending on how many standard deviations the data falls from the mean. This method assumes an odd number of bins – for example, to divide the data into three bins for data with mean m and standard deviation s, the bin boundaries would be placed at $[m - s, m, m + s]$. Note that like EqIntWid, one or more bins might not be populated. We include this technique since it is common for biologists to think of gene expression levels in terms of standard deviations away from the mean and this standard deviation technique was used previously for discretizing gene expression data [YTC02]. This technique is a user-defined granularity, global, univariate discretization method.

A.2.4 Method 4: Fold Change (FC)

The fourth discretization method, *Fold Change (FC)*, discretizes in a manner similar to Std but creates bin boundaries based on fold-change. This measure is only applicable to gene expression data measured as the (logarithm base-2) ratio of the gene expression level in an experimental sample versus its level in a control sample. Assuming the values are $\log_2$ ratio data, bin boundaries are specified at multiples of 0.5, which translates to intervals of $\sqrt{2}$-fold change. Again, some bins might not be populated. We include this technique since fold-change of expression is perhaps the most natural concept for biologist to explain expression data, and because the technique was used previously [FLNP00] for gene expression data. This technique is a user-defined granularity, global, univariate discretization method.

A.2.5 Method 5: Nearest Neighbor (NN)

The fifth discretization method, *Nearest Neighbor (NN)*, discretizes data by choosing bin boundaries that maximize between-neighbor gaps in the sorted data vector. This is akin to unsupervised clustering, thus any clustering algorithm could be used [PREF01]. In our implementation, we sort the values of a single attribute and compute the *Euclidean distance* between points. With these distances, we then perform hierarchical clustering using Ward's linkage method [War63], a technique which minimizes within-cluster variance with respect to between-cluster variance, thereby clustering a level with its nearest neighbor. The criterion optimized for this combination of Euclidean distance and Ward's linkage is similar to that optimized using K-means clustering.[1] The resulting dendrogram is cut to yield the specified number of bins. Since the underlying mechanism is hierarchical clustering, this method can be characterized as a user-defined granularity, *bottom-up*, global, univariate discretization method.

A.2.6 Method 6: Monothetic Contrast Criterion (MCC)

The sixth discretization method, *Monothetic Contrast Criterion (MCC)*, uses the bottom-up iteration technique BINARY, as described in A.1.1, in conjunction with a criterion for evaluating binary cutpoints that favor those producing the most contrasted partitions. The Monothetic Contrast Criterion [dM93] maximizes the Euclidean distance among values assigned to different partitions while minimizing distance between values within a single partition. For a given vector of data, a binary cutpoint dividing the data into two partitions, C_i and C_j, is evaluated according to the formula:

$$\mathrm{MCC}(C_i, C_j) = \frac{|C_i| \cdot |C_j|}{|C_i \cup C_j|} \cdot (m_i - m_j)^2$$

where m_i is the mean of the values assigned to partition C_i and $|C_i|$ denotes the number of values assigned to partition C_i. Since each cutpoint evaluation only considers points assigned to the two partitions, this procedure can be characterized as a user-specified granularity, bottom-up, *local*, univariate discretization technique.

A.2.7 Methods 7 and 8: Total Mutual Information (TMIDLC)

The seventh and eighth methods, *Total Mutual Information (TMIDLC_I and TMIDLC_Q)*, use the top-down iteration technique DLC, as described in A.1.1, in conjunction with a criterion for evaluating coalescence based on Total Mutual Information [Har01]. The I/Q distinction refers to the type of initial, fine-grained discretization used in DLC, where TMIDLC_I initially uses interval discretization (as termed by [Har01], also known as EqIntWid), and TMIDLC_Q initially uses quantile discretization (as termed by [Har01], also known as EqFreqInt).

[1] This statement was made by Thomas Hofmann PhD `thomas.hofmann@ipsi.fraunhofer.de` in personal communication, based on his PhD research and general knowledge of the area.

At each iteration of DLC, given that all variables have been previously discretized into n discretization levels (or bins), all adjacent pairs are considered as candidates for coalescence by evaluating the Total Mutual Information score for each resulting merge. Total Mutual Information at level n is defined as the sum of the pairwise mutual information between all pairs of variables:

$$\mathrm{TMI}(n) = \sum_{x,y,x \neq y} \mathrm{MI}(x,y).$$

Mutual information is calculated from *entropy* and *cross-entropy*. Entropy for a variable x that can assume the values x_i is calculated as

$$H(x) = -\sum_i p_i \cdot \log_2(p_i)$$

where p_i is the probability of x_i occurring. These are typically computed as the frequency of x_i occurring. By convention, terms where the probability is zero (*i.e.*, $p_i \cdot \log_2(p_i) = 0 \cdot \infty$), are set to zero which effectively removes them from consideration. Cross-entropy is defined similarly as

$$H(x,y) = -\sum_{i,j} p_{ij} \log_2(p_{ij})$$

where p_{ij} is the probability of x_i co-occurring with y_j. Mutual information is then given as

$$MI(x,y) = H(x) + H(y) - H(x,y).$$

The intuition for TMIDLC is to retain as much information as possible between the variables during each step of the coalescence algorithm. This is our first example of a multivariate, or dynamic, discretization procedure as it considers the pairwise effect of discretization. This is also our first example of a top-down procedure. Note that the mutual information is only calculated on the values in adjacent intervals, making this a local procedure. Also note that although the TMI score can be plotted and used to guide the choice for the optimal number of bins by identifying the "knee" in the curve, to make it comparable to the other techniques, we simply iterate DLC until the user-specified number of bins is achieved. Our TMIDLC_I/Q procedures are thus characterized as user-specified granularity, *top-down, local, multivariate* discretization techniques.

A.2.8 Methods 9 and 10: Relative Unsupervised Discretization (RUDE)

The ninth and tenth methods, *Relative Unsupervised Discretization (RUDE_F and RUDE_C)*, use a technique called *mutual structure projection* to determine bin boundaries [LW00]. When discretizing a particular attribute (or *target*), the values of the other attributes are *projected* onto the values of the target attribute. Considering each of the other attributes as a source in turn, a candidate splitpoints are found which correlate strongly with changes in the source attribute's value distribution. The set of candidate splitpoints are collected for all source attributes and postprocessed to yield the final discretization. An overview of the algorithm is given as

1. **Preprocessing:** Discretize (via some unsupervised method) all source attributes that are continuous.

2. **Structure Projection:** Project the structure of each source attribute a onto the target attribute t:

   (a) **Filtering:** Filter the dataset by the different values of the (discretized) attribute a.

   (b) **Clustering:** For each such filtering criterion, perform a **clustering** procedure on (continuous) values of the target attribute and gather the splitpoints thereby created.

3. **Postprocessing:** Merge the splitpoints found.

Considering each step in more detail, we use EqIntWid for the initial discretization of the **Preprocessing** step. The **Structure Projection** step is illustrated in Figure A.5. Figure A.5 (a) shows the values for the continuous target attribute t along the bottom. The next few lines demonstrates the **Filtering** step where the same values of t are filtered by the values of the source attribute a, namely $a = 1, a = 2$, and $a = 3$. Note that the values for t are essentially uniformly distributed, yet distinct ranges become informative about the value of a. Any univariate discretization technique would likely return a relatively arbitrary discretization for t that does not reflect the distribution changes in a that are readily apparent from the filtering. Detecting the boundaries of such distribution changes is the goal of structure projection. The idea is to collect each such set of points where the distribution of a changes drastically, for all attributes a and then cluster the results, since the collection may contain duplicate or nearly identical values of t.

To perform the **Clustering** step, we deviate from the presentation in the original paper since the details of their algorithm are not fully outlined. The goal is to find values of t where there are marked changes in a. This can be viewed as a supervised discretization problem: using a as the class variable, find boundaries in the range of t that are predictive of a. We consider two popular supervised discretization methods and will describe them fully in Section A.2.8 and Section A.2.8 below; the variants of RUDE, named RUDE_F and RUDE_C), refer to which method was used in the Clustering step.

The goal of the **Merging** step is to control for an unnecessarily fine-grained discretization since the candidate bin boundaries are created independently for each source attribute in turn; the final collection may contain values nearly identical on the range of t. For the Merging step, we also deviate from the procedure given in the original paper. The Merging step of the paper relies on specification of a window-size parameter, merging splitpoints that fall within a window's width of one another. However, we found their procedure to perform poorly in practice since determining this window size is highly dependent on the distribution of the original data values. As an alternative, we cluster the collection of potential bin boundaries using hierarchical (complete linkage) clustering with Euclidean distance as the metric. The resulting dendrogram is cut to yield the user-specified number of bins. In the event that the original collection of potential bin boundaries has fewer than the user-specified number, the target attribute is simply discretized by EqIntWid instead.

Both variants of RUDE consider pairwise information while discretizing and both use the BINARY iterative procedure. Thus both variants can be characterized as user-specified granularity, top-down, local, multivariate discretization techniques.

RUDE_F: Clustering using Fayyad and Irani's ME-MDL

The first method uses the top-down procedure BINARY from Section A.1.1 and is attributed to *Fayyad and Irani* [FI93]. They combine a *minimal entropy heuristic* [Cat91] for evaluating candidate binary splits with the use of the *minimum description length principle* as a stopping criterion for the recursion. This algorithm is perhaps the most widely used supervised discretization method.

The minimal entropy heuristic uses class information entropy to evaluate candidate bin boundaries at each step in the BINARY procedure. Given a set of data instances C, and a partition boundary T for a feature which divides C into two sets C_i and C_j, the *class information entropy* $E(T;C)$ of the partition induced by T is given as

$$E(T;C) = \frac{|C_i|}{|C|}H(C_i) + \frac{|C_j|}{|C|}H(C_j)$$

where $H(C)$ is the same definition of entropy given in Section A.2.7. The boundary T_{min} which minimizes the class information entropy is chosen at each iteration of BINARY.

The minimum description length principle determines when to halt the recursion. Recursion continues until

$$Gain(T;C) < \frac{\log_2(N-1)}{N} + \frac{\Delta E(T;C)}{N}$$

where N is the number of instances in C,

$$Gain(T;C) = H(C) - E(T;C)$$

$$\Delta E(T;C) = \log_2(3^k - 2) - [k \cdot H(C) - k_i \cdot H(C_i) - k_j \cdot H(C_j)],$$

and k_i is the number of class labels represented in the set C_i.

RUDE_C: Clustering using CART

The second alternative used for the Clustering step also uses the top-down BINARY recursive partitioning algorithm and is based on *Classification and Regression Trees (CART)* [BFOS84]. For a target attribute t, a classification variable a, and a function $\mathcal{C}$ which represents the cost of misclassification, the algorithm recursively builds a tree; the root node contains all data instances and each interior node of the tree divides the set of instances by testing a predicate of the form $t < T$. Each node is assigned a predicted class label based on a function of $\mathcal{C}$ and the distribution of class labels for data instances at each node.

The splitting criterion at each node evaluates each potential splitpoint T according to the *Gini diversity index*, defined as

$$G(C) = 1 - \sum_{j=1}^{M} p_j^2$$

where C is a set of data instances, M is the number of values for the class variable and p_j is the percentage of samples in C for which the class variable is assigned the value j. Given a boundary splitpoint T, we define

$$\Delta G(T;C) = G(C) - |C_i| \cdot G(C_i) - |C_j| \cdot G(C_j)$$

where $|C_i|$ is the number of instances in the partition C_i. The boundary T_{max} which maximizes the Gini diversity index is chosen as the tree node value at each iteration of BINARY.

The procedure is applied repeatedly, growing a "maximal" tree where any further splitting is impossible either because the child nodes have only one instance each (or a user defined number each), the child nodes have a single class label in each or the two child nodes would have identical distributions of labels. Since the maximal tree likely overfits the data, the tree building stage is followed by a tree pruning stage, successively pruning leaves of the tree while trying to balance tree complexity against the cost of misclassification. In the implementation of CART used in our comparisons, impure nodes must have ten or more observations to be split and the cost of misclassification is $\mathcal{C}(i,j) = 1$ if $i \neq j$ and $\mathcal{C}(i,j) = 0$ otherwise.

Examples of the Clustering Techniques

Figure A.5 (b) and Figure A.5 (c) show the results of applying the two examples of supervised discretization methods, Fayyad and Irani's ME-MDL and CART, to the data from Figure A.5 (a). Bin boundaries are indicated by horizontal lines. Note that CART suggests three bin boundaries while ME-MDL suggests only two. In general, we found that Fayyad and Irani's procedure is more conservative in assigning bin boundaries than CART, tolerating more class label impurity in a given partition.

A.2.9 Method 11: Multivariate Discretization (MVD)

The eleventh method, *Multivariate Discretization (MVD)*, discretizes according to the rule that two regions of values of an attribute should only be in the same interval after discretization if the data instances falling in those regions have similar multivariate distributions, across all variables and combinations of variables [Bay01]. An overview of the algorithm is given as

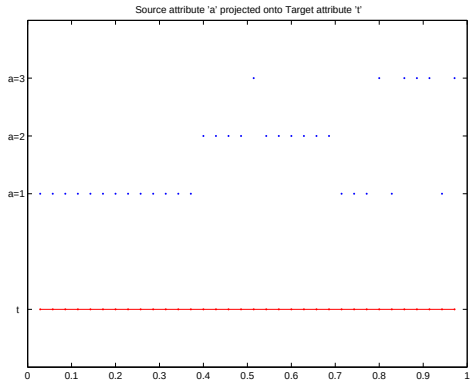
1. Finely partition all continuous attributes into n basic intervals using either EqIntWid or EqFreqInt.

2. Select an attribute A for discretization and select two adjacent bins C_i and C_j that do not have a known discretization boundary between them as candidates for merging. Let D_i and D_j be the set of data instances corresponding to those instances where $A = C_i$ or $A = C_j$, respectively.

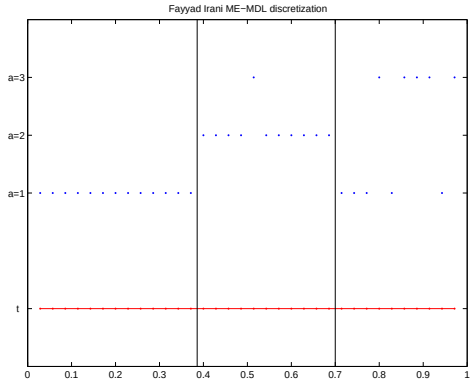
3. Test the null hypothesis H_0 that the distribution of instances in D_i is equal to the distribution in D_j, *i.e.*, $F_i = F_j$, against the alternative hypothesis $F_i \neq F_j$. If $F_i \approx F_j$, then merge C_i and C_j into a single interval. Otherwise place a discretization boundary between the two intervals.

4. If there are no eligible intervals, stop. Otherwise go to step 2.

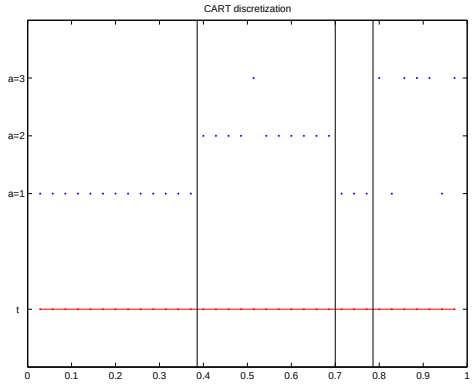
Note that to make MVD comparable to our other techniques which have a user-specified granularity, we modify the algorithm to iterate the procedure outlined above until the desired number



(a) Structure projection of source a on target t.



(b) Clustering via Fayyad and Irani's ME-MDL



(c) Clustering via CART

Figure A.5: Structure Projection

of bins is achieved. However, even at the end of iteration, there might be attributes with more bins that specified, as more definite boundaries might be asserted in step 3 than specified. In this case, bin boundaries are clustered by Euclidean distance and the two closest bin boundaries are merged. This process is iterated until the desired number of bins is reached.

There are a number of other details to specify for the algorithm. For step 1, we use EqIntWid per attribute, unless its use results in empty bins, in which case we use EqFreqInt for that attribute to ensure that all features begin the procedure with the same number of bins. For step 2, we cycle through the attributes in order, considering one merge per attribute. The candidate intervals are then chosen at random. Step 3 requires a multivariate test of differences and we use the STUCCO algorithm as in the original paper [Bay01, BP99]. STUCCO is motivated by the task of mining contrast-sets which tries to identify all conjunctions of attributes and their values whose distribution differs significantly across groups. We modify the STUCCO algorithm slightly since here our groups of instances are defined by the two candidate merging intervals for a particular attribute and the goal is then to find *any* combination of attribute-value pairs among the remaining variables whose distribution differs between the groups.

STUCCO looks for a conjunction of attribute-value pairs whose distribution is both statistically different between the two groups, as well as sufficiently large. If a *contrast set (cset)* is defined as a conjunction of attribute-value pairs, then let the *support*, $\text{support}(\mathsf{cset}, G)$, of a group G be the percentage of examples where the contrast-set is true. A *statistically significant* contrast-set is one where

$$P(\mathsf{cset}{=}\mathsf{True}|G_i) \neq P(\mathsf{cset}{=}\mathsf{True}|G_j)$$

which can be evaluated using a standard χ^2 test for independence as a significance level α, which tests the null hypothesis that the support for the contrast-set is the same across groups, *i.e.*, that contrast-set support is independent of group membership. A *sufficiently large* contrast-set is one such that

$$|\text{support}(\mathsf{cset}|G_i) - \text{support}(\mathsf{cset}|G_j)| \geq \mathsf{mindev}$$

where mindev is a user defined threshold specifying what the minimum deviation in support size must be in order for the two distributions to be considered different. The parameters of the algorithm are thus the significance level α and the minimum deviation mindev.

The search for statistically different and significantly large contrast-sets considers all conjunctions of attribute-value pairs until one is found. This search can be viewed as a tree, where the root is the empty conjunction, and each level of the search tree adds one more attribute-value pair to the conjunction on the previous level, specializing the node further, until at the leaves, there is a full assignment of values to all attributes. For two groups which have no qualifying contrast-sets, meaning the intervals will be merged in the top-level MVD algorithm, this requires exhaustively evaluating an exponential number of tests. For this reason, the original paper lists several strategies for pruning the search tree, based on the intuition that for a node in the search tree that fails the two tests outlined above, any further specialization of that node will also fail. We implemented all pruning strategies and refer the reader to the original paper for the specific details of those

strategies [BP99]. However, even with these strategies, MVD is an expensive procedure and for this reason is only applicable on a few of the datasets considered in our comparison of techniques.

All other examples of multivariate discretization techniques we consider are pairwise techniques; MVD is the only fully multivariate technique. Each test for merging considers only those data instances which fall in the two intervals considered for merging, making MVD a local algorithm. Thus, MVD is characterized as a user-specified granularity, bottom-up, local, multivariate discretization techniques.

A.2.10 Methods 12-16: Individual correlation of Discrete to Real (IDR)

The twelfth through sixteenth methods are variants based on the *Individual correlation of Discrete to Real (IDR)* criterion. We developed these novel techniques based on our experience in working with gene expression data, where calculating the correlation between pairs of gene expression profiles is integral to data analysis. For each individual attribute, these techniques attempts to preserve the Pearson correlation of the discretized version of the data with respect to the continuous, real-valued version of that attribute's data.

IDR and the Pearson correlation coefficient (Scale-To-Mean)

The Pearson sample correlation coefficient assumes that both samples are continuous, which makes its application problematic in this context. Discretization of continuous values involves a non-linear transformation of the original data scale: the distance between adjacent bin assignments does not reflect the distance between the original data values. Figure A.6 illustrates this phenomenon: the middle line in each plot shows the numberline with sorted data values for a continuous variable and the top line illustrates three bins of discretization, spaced one unit apart. The dashed lines between numberlines indicate bin boundaries. Note that the correspondence of one unit of measurement on the top line is stretched non-linearly and non-uniformly when mapped onto the range of the original values; in the top plot showing Discretization Method A, the first and second bins are drastically collapsed while the third bin is greatly stretched. For Discretization Method B, the first and third bin are stretched while the distance between bin boundaries demarking the second bin remains roughly equal to one unit of measurement.

To counteract this stretching effect when computing the Pearson correlation coefficient, we first transform the discrete vector into continuous space by replacing each point assigned to a particular bin by the mean value for all (continuous) data points assigned that bin.[2] The bottom line in each plot of Figure A.6 illustrates the resulting vector. Note that the distance between adjacent bins is no longer equal to one unit, rather it has been stretched to reflect the relative (average) distance of points in one bin to those in the adjacent bins. We refer to this procedure as *scale-to-mean*.

The severity of the distortion of scale depends both on the distribution of the continuous data as well as the discretization scheme. The goal of IDR is to assign bin boundaries that will distort

[2] Special thanks to Dr. David Laidlaw of Brown University for this suggestion.

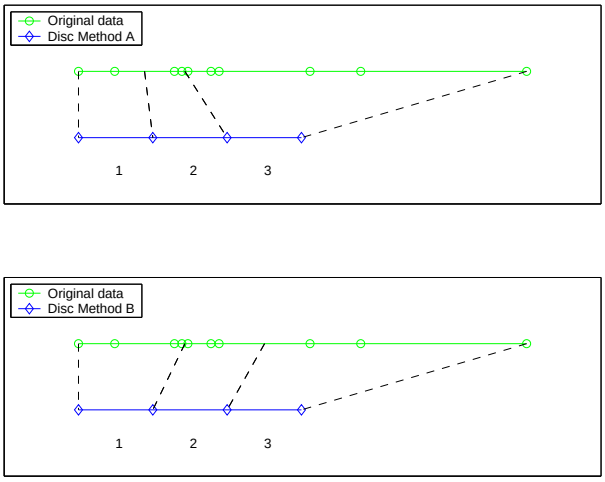


Figure A.6: Discretization results in non-linear mapping

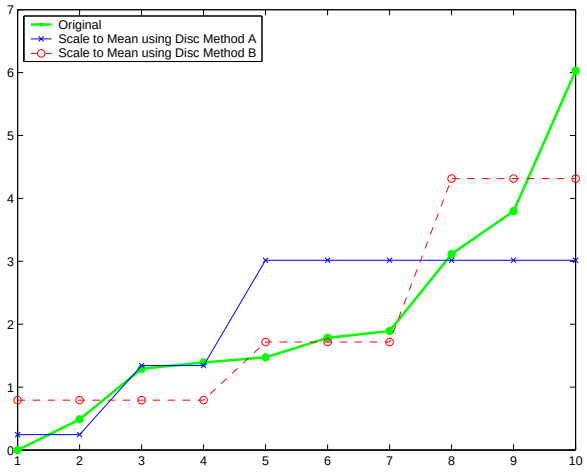


Figure A.7: Correlation of Discrete to Real

the scale in such a way as to preserve the shape of the curve when the data is viewed as a profile. The amount by which a particular discretization method preserves profile shape can be quantified by calculating the Pearson correlation coefficient between the original data vector and the resulting (scaled-to-mean) discrete vector, the criterion optimized by IDR.

Figure A.7 shows the original data along with the results of applying the scale-to-mean transform to the same discretizations (Method A and Method B) shown in Figure A.6. Visually, we can see that Method B more closely preserves the shape of the original data and this is reflected in the value of the correlation coefficient: the correlation between the continuous and discrete versions using Method A was 0.6823 while the correlation using Method B was 0.8852 (a value of 1.0 implies identity).

IDR Pseudocode

The pseudocode for the IDR discretization procedure is given in Algorithm 1. Essentially, each row t of the matrix R represents the data instance for variable t. The algorithm searches for the bin boundary assignments that maximizes the Pearson correlation coefficient between the data for t and the scaled-to-mean version implied by the bin boundary assignment. Searching for the correlation coefficient which maximizes over all possible assignments of k bin boundaries for a vector of length n is an $\mathcal{O}(\binom{n}{k})$ problem. Thus, we propose several variants of IDR which differ in their approach to this search problem: each provides a stopping criterion for the **while** loop as well as the GenerateCandidateDiscreteVector procedure. We consider each variant in turn in the following sections.

Algorithm 1 Pseudocode for IDR Discretization

Input: R, an m by n data matrix of real numbers
Input: N, the target number of bins
Output: D, an m by n data matrix of integers
  for each row t of the matrix R **do**
    Initialize $MaxDiscVec = \{\}$ and $MaxCorr = 0$;
    while some stopping criterion not met **do**
      $c =$ GenerateCandidateDiscreteVector(t, N);
      $d =$ ScaleToMean(c, t);
      $r =$ PearsonCorrelation(d, t);
      if $r > MaxCorr$ **then**
        $MaxDiscVec = c$;
        $MaxCorr = r$;
      end if
    end while
    Set the corresponding row of D to $MaxDiscVec$.
  end for

Method 12: Full version of IDR (IDRFull)

Method 12 implements the full search over all possible assignments of k bin boundaries for a vector of length n. The GenerateCandidateDiscreteVector procedure simply enumerates all possible assignments, while the stopping criterion checks that all possibilities have been enumerated. Considering the data vector for each variable t_i in turn, where $i = 1, \ldots, m$, yields an algorithm with complexity $\mathcal{O}(m \cdot \binom{n}{k})$. IDRFull is characterized as a user-specified granularity, global, univariate discretization technique.

Method 13: Restart version of IDR (IDRRestart)

Method 13 implements an approximation of the Full IDR version which is motivated by examining the landscape of the function being optimized. Figure A.8 depicts a two-dimensional representation of the objective function for a 3-bin problem. The vertical axis shows the decision for where to place the left boundary in a sorted vector of values, and the horizontal axis shows the decision for the

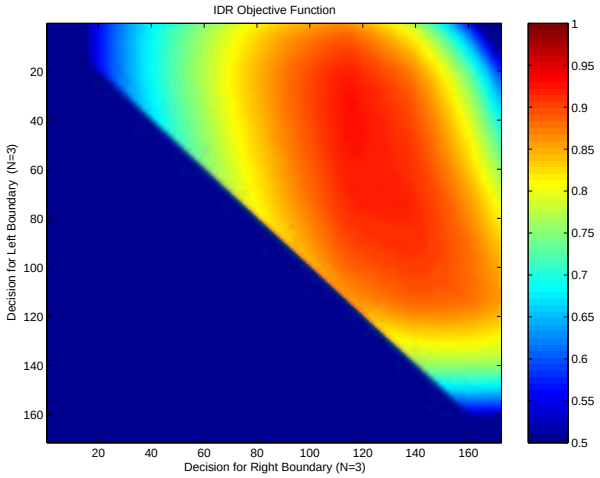


Figure A.8: Example of the IDR objective function landscape

right boundary (which is always greater than the left boundary, thus the upper triangular matrix of values). The color represents the correlation coefficient between the discrete version of the vector for the boundary decision $(left, right)$ and the continuous version. The correlation color scale, as shown by the colorbar to the right of the figure, has been adjusted to highlight the maximum of the function.

Further examples are shown in Figure A.9 and Figure A.10, where the two-dimensional colorplot is augmented with two three-dimensional projections of the same data. Beneath each three-dimensional curve is a contour plot showing the elevation of the function. Note that the views shown in the two-dimensional colorplots are the same as viewing the 3-D versions from underneath. These three examples highlight the three most typical profiles. The first in each figure contains a very narrow ridge near the boundaries of the function which results from data vectors that have outliers. The second in each figure shows a steep L-shaped ridge, while the third example in each figure shows a smoother surface.

It is easy to observe that the surface of the objective function for IDR is virtually convex and seemingly smooth on an example 3-bin problem which can be viewed in two dimensions. A natural question would be to ask whether the function is relatively smooth in higher dimensions. Figure A.11 shows an example for a 4-bin problem. The function on each of the planes in these examples does appear to exhibit the same smooth behavior. Observing examples such as this provides the motivation for the IDRRestart approximation of IDRFull. Beginning with an initial assignment of bin boundaries, the GenerateCandidateDiscreteVector procedure performs a simple gradient ascent on the objective function. Random *restarts* are used to avoid local maxima, hence the name IDRRestart.

The IDRRestart algorithm begins with an initial assignment of bin boundaries and calculates the Pearson correlation between the resulting discretized vector and the continuous vector. New candidate discrete vectors are found by examining all 'neighbors' of the current set of bin-boundaries – all combinations of adjusting each bin boundary one datapoint to the right or to the left in the

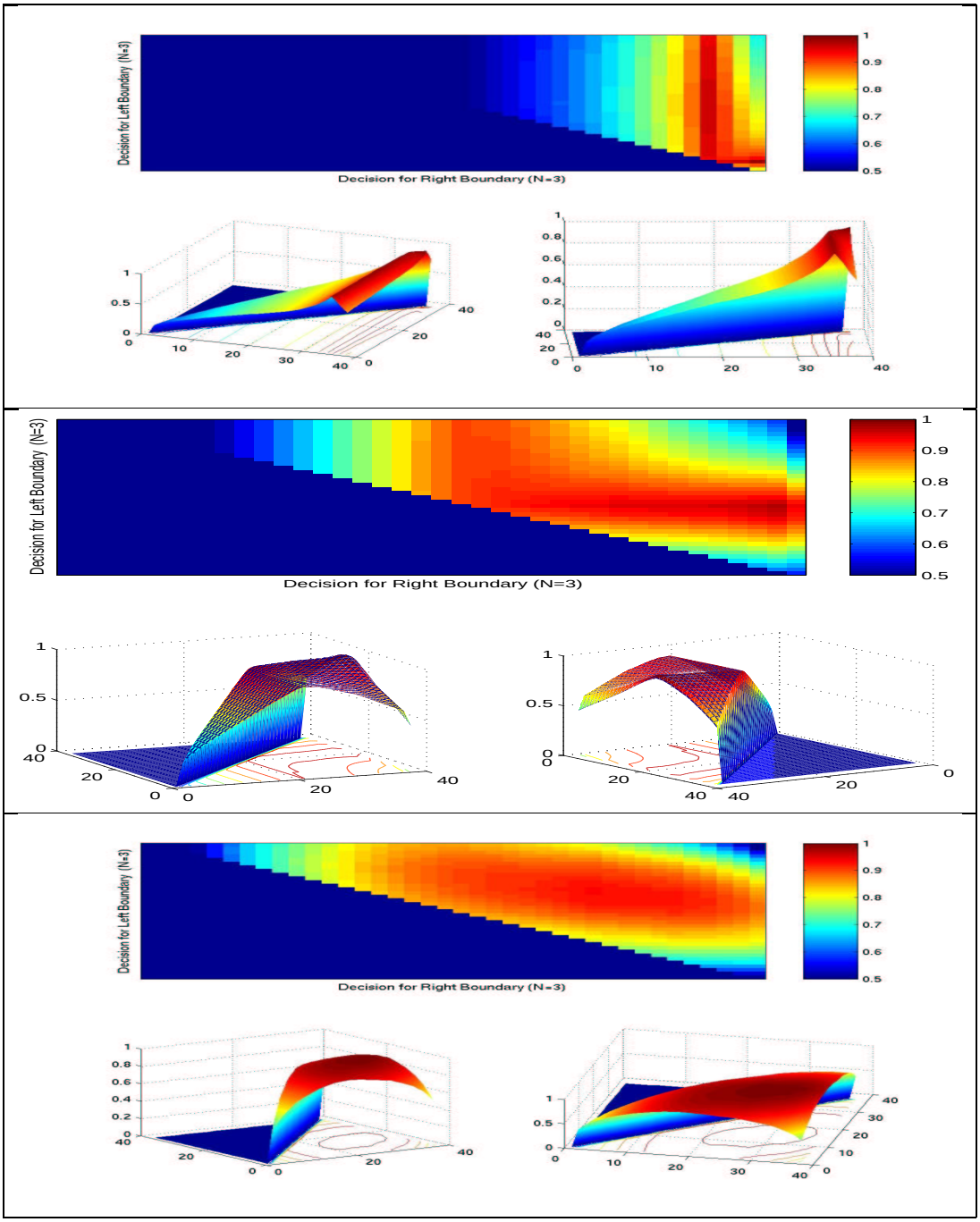


Figure A.9: Examples of the IDR objective function landscape (39 samples)

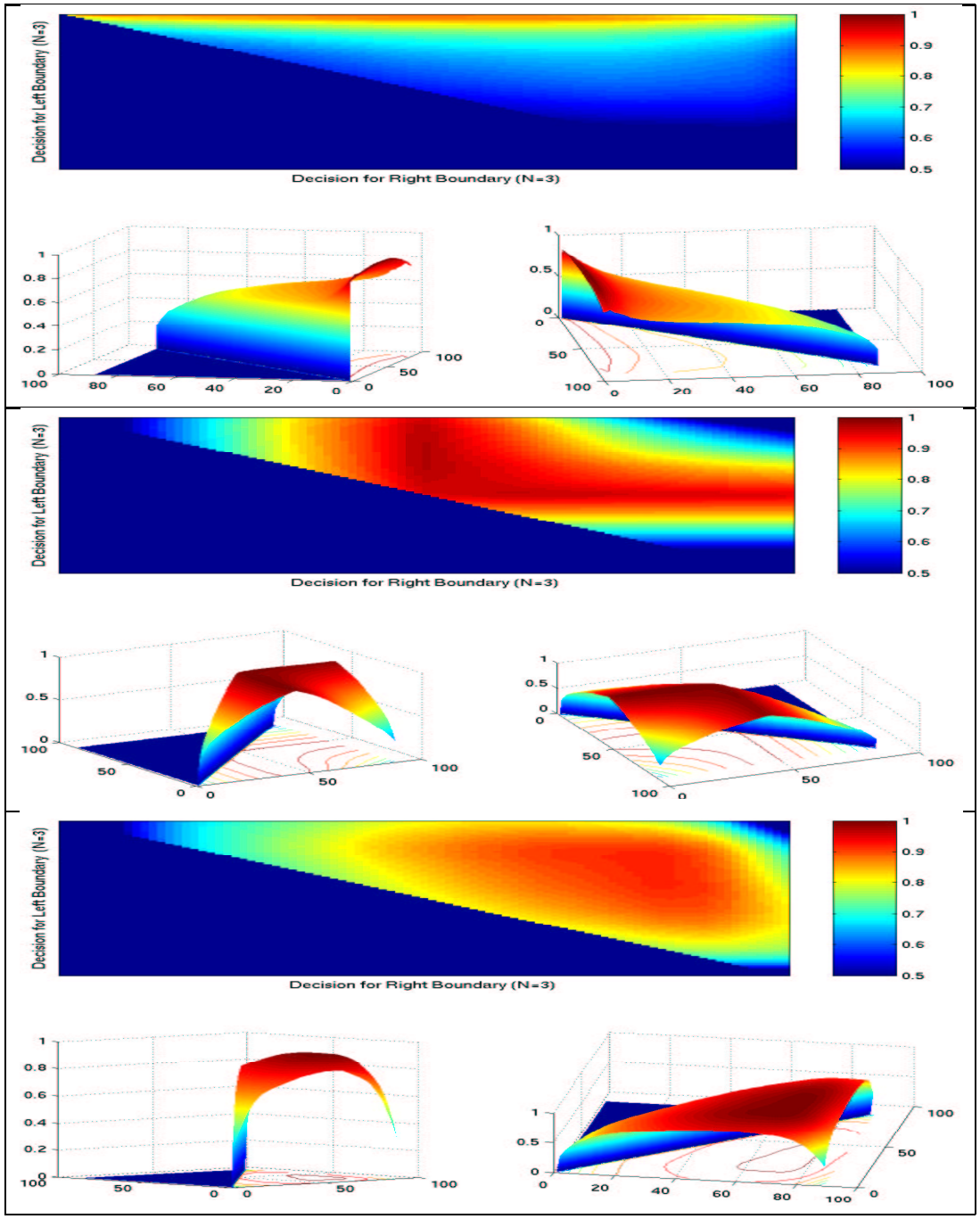


Figure A.10: Examples of the IDR objective function landscape (90 samples)

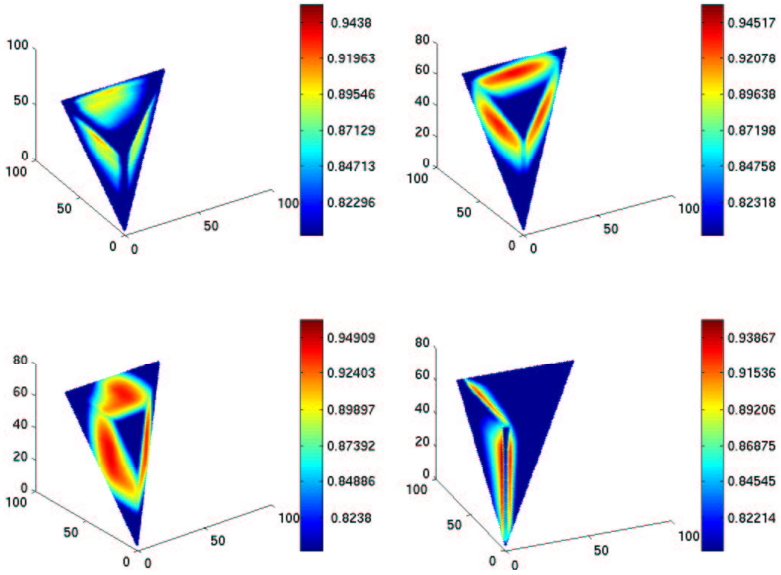


Figure A.11: IDR objective function for 4-bin examples.

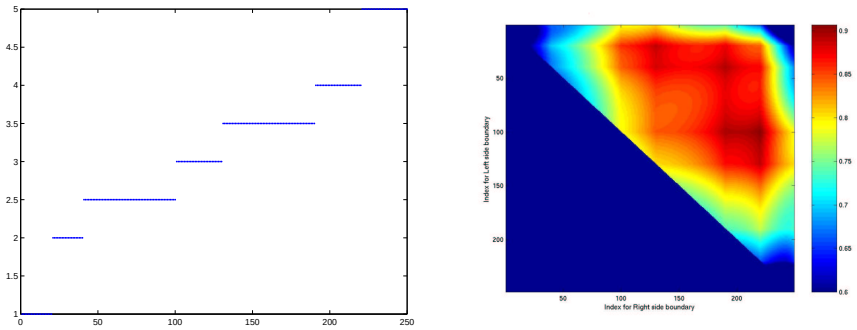


Figure A.12: IDR objective function for extreme example of local maxima

sorted data vector. For example, for a vector with the current 3-bin boundary assignment (4,7), the values (3,7), (5,7), (4,6) and (4,8) are evaluated. For each of the neighboring bin boundary assignments, the Pearson correlation coefficient is calculated and the maximum among the neighbors is chosen as the next point in the search. The stopping criterion is simply to continue until the current candidate is the maximum among its neighbors.

The landscape of the objective function can be quite complex and prone to local maxima, as can be seen in the contrived example shown in Figure A.12. The data clearly separates into seven distinct ranges, as shown in the plot of the sorted vector of values on the left. However, for the purpose of assigning the data to three bins, there are several nearly equivalent optima for choosing the 3-bin boundary assignments, depending upon which neighbor is clustered with each of the two longer stretches of values. These choices correspond to the three distinct peaks in the objective function plot on the right.

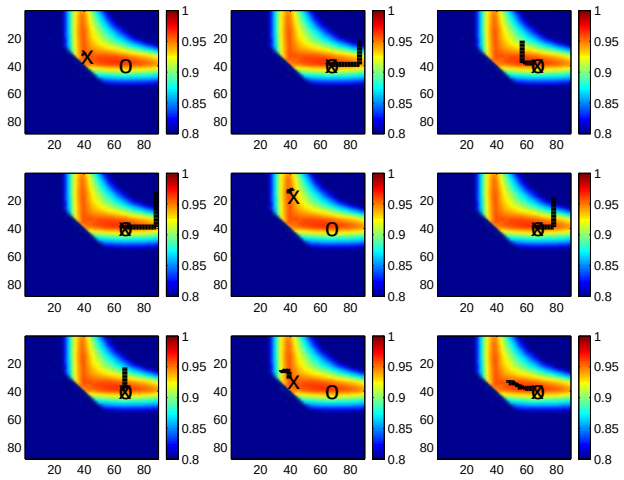


Figure A.13: Trace of Random Restarts for IDRRestart

To help alleviate the problem of local maxima, we use a number of random restarts, typically ten, and take the best resulting bin assignment among those restarts. The first of the ten restarts uses the boundaries for EqIntWid as the initial bin boundary assignment, unless that assignment is degenerate in which case random initial boundaries are chosen.

Figure A.13 shows an example of the search path for nine of ten random restarts superimposed upon a two dimensional color plot like those of Figures A.8 through A.10. The color scale is adjusted to highlight the maximum of the function. The black line in each plot illustrates the path of candidates chosen in each iteration, where the (local) maximum found is marked with an 'X'. The global maximum value of the function is marked with an 'O'. Note that in most of the nine plots, the search found the global maximum.

In contrast, Figure A.14 shows an example where IDRRestart fails to find the global maximum. As in the previous figures, each plot is a projection of the correlation function for a 3-bin problem for one of the ten restarts. The color scale has been adjusted to elucidate the peak values. Note that in this example, all of the restarts climb to a sub-optimal value. Two of the nine plots are enlarged in Figure A.15 to show clearly the arrival at a local maximum.

With the aid of the restarts, this approximation to IDRFull is highly effective, examining only a small fraction of the number of possible assignments considered by IDRFull, yet nearly always resulting in exactly the same bin-boundary assignments as the Full version. For example, for a variable with 173 samples, the Full version of IDR for a 5-bin problem examines 35,208,615 discrete vectors while the Restart version examines a *total* of 6279 for all ten restarts, making it an average of 628 per restart. The performance of IDRRestart will be quantified further in Section A.4. IDRRestart is characterized as a user-specified granularity, global, univariate discretization technique.

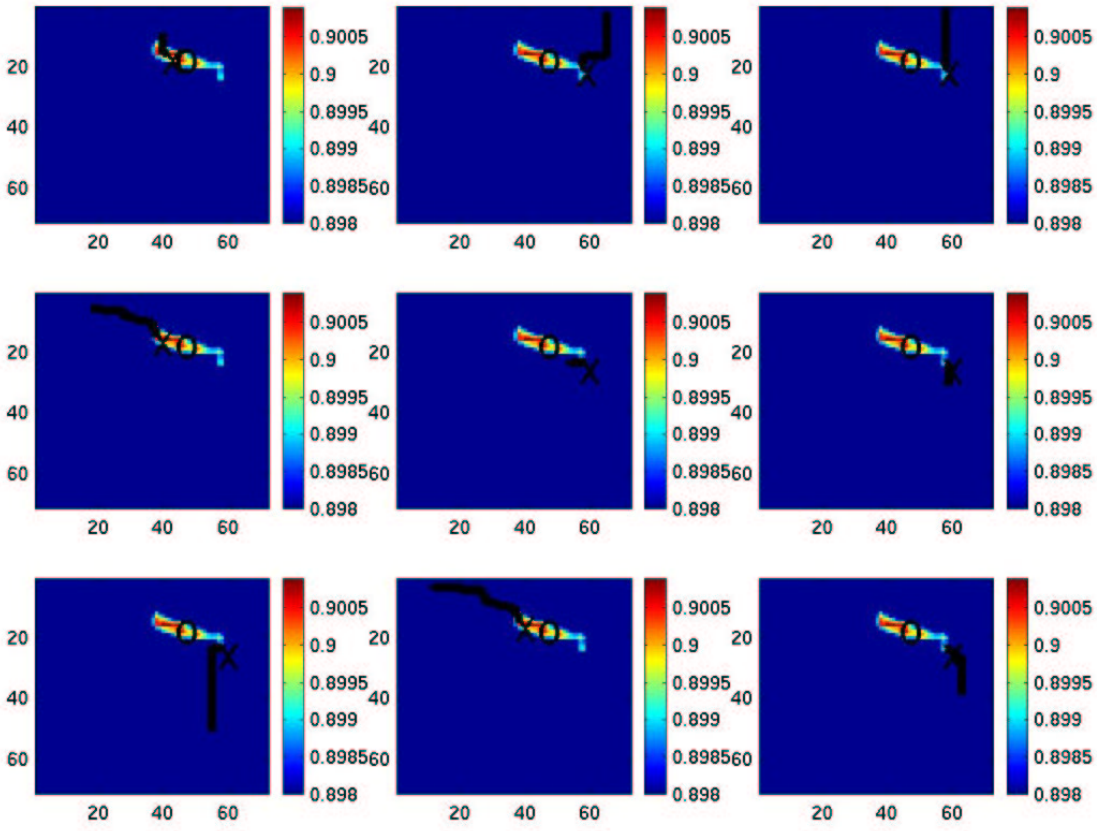


Figure A.14: Example where IDRRestart fails to find maximum

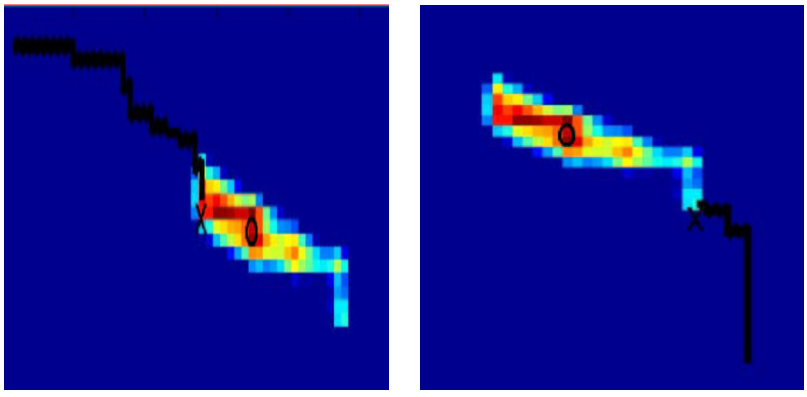


Figure A.15: Enlarged view of where IDRRestart fails to find maximum

Method 14: Split version of IDR (IDRSplit)

Method 14 implements another approximation to IDRFull which attempts to maximize the Pearson correlation in a piecewise fashion across the vector of values. Bin boundary limits are distributed across the length of the vector and candidate bin-boundary assignments from GenerateCandidate-DiscreteVector must respect those limits. For a vector of length n and a target number of bins N, the length of the vector is *split* into $n-1$ subproblems by placing $n-2$ bin boundary limits, hence the name IDRSplit.

Consider Figure A.16 where the bin boundary assignments for the Data Vector must initially respect the beginning and end of the vector, as signified by the dark vertical rectangles in the line at the top of the figure labelled **B** and **E** respectively. For a four bin problem, which must assign three bin-boundary divisions, two bin boundary limits are then distributed equidistant across the range of values as shown by the additional two vertical rectangles labelled **L1** and **L2** in the second line in the figure. These assert that while deciding the three bin-boundary divisions, the leftmost bin-boundary (**b1**) can range only between **B** and **L1**, the middle bin-boundary (**b2**) can range only between **L1** and **L2** and the rightmost bin-boundary (**b3**) can range only between **L2** and **E**. Given a bin boundary assignment to **b1, b2** and **b3**, as in the third line in figure, the resulting 4-bin discretized version is compared to the continuous version by calculating the correlation between them.

IDRSplit considers all possible combinations for the three bin boundary assignments which do not result in a degenerate solution (*i.e.*, having empty bins) and chooses the one that maximizes the correlation. By splitting the problem into subproblems, the savings are considerable because the possible values for each bin-boundary are limited. For example, for a variable with 173 samples, the Full version of IDR for a 5-bin problem examines 35,208,615 discrete vectors while the Split version examines 3,657,193. IDRSplit is characterized as a user-specified granularity, global, univariate discretization technique.

Intuitively, this procedure can be viewed as pinning down the endpoints of the subproblem and adjusting the splitpoint within that subproblem to maximize the correlation locally. We considered an alternative based on this intuition (*IDRLocal*) that performed these local optimizations, calculating the correlation only within each subproblem and then maximizing the sum of the correlation coefficients within each local subproblem. The idea was that the global correlation could be optimized by considering a combination of independent subproblems, where calculating correlation between the discrete vector and the real vector is confined to the datapoints within the subproblem. Experiments proved this not to be the case; this alternative performed poorly. The reason owes to the fact that the actual assignment of datapoints to discrete bins spans adjacent subproblems, which have numerical contributions in terms of the mean and variance as used in calculating correlation. Consider the discretized vector shown in Figure A.16. Note that the second bin (depicted by a circular tile pattern) stretches from **b1** to **b2**, spanning both the first and second subproblem. Calculating the correlation independently in the subproblem for **b1** fails to consider how many datapoints will also be assigned to bin 2 when deciding **b2** for the second subproblem and this can

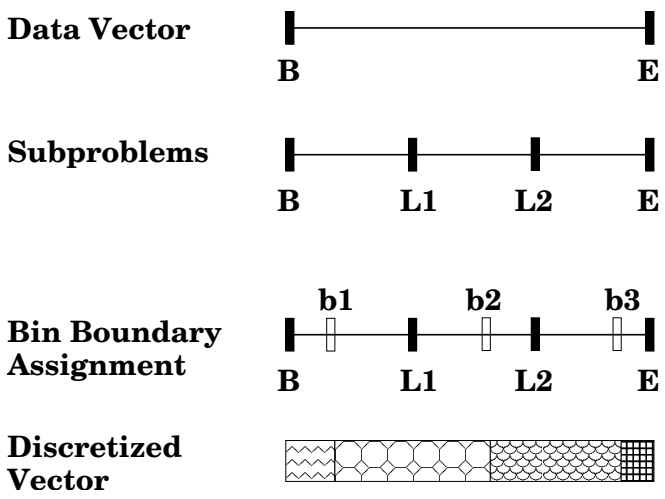


Figure A.16: Example for IDRSplit

have a dramatic effect on the difference between the (sum of) local correlations and the true, global correlation.

Method 15: Binary version of IDR (IDRBinary)

Method 15 implements an approximation to IDRFull that uses the top-down binarization procedure described in Section A.1.1. The GenerateCandidateDiscreteVector procedure considers all binary splitpoints within a given partition and the stopping criterion ensures that the target number of bins is reached. This procedure is thus characterized as a user-specified granularity, top-down, local, univariate discretization technique.

Note that IDRBinary is similar to the alternative IDRLocal discussed in the previous section since the correlation is calculated independently within each subproblem. Recall that for the BINARY procedure, when considering the binary cutpoint within a subproblem, the discretization of the remainder of the datapoints outside the subproblem retains the discretization decisions from the previous iteration. Adjacent partitions will thus not share values with their neighbors as demonstrated for IDRLocal and the correlation can be calculated from independent contributions. Also, in IDRBinary the bin boundary limits are not set to equidistant intervals a priori, as in IDRLocal; rather they are evolved from decisions made in the previous iterations.

Method 16: Discretization Level Coalescence version of IDR (IDRDLC)

Method 16 implements an approximation to IDRFull that uses the bottom-up discretization level coalescence procedure described in Section A.1.1. For the initial fine-grained discretization, we use EqIntWid. The GenerateCandidateDiscreteVector procedure implements the coalescence algorithm which, at each level of discretization, considers merging adjacent bins. The stopping criterion ensures that the target number of bins is reached by halting DLC at the appropriate level in the tree.

IDRDLC is characterized as a user-specified granularity, bottom-up, global, univariate discretization technique.

A.2.11 Methods 17-21: Spearman Individual correlation of Discrete to Real (SIDR)

In the above IDR variants, the Pearson correlation coefficient was calculated between data profiles. One alternative is to use instead the Spearman correlation which uses ranks instead of the continuous values. Thus, we offer variants of IDR based on the Spearman coefficient for comparison. However, at least in the gene expression applications motivating our investigation of discretization techniques, we are interested in preserving the structure of the original continuous distribution. Spearman rank correlation, by its very nature of being a non-parametric technique, discards precisely this information and is therefore counter to our purposes. Results using the Spearman correlation will therefore quantitate the loss of information as a result of using ranks.

The seventeenth through twenty-first methods are identical to the *Individual correlation of Discrete to Real (IDR)* methods discussed in the previous section except they use the *Spearman* rank correlation coefficient instead of the Pearson sample correlation coefficient in Algorithm 1. The variants are thus labelled **SIDRFull, SIDRRestart, SIDRSplit, SIDRBinary and SIDRDLC** and have the same characteristics as their IDR counterparts.

A.2.12 Methods 22-26: Pairwise correlation of Discrete to Real (PDR)

The twenty-second through twenty-sixth methods are variants based on the *Pairwise correlation of Discrete to Real (PDR)* criterion. These methods are novel methods which, like IDR, were motivated by our work with gene expression data. The PDR variants try to preserve the pattern of pairwise Pearson correlations between variables. Using the real-valued, continuous data for n attributes, a vector of pairwise Pearson correlation coefficients ρ_R is calculated between a target attribute t and the remaining $n-1$ attributes, where the R indicates that the real-valued version of t was used. Candidate discrete vectors for t are then compared against the continuous versions of the remaining $n-1$ attributes to create another vector of correlation coefficients ρ_D, where the D signifies that the discrete version of t was used. Calculating the correlation between a discrete vector and a continuous vector uses the same scale-to-mean technique as discussed for IDR in Section A.2.10.

The PDR objective function is to minimize the Euclidean distance between the two vectors of correlation coefficients, ρ_R and ρ_D. Note that this procedure does *not* try to maximize the total pairwise correlation, as is the objective in TMIDLC which tries to maximize the total pairwise mutual information. Trivially assigning all vectors to have the same discretization would result in maximum total pairwise correlation but would not preserve the original relationships between variables. Rather, PDR tries to reproduce the joint correlational structure in the discrete space by maximizing the correspondence between the vector of pairwise correlations using the continuous version of t and the vector of pairwise correlations using the discrete version of t. Thus, if two

attributes were poorly correlated in the original data, they will likely remain poorly correlated in the discrete data. The objective of the PDR method is distinct from the objective of the IDR method since the optimal bin assignment for preserving correlation to self (*i.e.*, IDR) might not be the optimal bin assignment for preserving the pairwise correlational structure of the original data.

PDR Pseudocode

The pseudocode for the PDR discretization procedure is given in Algorithm 2. Essentially, each row t of the matrix R represents the data instance for variable t. The algorithm searches for the bin boundary assignments that minimize the Euclidean distance between the two vectors of pairwise correlation coefficients, ρ_R and ρ_D. Searching for an assignment that minimizes over all possible assignments of k bin boundaries for a vector of length n and m attributes is an $\mathcal{O}(\binom{n}{k} \cdot m)$ problem. The same variants considered for IDR are presented for PDR in the following sections.

Algorithm 2 Pseudocode for PDR Discretization

Input: R, an m by n data matrix of real numbers
Input: N, the target number of bins
Output: D, an m by n data matrix of integers
  for each row t of the matrix R **do**
    R_t = { rows i of R $|i \neq t$}
    ρ_R = PearsonCorrelation(R_t, t);
    Initialize $MinDiscVec = \{\}$ and $MinDist = 0$;
    while some stopping criterion not met **do**
      c = GenerateCandidateDiscreteVector(t, N);
      d = ScaleToMean(c, t);
      ρ_D = PearsonCorrelation(R_t, d);
      $dist$ = EuclideanDist(ρ_R, ρ_D);
      if $dist < MinDist$ **then**
        $MinDiscVec = c$;
        $MinDist = dist$;
      end if
    end while
    Set the corresponding row of D to $MinDiscVec$.
  end for

Method 22: Full version of PDR (PDRFull)

Method 22 implements the full search over all possible assignments of k bin boundaries for a vector of length n. The GenerateCandidateDiscreteVector procedure simply enumerates all possible assignments, while the stopping criterion checks that all possibilities have been enumerated. Considering the data vector for each variable t_i in turn, where $i = 1, \ldots, m$, yields an algorithm with complexity $\mathcal{O}(m^2 \cdot \binom{n}{k})$. PDRFull is characterized as a user-specified granularity, global, multivariate discretization technique.

Method 23: Restart version of PDR (PDRRestart)

Method 23 implements an approximation of the Full PDR version which is motivated by examining the landscape of the objective function. Figure A.17 depicts a two-dimensional representation of the objective function for a 3-bin problem. The vertical axis shows the decision for where to place the left boundary in a sorted vector of values, and the horizontal axis shows the decision for the right boundary (which is always greater than the left boundary, thus the upper triangular matrix of values). To make the plots correspond to the IDR plots where red corresponds to desirable regions of the objective function, the color represents the *negative* Euclidean distance between the two correlation coefficient vectors. Thus optimal values (*i.e.,* Euclidean distance near zero) are shown in reds, while the burgundy region in the lower diagonal of the matrix represents the value zero. The colorbar, shown to the right, has been scaled to highlight the optimum of the function.

As can be seen in Figure A.17, the function is not as smooth as for the IDR examples (shown in Section A.2.10) yet there are still areas of maximal value. Figure A.18 shows a three dimensional version of the examples in Figure A.17 where the function surface and the contour map underneath the curve is considerably more complicated than the analogous plots of IDR shown previously in Figure A.9 and Figure A.10. Note that the shading on the plot is interpolated which incorrectly shades the 'wall' along the diagonal of the datamatrix; in actuality the values drop instantaneously from zero.

Figures A.19 through A.21 show the trace of the PDRRestart search for nine of ten restarts. The search path is depicted by a dark black line, ending in an 'X' which is the final result. Note that in contrast to the IDRRestart examples, the search path is much shorter (reaching a local maxima more quickly) and rarely finds the global maximum (marked with an 'O').

With a much more convoluted objective surface, PDRRestart is thus much more reliant on the initial starting value and would benefit from a large number of restarts to better explore the function landscape. Even with many restarts, the number of candidates explored is vastly reduced from the number considered by PDRFull. PDRRestart is characterized as a user-specified granularity, global, multivariate discretization technique

Method 24: Split version of PDR (PDRSplit)

Method 24 implements an approximation to PDRFull which splits the length of the data vector into distinct ranges that act as limits on the potential bin boundaries, as in IDRSplit (see Section A.2.10). PDRSplit is characterized as a user-specified granularity, global, multivariate discretization technique.

Method 25: Binary version of PDR (PDRBinary)

Method 25 implements an approximation to PDRFull that uses the top-down binarization procedure described in Section A.1.1. The GenerateCandidateDiscreteVector procedure considers all binary splitpoints within a given partition and the stopping criterion ensures that the target number of

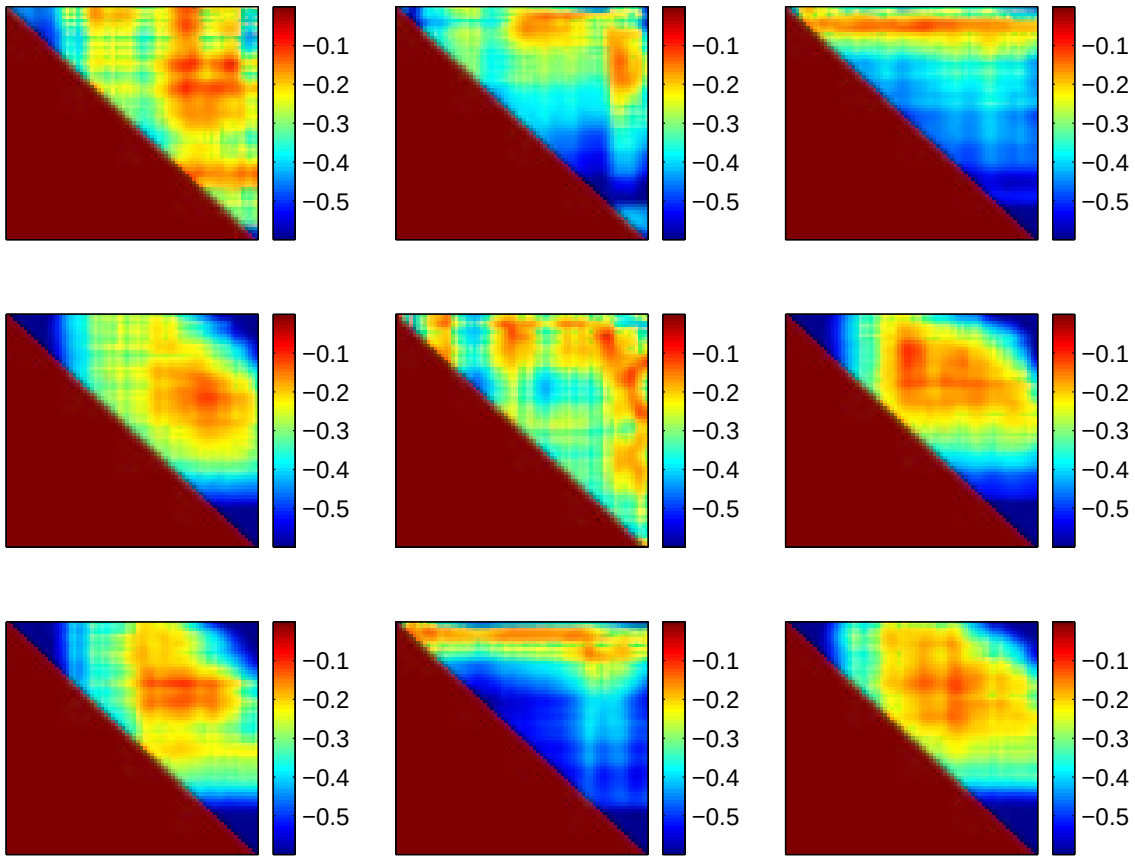


Figure A.17: Nine examples of the PDR objective function landscape

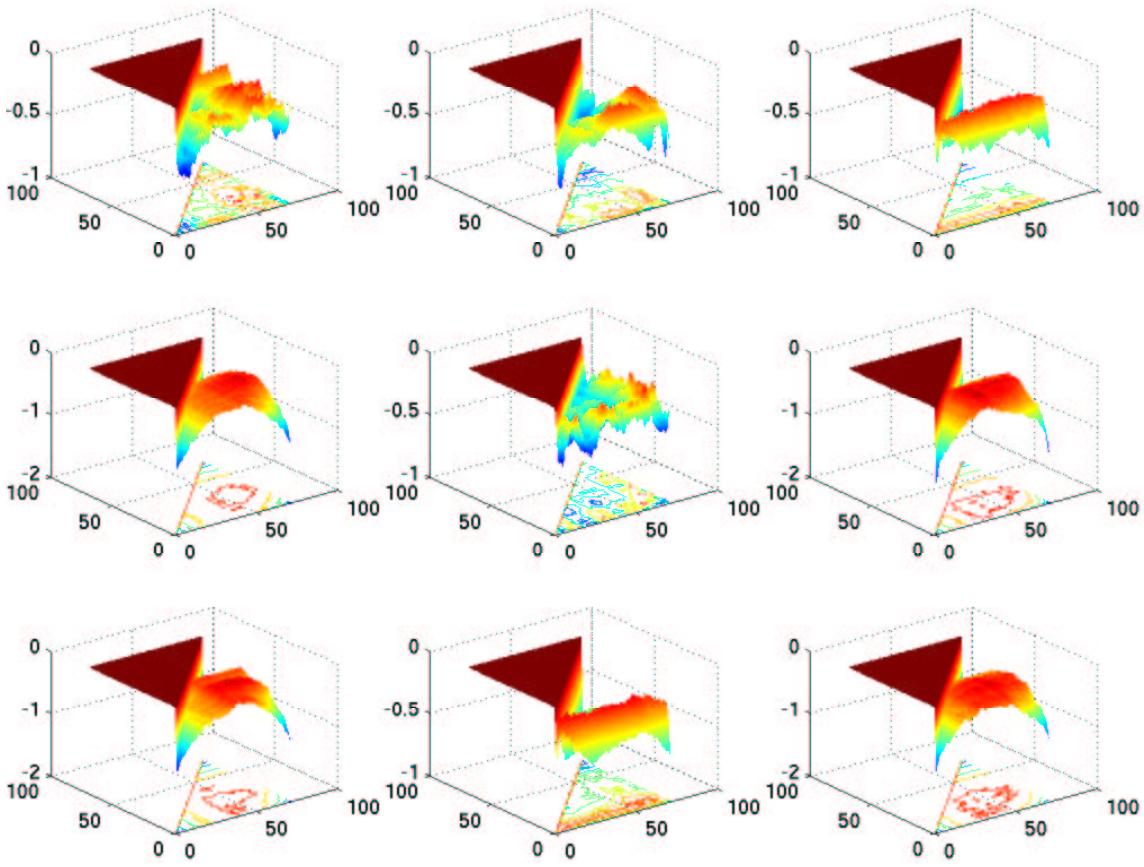


Figure A.18: Three dimensional views of PDR landscapes from Figure A.17.

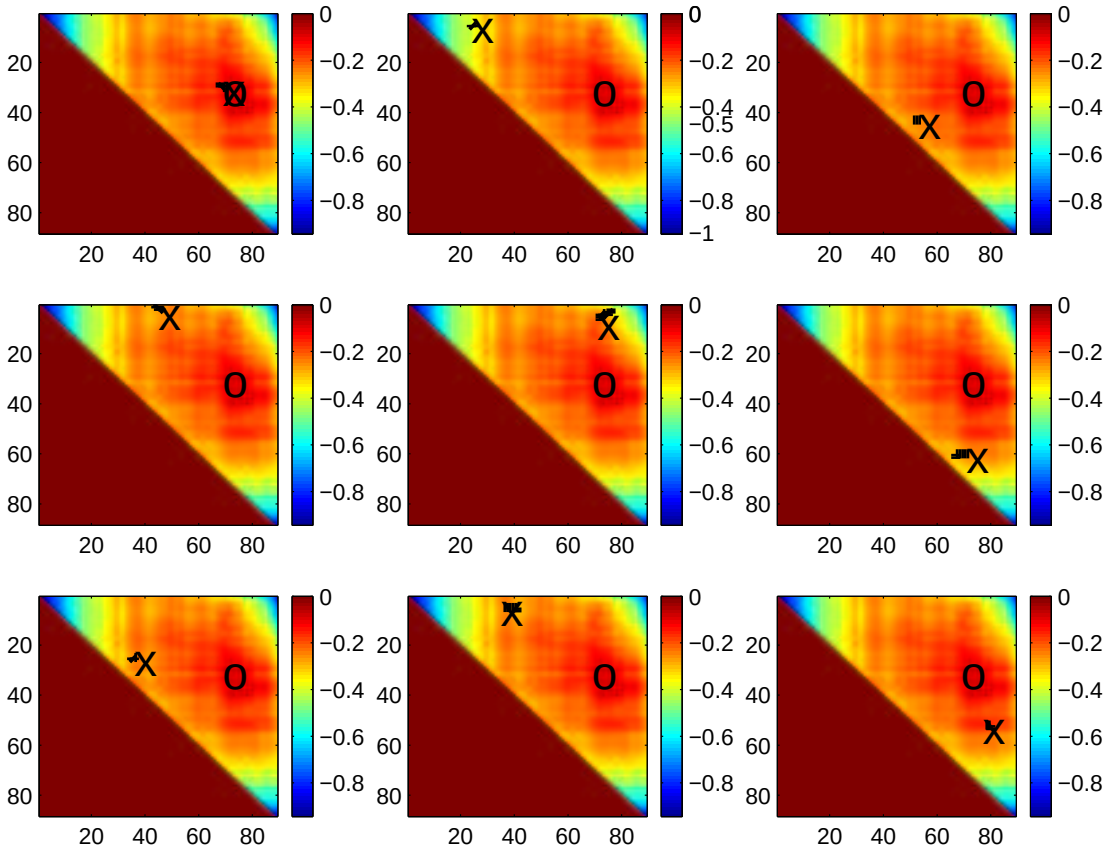


Figure A.19: PDRRestart finds the global maximum

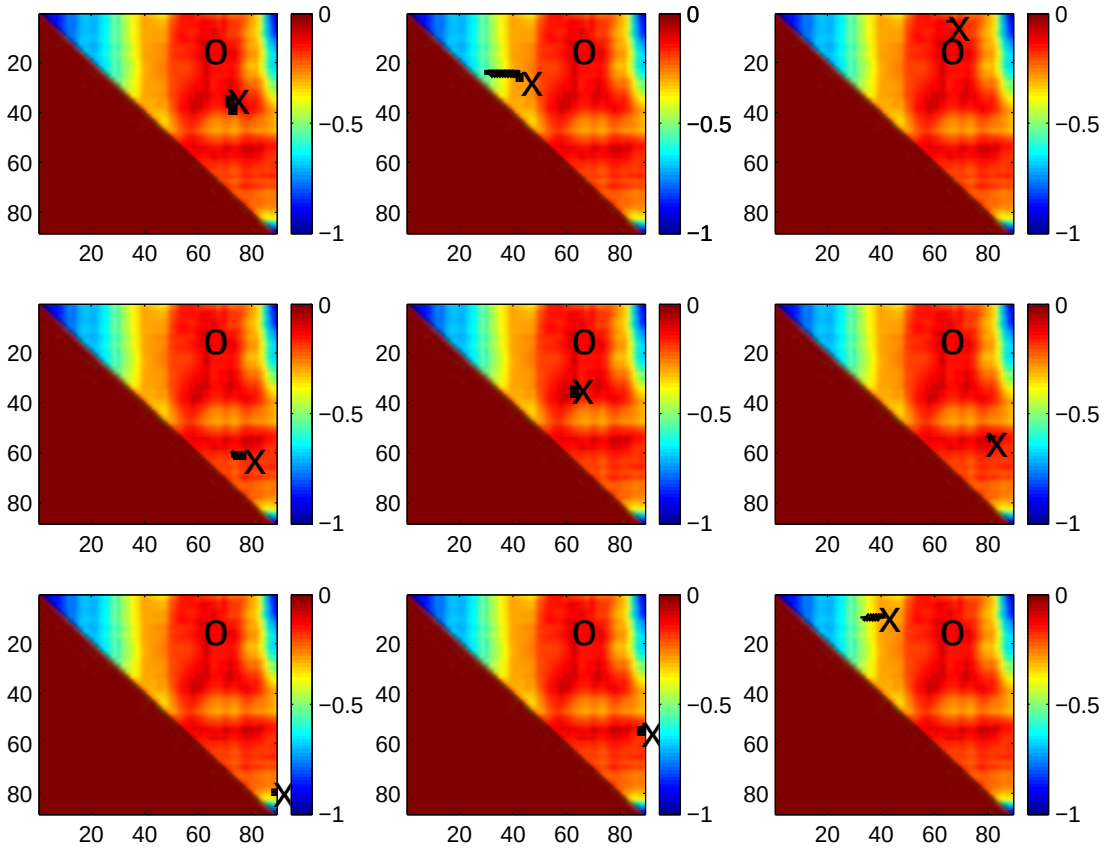


Figure A.20: PDRRestart fails to find the global maximum

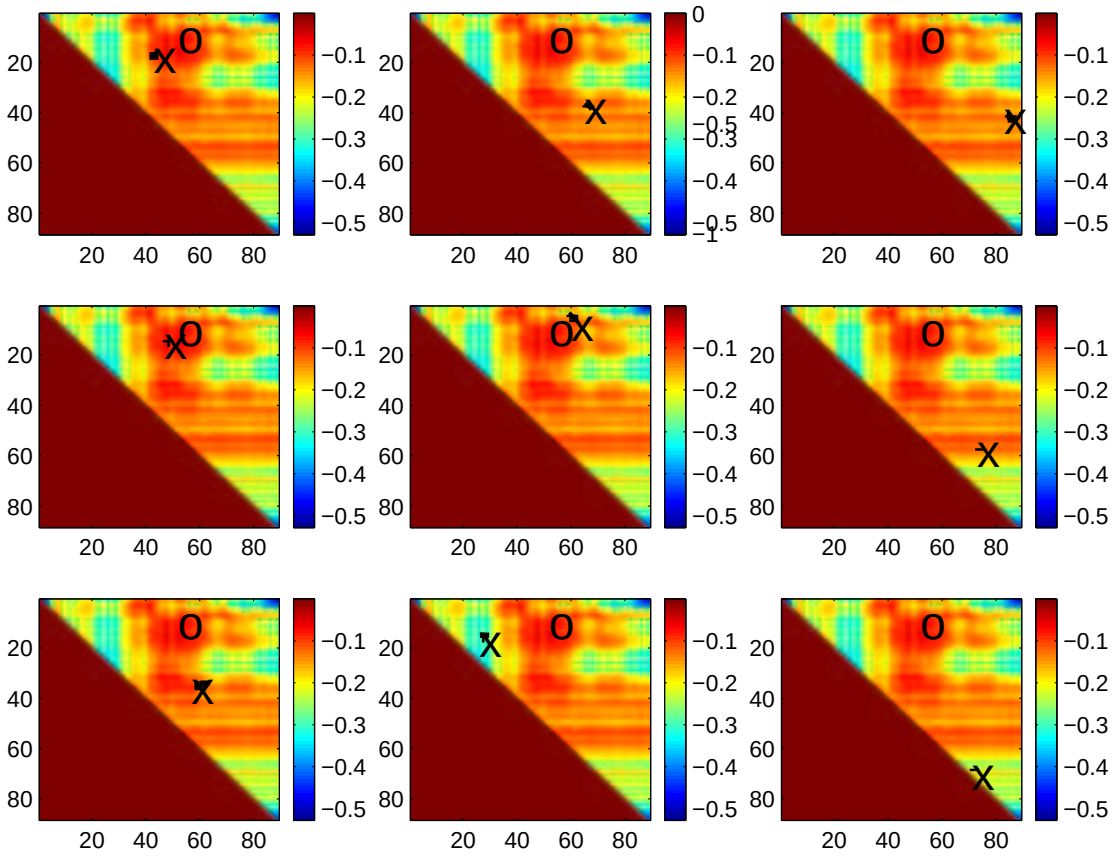


Figure A.21: PDRRestart fails to find the global maximum

bins is reached. This procedure is thus characterized as a user-specified granularity, top-down, local, multivariate discretization technique.

Method 26: Discretization Level Coalescence version of PDR (PDRDLC)

Method 26 implements an approximation to PDRFull that uses the bottom-up discretization level coalescence procedure described in Section A.1.1. For the initial fine-grained discretization, we use EqIntWid. The GenerateCandidateDiscreteVector procedure implements the coalescence algorithm which, at each level of discretization, considers merging adjacent bins. The stopping criterion ensures that the target number of bins is reached by halting DLC at the appropriate level in the tree. PDRDLC is characterized as a user-specified granularity, bottom-up, global, multivariate discretization technique.

A.2.13 Methods 27-31: Spearman Pairwise correlation of Discrete to Real (SPDR)

The twenty-seventh through thirty-first methods are identical to the *Pairwise correlation of Discrete to Real (PDR)* methods discussed in the previous section except they use the *Spearman* rank correlation coefficient instead of the Pearson sample correlation coefficient in Algorithm 2. The variants are thus labelled **SPDRFull, SPDRRestart, SPDRSplit, SPDRBinary and SPDRDLC** and have the same characteristics as their PDR counterparts.

A.3 Experimental Design

In order to compare the performance of the discretization techniques, we consider a variety of different datasets and a variety of evaluation measures. As stated earlier, this comparison is the only large scale study of unsupervised discretization methods available, to our knowledge.

The challenge of such a comparison study is in providing good evaluation measures. In the supervised context, most evaluations of performance utilize some function of the classification variable. The fidelity of the discretization is measured with respect to the ability to make the same classifications using the discretized data as when using the continuous data. However, unsupervised discretization, by definition, cannot optimize such a metric and a major contribution of this thesis is in characterizing what is meant by fidelity in the unsupervised context. In Section A.3.1, we present several criteria that attempt to measure the amount of information lost by discretization in terms of how well a technique can preserve the joint correlational structure among the variables.

In Section A.3.2, we discuss each of the datasets used in this study. We consider a range of datasets, varying not only the number of variables and number of samples, but the original application area for the dataset. Our examples include several gene expression datasets, which will be used in our task, as well as several from other machine learning contexts. In Section A.3.3, we list the parameter settings used in the various algorithms.

A.3.1 Performance Evaluation Measures

In the case of supervised discretization methods, evaluation measures are typically some function of the class variable. A successful discretization is one which allows the lowest misclassification rate. Even in the few published evaluations between unsupervised discretization methods, performance is usually calculated on datasets that are inherently classification problems: one or a few distinguished variables are some function of the other (continuous) variables [LW00, DKS95, Bay01]. Typically the unsupervised methods are compared against a supervised method on the same task, usually the Fayyad and Irani's procedure discussed in Section A.2.8.

Formulating an evaluation measure for unsupervised discretization methods often depends on the application. In the following sections, we consider a number of different applications for discretized data and formulate measures based on criterion suitable for those applications.

Mean Squared Error (MSE)

The first evaluation measure we consider derives from work in image compression. Discretization can be viewed as a type of *compression* where an (infinite) range of continuous values is mapped to single discrete value. One compression algorithm often used in Computer Graphics for image compression is *Vector Quantization (VQ)* whereby a finite set of vectors, called *codewords*, is selected and the continuous data is then mapped to an index in the set of codewords. The science of VQ algorithms is in defining the set of codewords. The most popular algorithm, named LBG [LBG80], learns a set of codewords from a training set of data by a KMeans-like clustering algorithm. The cluster centers then become the codewords.

The difference between the objective of VQ algorithms and discretization algorithms is that the codewords of the VQ algorithm reside in the same continuous space as the original data while in our case, the discretized vectors lie in discrete space. However, the study of compression algorithms suggests several measures for estimating the quality of an algorithm in preserving the original data.

Ideally the quality of image compression by a VQ algorithm is judged by humans who rank the amount of visual distortion in the resulting image. Aside from such subjective perceptual measures which have no analog in our application, the most popular measure for distortion by compression algorithms is *Mean Squared Error (MSE)*.

A dataset of n variables, each with m samples, can be viewed as the data matrix (*i.e.*, a two-dimensional image). Let R be the continuous valued dataset and D the (scaled-to-mean) version of the result of some discretization procedure on R. Then the Mean Squared Error is given as:

$$MSE(R, D) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (R(i, j) - D(i, j))^2.$$

Note that MSE is simply the average squared Euclidean Distance between R and D. A related measure used in signal processing is *Peak Signal to Noise Ratio (PSNR)* which is inversely proportional to MSE. We are interested in measuring relative performance of the various discretization schemes so use of either is equally appropriate.

We include the MSE measure in our comparisons due to its simplicity, familiarity and popularity. However, it is not an ideal measure for capturing the quality of our discretization procedures. MSE computes the average point by point difference between the continuous data matrix and a discretized version, treating each value (i, j) in the matrix as an independent sample. This is not true in our data: the columns of each row in the matrix are correlated since each row is data from one variable and the (row) data vectors for different variables are often correlated due to the biology. For this reason, we would also like measures which are functions of both the row vectors and the column vectors.

The measures discussed in the following two sections offer a compromise: they first transform the original datamatrix into a new dataspace via a function of both the rows and columns, then take the MSE in the transformed space.

Wavelet Distance (WAVD using Haar, Daubechies(4), Coiflet(4))

The wavelet distance measures considered in our comparisons also derive from an application in Computer Graphics: image querying. The objective is to search a large database of images for an image similar to a query image. The query is often a hand-sketched or low-resolution version of the original, just as our discretized data can be considered a lower-resolution version of the original data matrix. Since the query may be very distorted from the original, a point-by-point approach like MSE will not be as successful as a similarity metric which considers functions over the entire data matrix.

One popular approach to image querying is to decompose images using *wavelets* and compare the coefficients of the wavelet decompositions [JFS95]. Wavelets can hierarchically decompose functions, allowing any signal to be represented at increasingly coarser levels of details. Depending on the family of wavelets used in the decomposition, many of the wavelet coefficients become (small or) zero values, allowing a sparser representation of the original signal with little or no data loss. Wavelet families are characterized by whether the wavelet function provides an orthogonal basis (*i.e.,* non-redundant) or not, and the number of vanishing moments (*i.e.,* continuity and regularity conditions). In general, wavelets with more vanishing moments are well suited for smooth, oscillating signals while wavelets with fewer vanishing moments are better for signals with local singularities [LPIS00].

Since the interesting biology is characterized by outliers in our data, we want wavelets which compactly capture the irregularities. We make use of three wavelet families which are orthogonal and possess a small number of vanishing moments: *Haar*, *Daubechies* and *Coiflets*. Plots of the wavelet function are shown in Figure A.22. Note that the wavelet functions have a finite region where the function is non-zero (a phenomenon known as *compact support*) and the continuity of the function varies with each wavelet family shown.

Haar wavelets are the simplest example of an orthogonal basis yet have proven useful in a broad range of applications, despite their simplicity. Due to the "square" nature of the waveform, as can be seen from Figure A.22(a), Haar wavelets are less successful in applications (such as curve

editing and animation from Computer Graphics) which require more continuous function approximations [SDS95]. The Daubechies family of wavelets, shown in Figure A.22(b), are a generalization of the Haar wavelet that offer continuity. They are parameterized by the number of vanishing moments N (where Haar corresponds to N=1) and have a finite number of continuous derivatives. Coiflets, shown in Figure A.22(c), are also parameterized by the number of vanishing moments yet offer an extra dimension of continuity by requiring both the wavelet (mother function) and the scaling (father function) transformations to have the specified number of vanishing moments.

For both Daubechies and Coiflets, regularity increases with the number of vanishing moments. Keeping in mind that signals with gross irregularities (like our gene expression datasets) are better modelled by functions with fewer vanishing moments, we choose N=4 for both wavelet families. All three wavelet transforms operate on a two-dimensional matrix by applying the transform first to the rows to yield a set of coefficients with the same dimension as the original data, then applying the transform to the columns of this result, again producing a set of coefficients that are the same dimension as the original data.

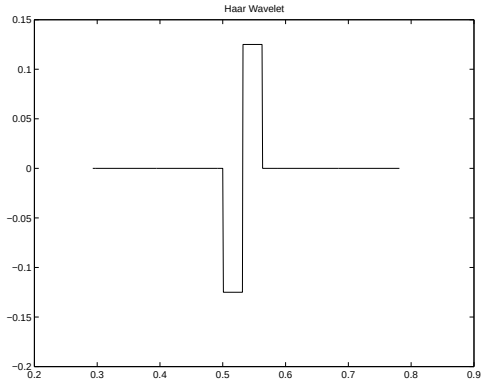
To evaluate the quality of the discretization procedures, we compare the similarity of the wavelet decomposition of the discretized data with the wavelet decomposition of the original data. Analogous to the image querying application, we are searching for the discretized result that most closely resembles the original. We compute a distance metric, which is the opposite of similarity. Let W_R be the set of wavelet coefficients found by applying the wavelet transform W on the continuous datamatrix R (with n rows and m columns) and W_D be the vector of wavelet correlation coefficients found by applying the same wavelet transform on a (scaled to mean) datamatrix resulting from some discretization method D. Then the *wavelet distance ($WAVD_{Haar}$, $WAVD_{Dau4}$ or $WAVD_{Coi4}$)* for D is found as:

$$\mathrm{WAVD}(R, D) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (W_R(i,j) - W_D(i,j))^2.$$

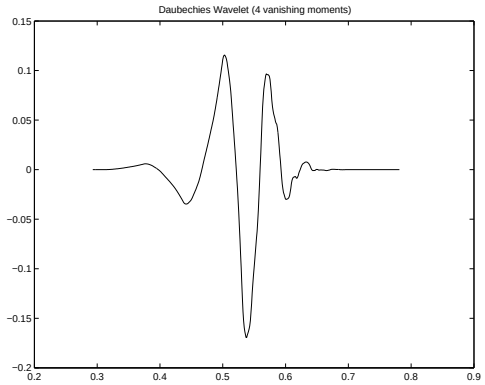
Principal Components Analysis Distance (PCAD)

The wavelet distance evaluation criteria measure the distance between the original and the discretized data in a transformed space, convolved in both the time and frequency domains. We consider an alternative based on *Principal Components Analysis (PCA)* which uses similar ideas yet transforms in a geometric space.

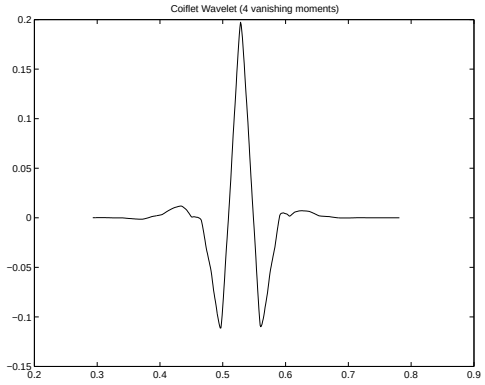
PCA is a multivariate procedure that produces a set of orthogonal vectors which serve as axes in a new data space. The vectors (*i.e.,* principal components) are found such that projections of the original data into the new coordinate system defined by the vectors (as axes) captures the dimensions of maximum variability in the data. The principal components are ordered such that the first component captures the dimension along which there is maximum variability in the data. The second component also captures the dimension of maximum variability yet is constrained to be uncorrelated (*i.e.,* orthogonal) to the first axis. The third and following components continue, finding the axes of maximum variation while remaining orthogonal to the previous components.



(a) Haar Wavelet



(b) Daubechies 4 Wavelet



(c) Coiflet 4 Wavelet

Figure A.22: Examples of wavelet functions: the Haar (Dau1), the Daubechies 4 (Dau4) and the Coiflet 4 (Coi4).

The set of (correlated) variables in the original data are thus transformed to a set of uncorrelated variables which are linear combinations of the original variables. The PCA vectors form a matrix of coefficients of the same dimension as the original data. There can be as many actual principal components as the minimum dimension of the original datamatrix.

As in the case of wavelets where low or zero wavelet coefficients could be discarded with minimal loss of information, PCA is a valuable tool for compression and data approximation as the later components can be discarded to reduce the dimensionality of dataset. This is achieved by setting the appropriate columns in the PCA coefficient matrix to zero. One of the most useful applications for such a data reduction is for visualization purposes. In Section A.3.2, two-dimensional graphs using two principal components provide a visual overview of the datasets used in our comparisons.

Figure A.23 demonstrates the PCA for two dimensional data, for which there can be a maximum of two principal components. In the top plot, the orthogonal red and green lines show the principal components which can easily be seen to lie along the axes of maximum variability in the data. The middle plot shows the original data rotated into the new coordinate space defined by these axes. The bottom plot shows the one-dimensional projection of the data onto the first principal component, a data reduction technique which results in the minimum loss of information in the data.

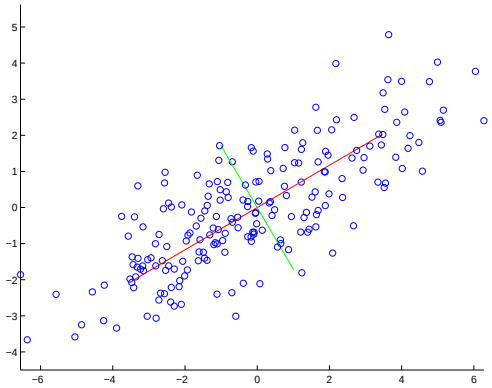
To use PCA as an evaluation measure for the discretization procedures, we compute the PCA vectors for the original data and rotate both the original data and the (scaled to mean) discretized version into the coordinate system defined by those PCA vectors. We then compute the MSE on the set of rotated vectors. Since MSE is invariant under linear transformations, we discard all but the top 10 principal components prior to the rotation by setting the corresponding columns in the PCA coefficient matrix to zero. Let P_i be the (ten-component) matrix for a dataset i rotated according to a set of PCA vectors P. Then the *PCA distance (PCAD)* is defined as:

$$PCAD(R, D) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (P_R(i,j) - P_D(i,j))^2.$$

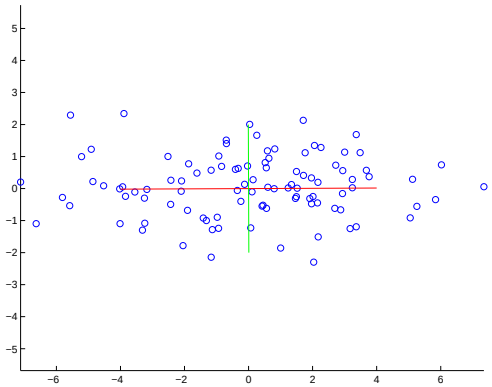
Figure A.24 demonstrates an example use of PCAD for three discretization method. We first apply PCA to a continuous dataset for nine variables to yield the axes P. Points marked as a red 'X' in each plot show the continuous data for nine variables projected into the PCA space defined by the first two principal components (P_R). The blue circles depict the two-component projection of the (scaled-to-mean) result of some discretization procedure D, projected according to P_D. Note that in Figure A.24(a), many of the circles completely overlap the Xs, whereas in Figure A.24(b) and Figure A.24(c) there are very few overlaps. By the PCAD measure, *Disc 1* would therefore have a better score than *Disc 2* or *Disc 3*. Similarly *Disc 2* scores better than *Disc 3*.

Pairwise Correlation Histograms (CorrHist and CorrHistRel)

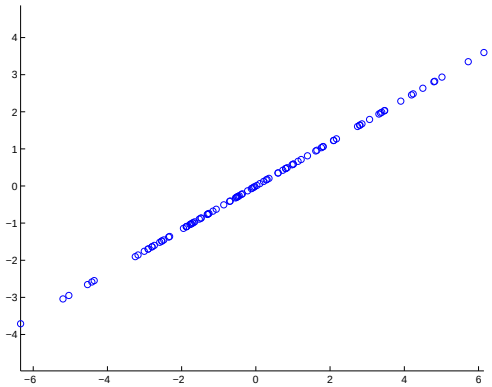
Our primary motivation is to capture dependencies between genes by creating models from gene expression data. This application offers its own evaluation metric. One typical approach to understanding gene expression data involves treating the expression data as profiles, such as for time series



(a) Two-dimensional data with two Principal Components

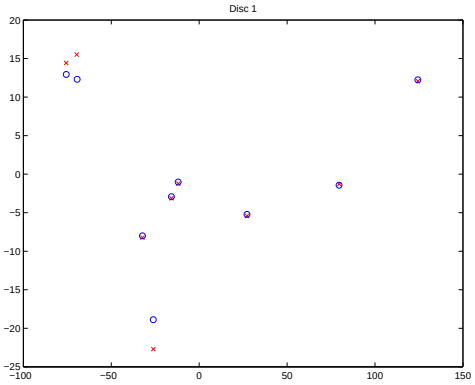


(b) Data rotated according to Principal Components

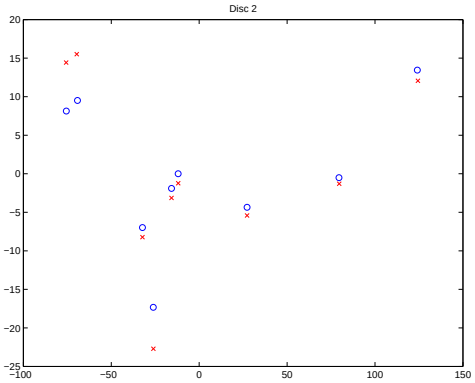


(c) Data projected onto first Principal Component

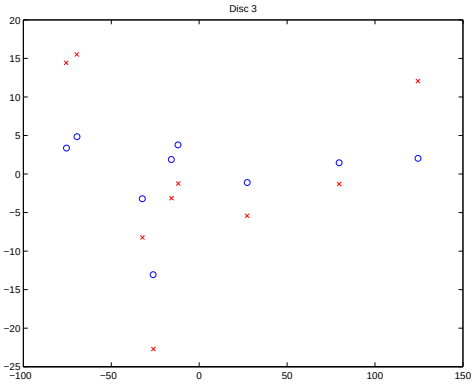
Figure A.23: PCA projections of Two-dimensional Data.



(a) PCAD for Disc 1.



(b) PCAD for Disc 2.



(c) PCAD for Disc 3.

Figure A.24: Example of applying PCAD.

data, and clustering the gene expression profiles to yield groups of genes whose expression profiles are similar. A fundamental concept in such clustering algorithms is to compute the similarity between profiles and one popular similarity measure is the Pearson correlation coefficient between profiles. Since clustering operates on the pairwise correlational structure of the data, it is desirable that any discretization procedure preserves that structure.

The *Pairwise Correlation Histogram evaluation measures (CorrHist* and *CorrHistRel)* introduced in this section attempt to quantify how well a discretization technique was able to preserve the set of pairwise correlations between variables. For a continuous-valued dataset R of n variables, let ρ_R be the length $n(n-1)/2$ vector of pairwise Pearson correlations calculated from R. Similarly, let ρ_D be the set of pairwise correlations calculated on the (scaled to mean) discrete dataset D which results from applying some discretization procedure to R. An 'ideal' discretization procedure would allow $\rho_R = \rho_D$, perfectly preserving the set of pairwise correlations. The inherent loss of information upon discretization means that such an ideal does not exist. However, we can quantify the loss of information by observing the total difference between ρ_R and ρ_D. Let CD be the $n(n-1)/2$ vector of absolute pointwise differences between ρ_R and ρ_D[3]:

$$CD(R, D) = |\rho_R - \rho_D|.$$

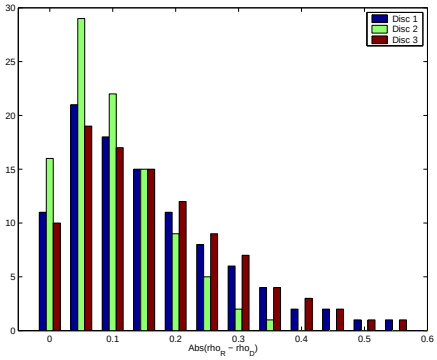
We can create a histogram of the set of differences CD, where the ideal distribution would have heavy support for values near zero. Figure A.25(a) demonstrates such a histogram for three hypothetical discretization techniques *Disc 1, Disc 2* and *Disc 3*. The vertical axis shows the number of differences, normalized to percentages, while the horizontal axis shows the actual difference. Note that the second technique, *Disc 2*, has a higher concentration of differences around the zero value and would thus be considered the better discretization technique.

When evaluating many different discretization techniques, histograms such as that of Figure A.25(a) soon become cluttered and unintelligible. Since our emphasis is on the (near) zero differences, we choose to show only the first two intervals, corresponding to differences of 0 and 0.05, as shown in Figure A.25(b) for the same data as in Figure A.25(a). We refer to such a histogram as the *Pairwise Correlation Histogram (CorrHist)* evaluation measure.

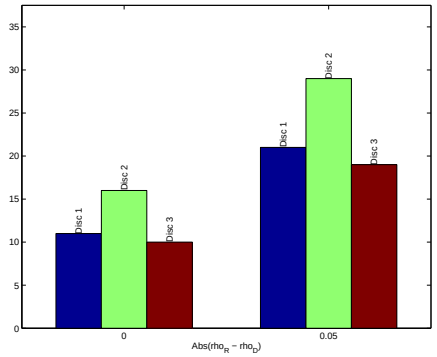
Figure A.25(c) shows the CorrHist for eight hypothetical discretization techniques, with each bar labelled *Disc 1* through *Disc 8*. Note that the label *Disc 4* has no bar, signifying that the technique was not applicable for this dataset. From the figure, we see that *Disc 5* and *Disc 6* are tied for the maximum number of differences of 0, yet *Disc 5* is higher than *Disc 6* for differences of 0.05, making *Disc 5* the best technique overall by this measure.

We might wish to scale the difference in correlation based on the actual value of the correlation since the significance of the difference depends on the magnitude. A difference of 0.01 when the real correlation is 0.98 is not as significant as when the original correlation was 0.01. Thus, we introduce a

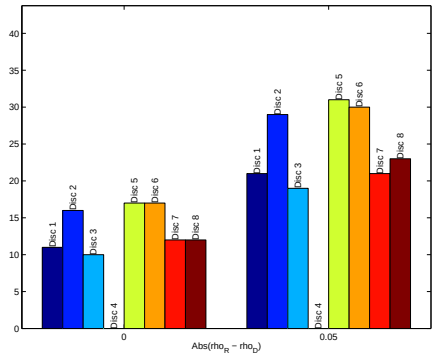
[3]Note that CD is similar to the quantity being optimized by the PDR techniques of Section A.2.12 though in PDR, correlations in the ρ_D term are calculated between the discrete version of a row i and the *real* versions of the remaining rows, while in CD, ρ_D is calculated between the discrete version of a row i and the *discrete* versions of the remaining rows.



(a) Correlation Histogram for three discretization techniques.



(b) Lower portion of previous histogram.



(c) Lower portion of Correlation Histogram for eight discretization techniques (CorrHist).

Figure A.25: Correlation Histograms (CorrHist)

second pairwise correlation histogram measure which shows the *relative* correlation difference CD_{rel}:

$$CD_{rel}(R, D) = \frac{|\rho_R - \rho_D|}{\rho_R}.$$

We refer to histograms of this difference as the *Relative Pairwise Correlation Histogram (CorrHistRel)* evaluation measure.

Hierarchical Clustering Comparisons (TreeComp)

The previous measures, CorrHist and CorrHistRel, attempt to quantify the loss of pairwise dependency information by examining the point-by-point difference between the set of pairwise correlations. We are interested not only in preserving the correlation for an individual pair independently but in preserving the overall joint correlation structure, *i.e.,* to preserve the correlation for all pairs simultaneously. One way to measure the ability of a discretization procedure to preserve joint correlations is to investigate whether (deterministic) algorithms operating on the discretized data arrive at the same results as when operating on the continuous data. We consider the particular task of *clustering* and calculate the differences when clustering using the continuous data and when using the discretized data. We formulate an evaluation measure based on *Hierarchical Clustering Comparisons (TreeComp).*

Hierarchical clustering is a bottom-up clustering technique which begins with singleton clusters and iteratively merges two clusters until a single cluster is achieved. We choose to use the *Pearson correlation coefficient* to compute the similarity between clusters and *complete linkage* to govern cluster merge decisions. The TreeComp evaluation measure examines the sequence of merges L_R when clustering the continuous valued dataset R and the sequence of merges L_D when clustering the (scaled to mean) discrete dataset D resulting from some discretization method. At each stage in the merging process, there exists a collection of clusters in both L_R and L_D. Considering the presence of a pair in L_R as a 'positive' and the absence of a pair in L_R as a 'negative', we evaluate the quality $Q_{D,s}$ of a discretization at merge level s by counting the number of true positives and true negatives in L_D with respect to L_R and divide by the total number of possible pairs:

$$Q_{D,s} = \frac{\text{true positives} + \text{true negatives}}{\text{number of pairs}}.$$

We provide an example in Figure A.26 which shows the result of clustering a dataset of five variables: $\{a, b, c, d, e\}$. The dendrogram on the left depicts the series of merges when clustering the continuous dataset R for these variables. The dendrograms to the right depict the series of merges when clustering the (scaled to mean) discretized datasets, D_1 and D_2 which resulting from two different discretization methods. We begin at the root of the dendrogram (N=1) where we can then divide the data into two clusters. The set of clusters in R is $[\{abcd\}, \{e\}]$, which is identical to the cluster set of D_1 at $N = 2$, giving a score of $Q_{D_1,1} = 10/10 = 1$ since there are six true positives (ab, ac, ad, bc, bd, cd) and four true negatives (ae, be, ce, de) out of 10 possible pairs. The cluster set for D_2 is $[\{abc\}, \{de\}]$, which has three true positives (ab, ac, bc) and three true negatives (ae, be, ce),

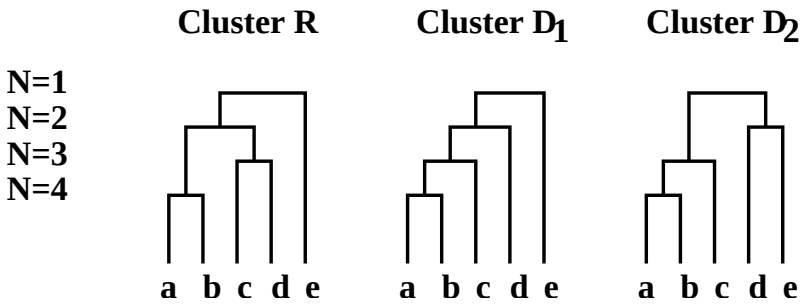


Figure A.26: Hierarchical Clustering

yielding $Q_{D_2,1} = (3+3)/10 = 0.6$. We can continue computing the quality scores for $N = 3$ and $N = 4$.

Figure A.27(a) plots quality scores $Q_{D_i,s}$ over the levels s for each of three discretization procedures D_i for a hypothetical dataset. Note that *Disc 2* has a higher quality score than the other curves for many of the levels N. If the correlation structure could be perfectly preserved after discretization, the value $Q_{D_i,s}$ would be 1 over all levels, meaning that the dendrogram L_{D_i} was the same as the dendrogram L_R. Thus, we consider discretization procedures which are at the maximum of the curve over all levels to be superior. When comparing many discretization techniques, the plot becomes unreadable so we report a summary value of the number of levels a particular technique is rated superior to others (or tied for superior).

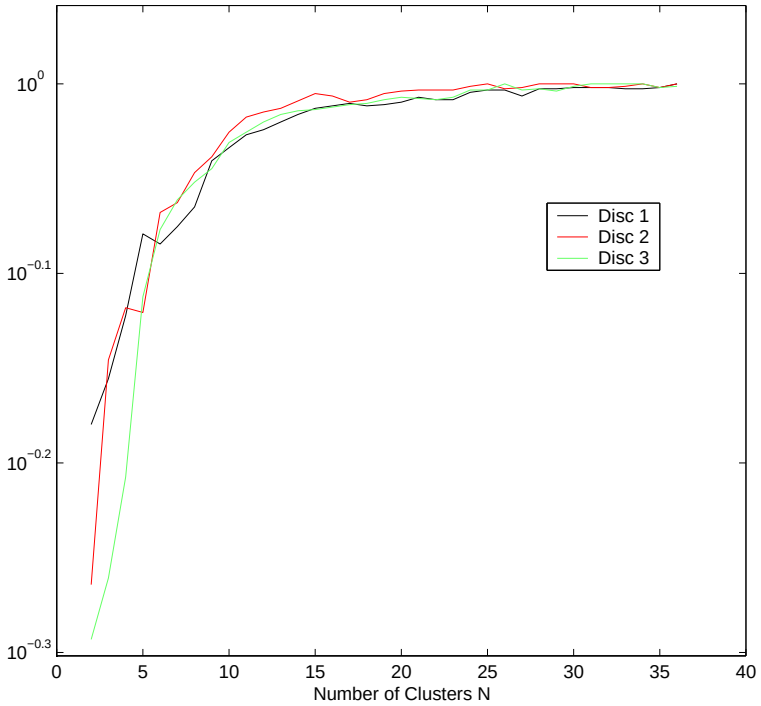
Let $rank(D_i, s)$ be the rank of quality scores $Q_{D_i,s}$ for a discretization procedure D_i over all discretization procedures at level s, and let $\delta(D_i, s)$ be equal to 1 if D_i has the best (tied) rank and 0 otherwise. The *hierarchical clustering comparison (TreeComp)* evaluation measure is then calculated as the number of levels in which a particular method achieves the best quality score relative to the other discretization methods:

$$TreeComp(D, R) = \sum_{s=1}^{n-1} \delta(D, s).$$

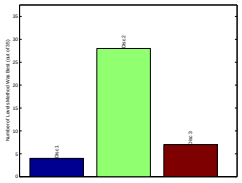
Figure A.27(b) plots the TreeComp scores for the data in Figure A.27(a). The vertical axis is the number of levels in which the particular discretization method was best, where ties are shared. *Disc 2* clearly outperforms the other two methods with respect to the ability to induce a hierarchical clustering most similar to the clustering found using the dataset R.

A.3.2 Datasets

Typical model induction problems feature a large number of samples across a few variables; in fact, most statistical techniques are optimized for large samples. However, gene expression data usually interrogates thousands of genes simultaneously across a disproportionately smaller number of samples. This reversal of dimensionality represents a unique challenge not only for model induction but for discretization. Though our original motivation is to learn Bayesian Networks from gene



(a) Plot of $Q_{D_i,s}$ for three discretization methods



(b) Plot of TreeComp for data in (a)

Figure A.27: Hierarchical Tree Comparisons (TreeComp)

expression data, we would like the discretization methods to perform well across a wide range of datasets, not just gene expression data.

In our comparisons, we consider a diverse collection of datasets from different applications, varying the size, type and intent of the dataset. We include seven different gene expression datasets, representing different microarray platforms, different organisms, different sample sizes, and different experimental conditions (**CF(Ratio), ADENO(Ratio) SPELLMAN, YVERT464, YVERT465, GASCH** and **HUGHES**). We also consider two datasets generated from fictitious and real Bayesian networks (**CG1** and **ALARM**) as well as three datasets from the UC-Irvine Machine Learning archive (**SEGMENT, IONO,** and **SONAR**). In the following sections, we discuss the salient aspects of each dataset in turn.

Since many of the expression datasets have missing values, we impute the value of a gene in a given (missing) sample as the average value in that sample of the five nearest neighbors (as measured by Euclidean distance across all samples) among those that have data measured in at least two-thirds of all samples. Also, due to the scale of the study which requires repeatedly applying several computationally intensive discretization procedures and evaluation metrics, we limit the number of variables in any dataset to a reasonable number. For the datasets with thousands of variables, we sample five subsets of 100 variables, with the first sample containing those variables with the highest variance across samples. The remaining four subsets are (potentially overlapping) random samples from the set of variables. The evaluation measures applied to the larger datasets thus become averages over the five subsets.

Human Cystic Fibrosis Oligonucleotide Data (CF and CFRatio)

The *CF* and *CFRatio* datasets derive from human lung tissue samples from three patients suffering from Cystic Fibrosis, as well as six control lung tissue samples [WLW+02]. Cystic fibrosis is an inherited disorder in which patients secrete very thick mucus from their lungs. The samples from CF patients were processed in duplicate, with one patient lending samples from two distinct lung locations, resulting in a total of eight CF samples. Experiments were performed using the Affymetrix Human 6800 oligonucleotide microarray, allowing *absolute* levels of gene expression to be measured for 6086 genes across the 14 tissue samples. We refer to this data as the *CF* dataset. We include this dataset to demonstrate a sample-limited application in Human patients and also to view the use of replicate samples (i.e., non-independent samples).

To make the ratio-based discretization method FC applicable to this type of gene expression data, we also created a dataset of *relative* levels of gene expression by dividing the data for each gene by the mean intensity across all samples, then taking the (base-2) logarithm of this ratio. Transforming the absolute expression levels from an oligonucleotide array platform makes the data roughly comparable to that from the second common gene expression platform, cDNA microarrays, which we give examples of later. We refer to the ratio dataset as the *CFRatio* dataset.

To gain a high-level appreciation for the characteristics of the datasets, we examine two-dimensional

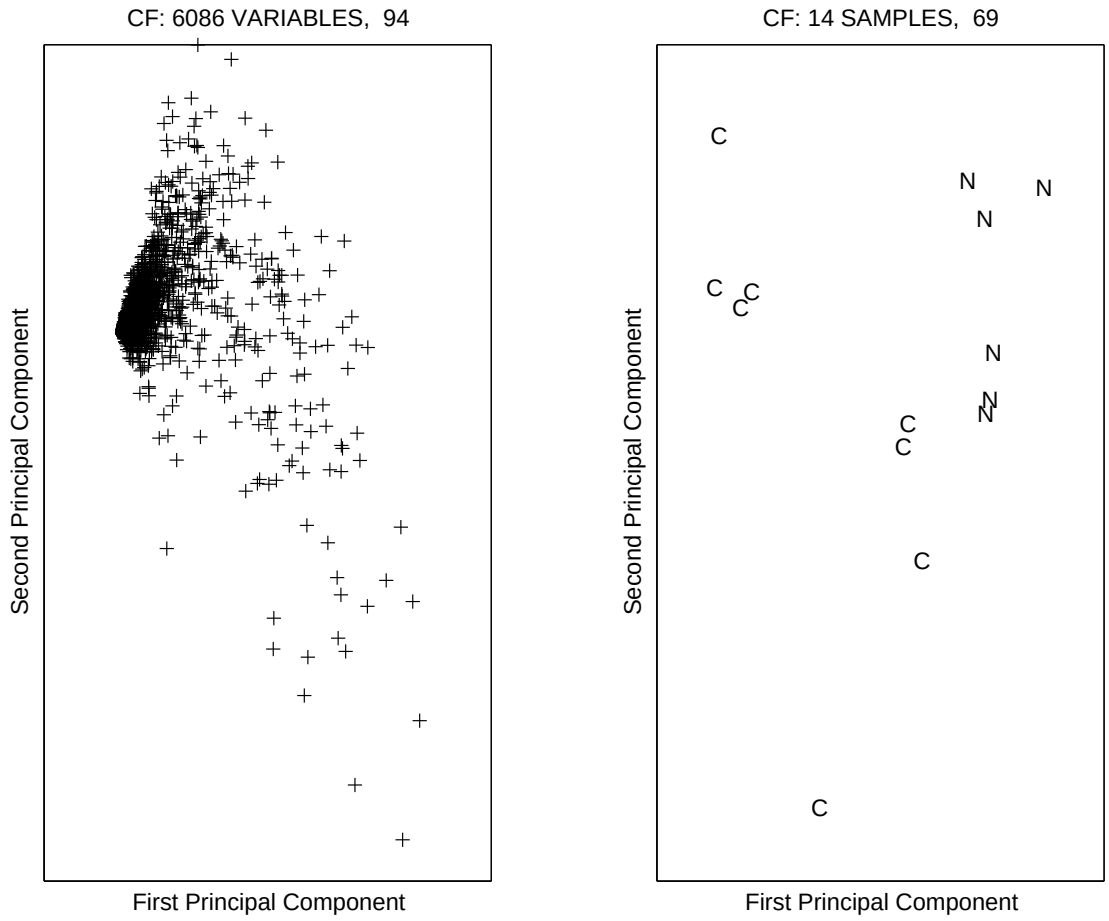


Figure A.28: PCA Projection of the CF dataset

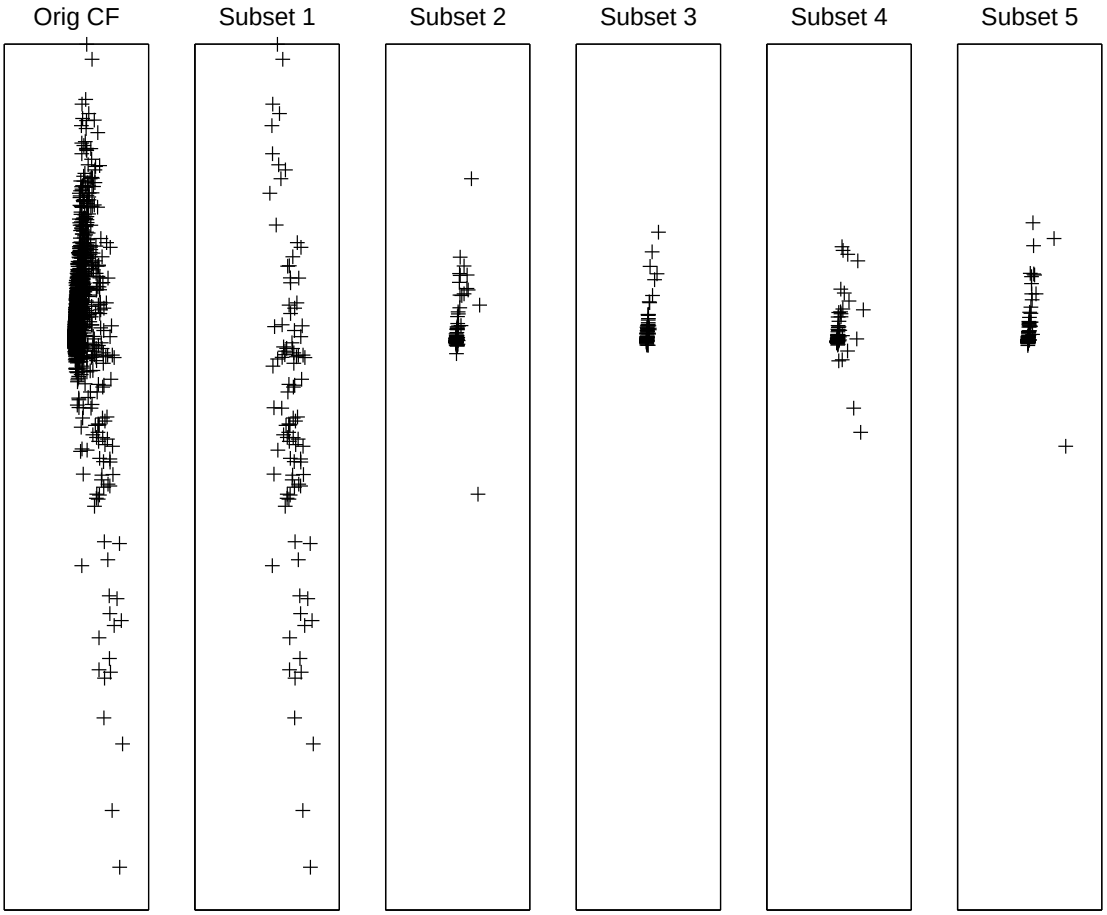


Figure A.29: PCA Projection of the five CF subsets

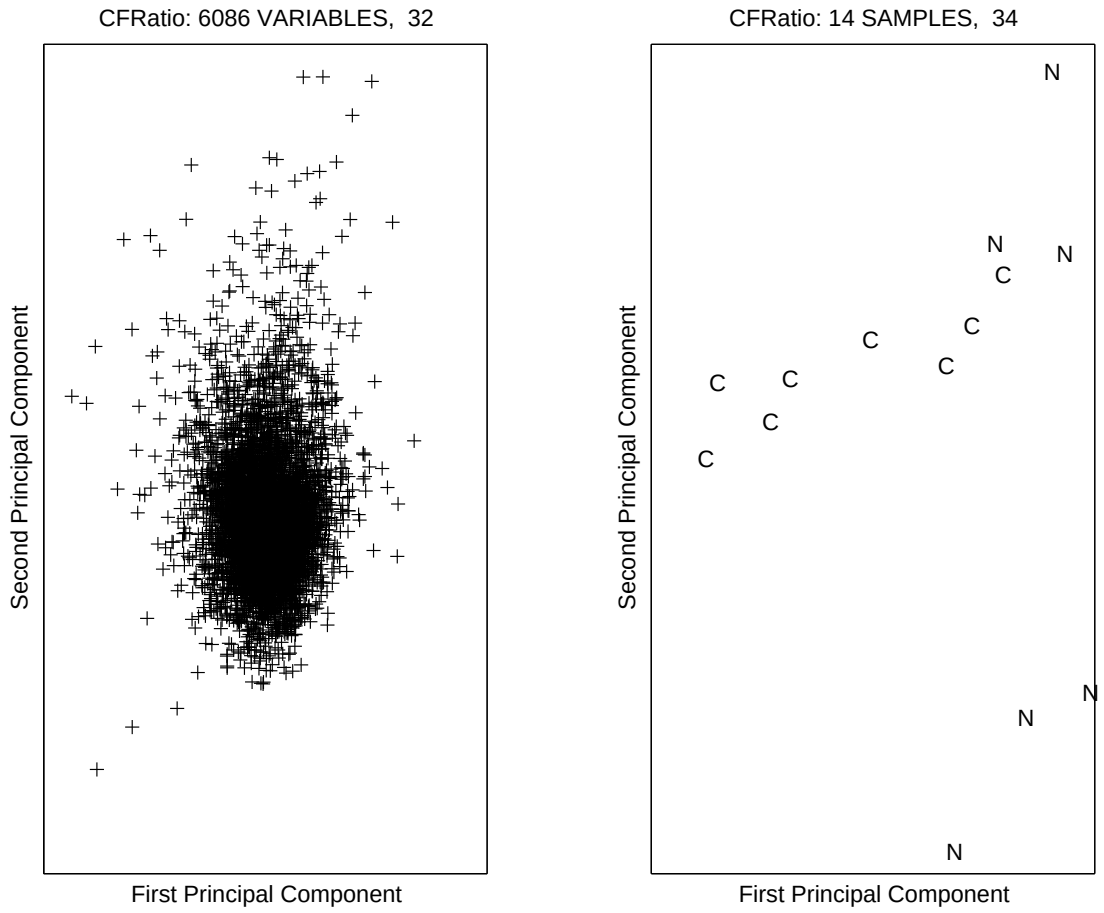


Figure A.30: PCA Projection of the CFRatio dataset

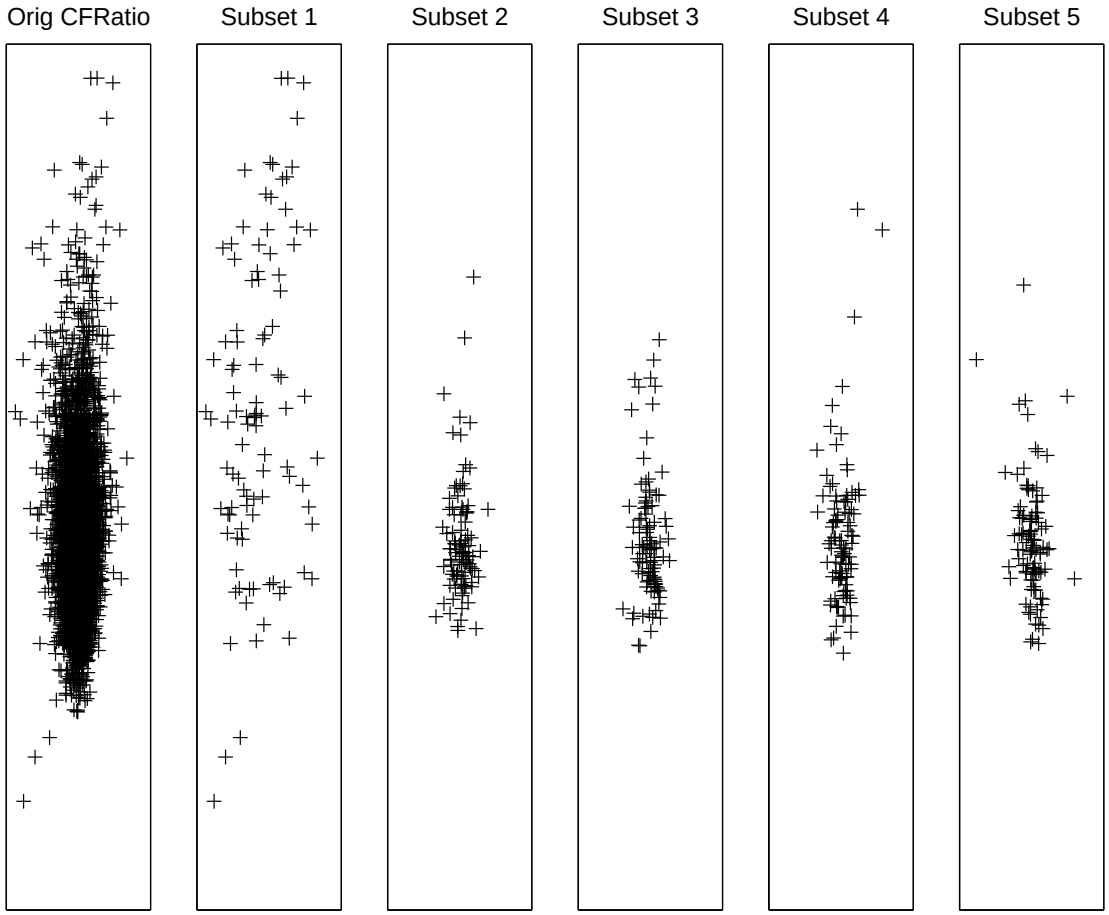


Figure A.31: PCA Projection of the five CFRatio subsets

projections of a Principal Components Analysis of the datamatrix applied to both the variables dimension and the samples dimension (*c.f.* Section A.3.1). The plot to the left in Figure A.28 shows the data for the 6086 variables with respect to the first two principal components, which in the case of the CF data, represents 94% of the variance in the data (the percentage is shown as '94' in the title of the plot). Note that the majority of points are clustered in a dense cloud in the center of the figure, with outliers fanning out around the cluster. The plot in the right of Figure A.28 shows the data for the 14 samples projected into the first two principal components (accounting for 69% of the variance). The samples are labelled as either cystic fibrosis samples ('C') or normal samples ('N'). Note that the class of 'C' versus 'N' is clearly discernable from the figure though the sample identity information is not explicitly included in the dataset.

Since the number of variables is large for this dataset, we sample five subsets of 100 variables. PCA projections of the subsets of variables are shown in Figure A.29, along with the original dataset for comparison. Note that Subset 1, the dataset created from the top 100 high variance variables, contains many of the outliers from the original dataset, while the other four (random) subsets are sampled from the densely populated cloud of points.

Figure A.30 and Figure A.31 show analogous plots for the ratio version of the CF dataset. Note that the cloud of points has become more circular in the plot on the left of Figure A.30 as a result of the data normalization. Also note that the percent variance explained has dropped to 32 percent. The 'C' versus 'N' distinction between samples remains clear in the plot on the right of Figure A.30 though the margin appears to be reduced. In Figure A.31, notice again that the first subset represents the 100 variables with the highest variance across samples.

Human Adenocarcinoma Oligonucleotide Data (ADENO and ADENORatio)

The *ADENO* and *ADENORatio* datasets also derive from human lung tissue samples for ten patients suffering from adenocarcinomas, a type of cancer of the lungs effecting cells lining the lungs. Samples were taken in duplicate from each patient, in both a normal and diseased portion of the lung, to yield a total of 39 tissue samples (one sample failed) [JJLL02]. The tissue was hybridized to Affymetrix Human HG_U95Av2 oligonucleotide microarrays, a second generation chip with 12625 genes. As with the CF dataset, we create a ratio dataset from the original data. We refer to the original dataset as the *ADENO* dataset and the corresponding $\log_2$ based ratio version as the *ADENORatio* dataset. We include this dataset as an example of a moderate-sized gene expression study with matched and replicated samples.

Figure A.32 and Figure A.33 depict the PCA projections of the datasets which have characteristics similar to their CF counterparts. Since the implementation of PCA used cannot compute for the full dataset of over 12000 variables, we show PCA projections for 6000 variables, a large enough sample to provide an accurate picture of the data while remaining tractable. The percent variance explained by the first two principal components is given as the last number in the title of each plot. Again the samples are labelled 'C' or 'N' corresponding to cancer and normal tissue samples. Note that the samples are also clearly separable in the PCA view, though the classification of samples

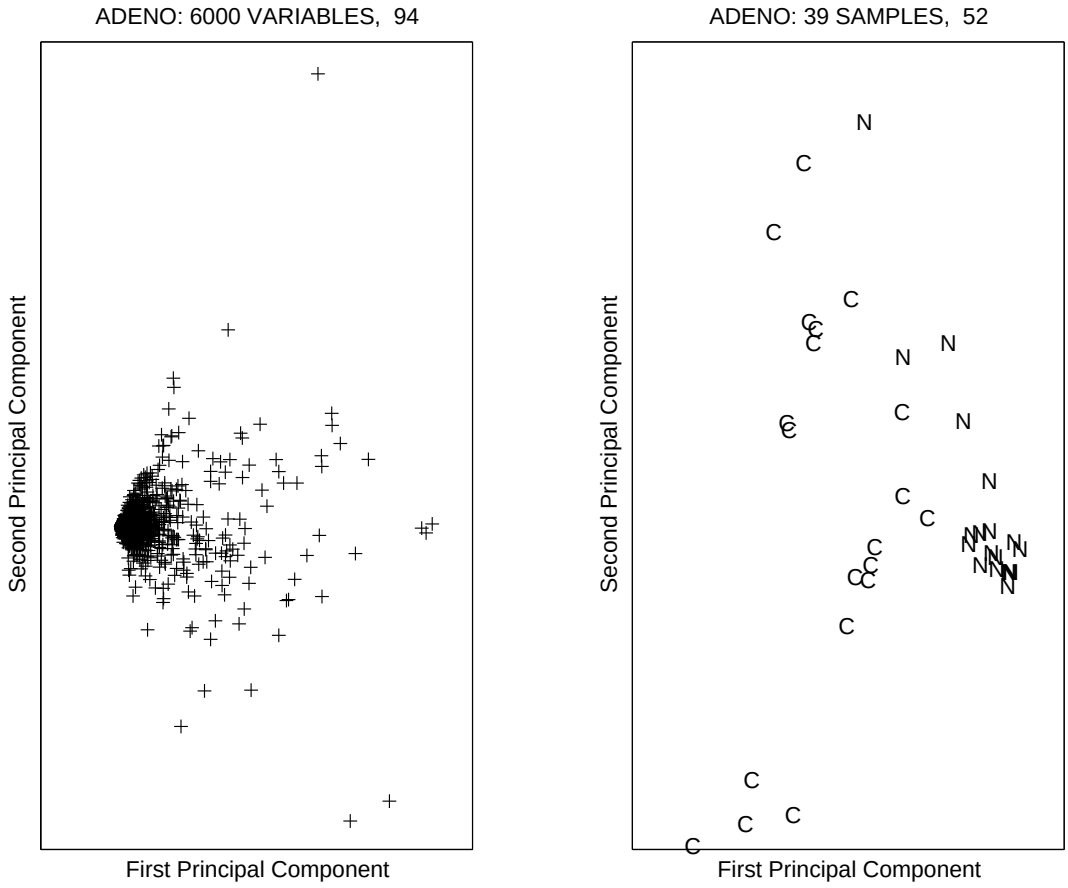


Figure A.32: PCA Projection of the ADENO dataset

was not explicitly supplied in the dataset. The subsets of 100 genes for ADENO and ADENORatio are shown in Figures A.34 and A.35 respectively.

Yeast Cell Cycle cDNA Data (SPELLMAN)

The *SPELLMAN* dataset derives from a study of the time series of gene expression for 5344 yeast genes across almost three full cell cycles of growth [SSZ+98]. Four different techniques were used for synchronizing cell development and samples were taken in roughly ten minute intervals, resulting in a total of 73 samples. Gene expression was measured by a cDNA microarray as the $\log_2$ ratio of expression relative to a reference time point for each synchronization method. We refer to this dataset as the *SPELLMAN* dataset. We include this example to demonstrate a moderate-sized time-series gene expression dataset in yeast.

PCA projections of the data are given in Figure A.36. Note that as in the ratio data plots of CFRatio and ADENORatio, the data occupies a roughly circular cloud of points in the plot to the left in both figures. The samples in the right-side plots are not labelled for this dataset. Figure A.37 shows the five subsets of 100 genes.

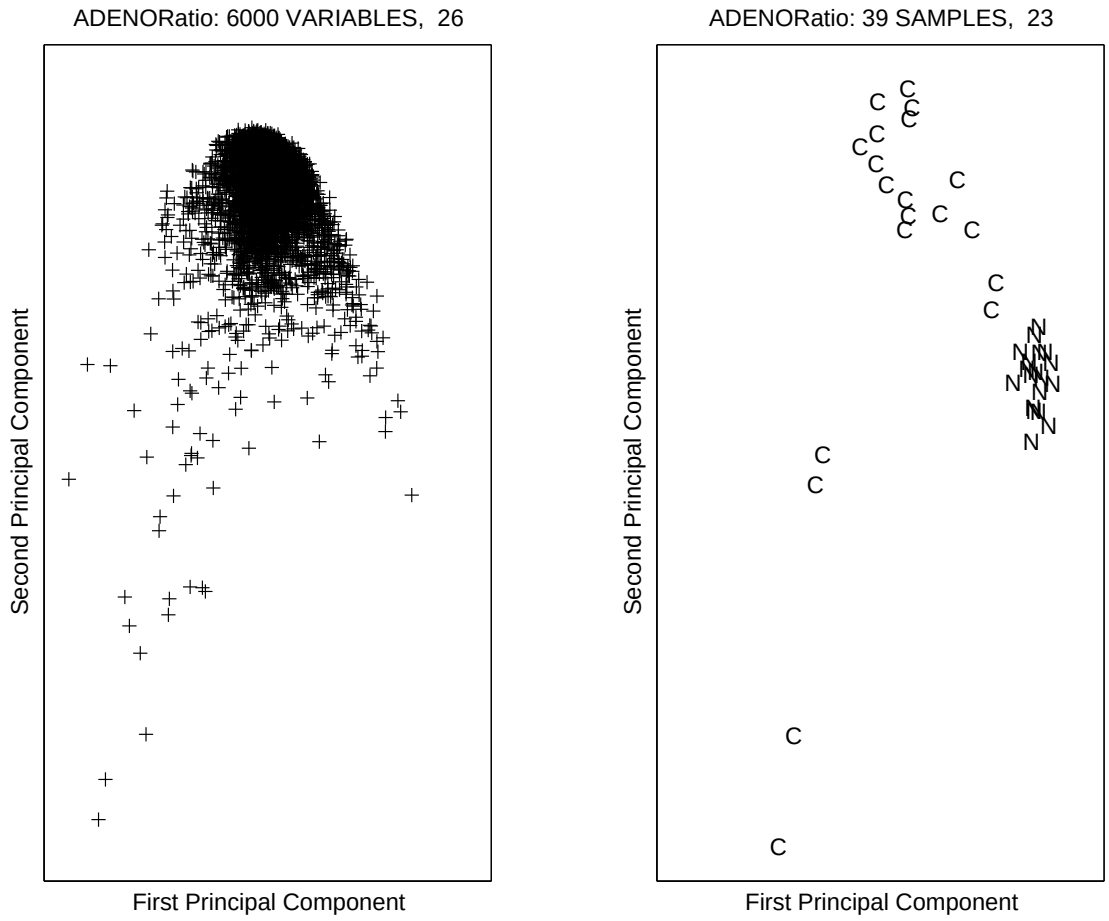


Figure A.33: PCA Projection of the ADENORatio dataset

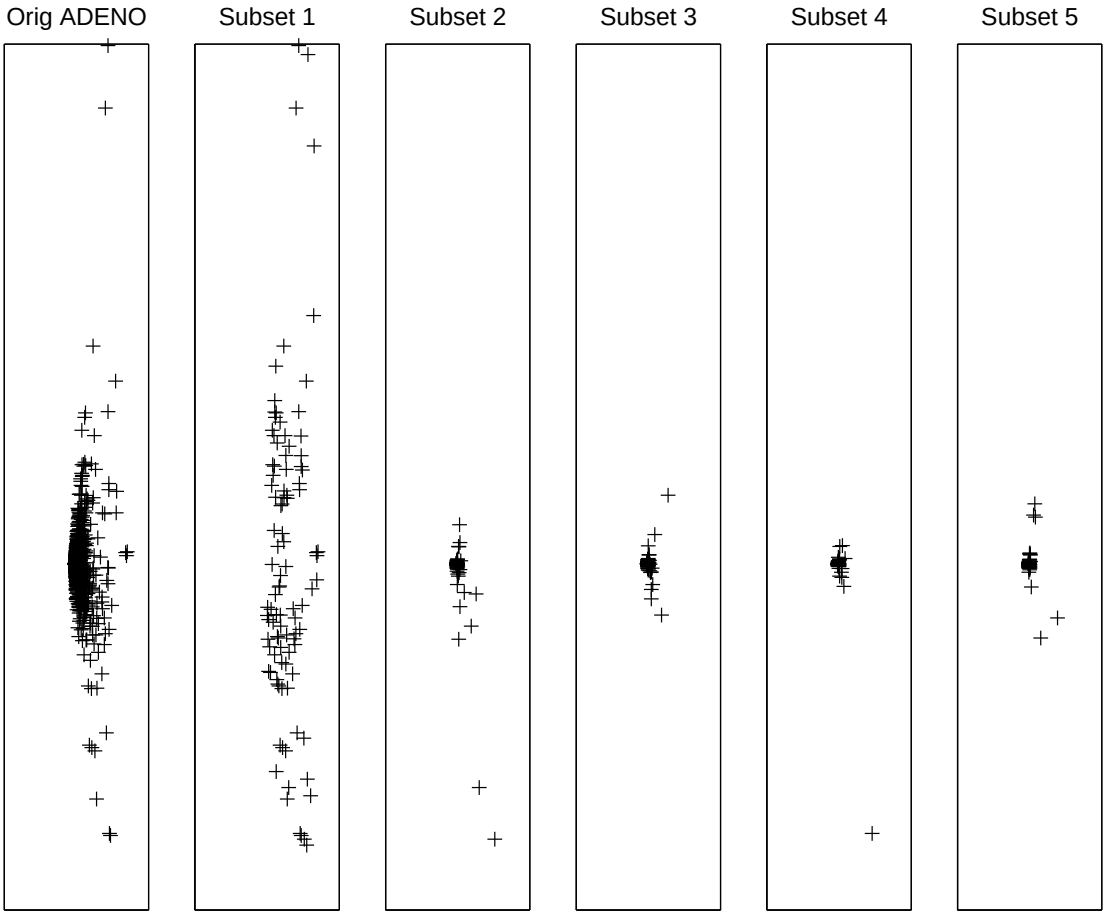


Figure A.34: PCA Projection of the five ADENO subsets

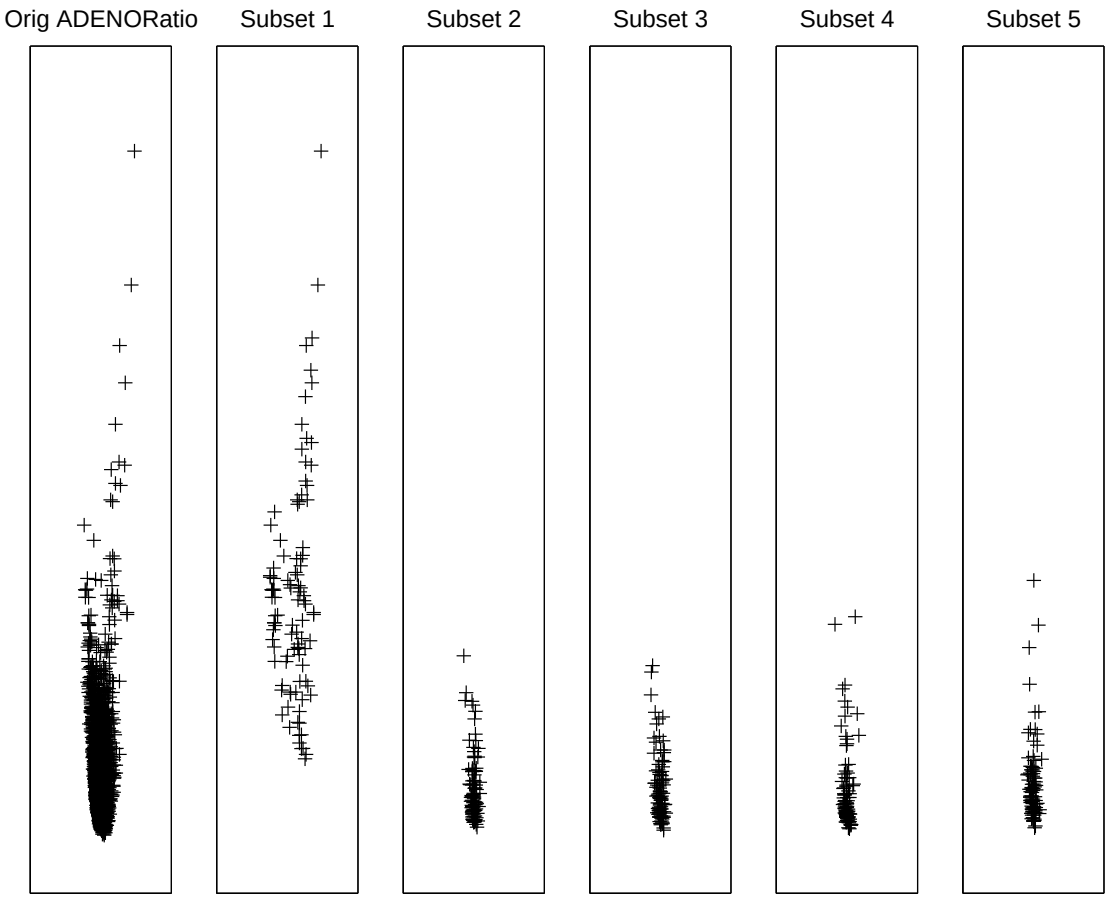


Figure A.35: PCA Projection of the five ADENORatio subsets

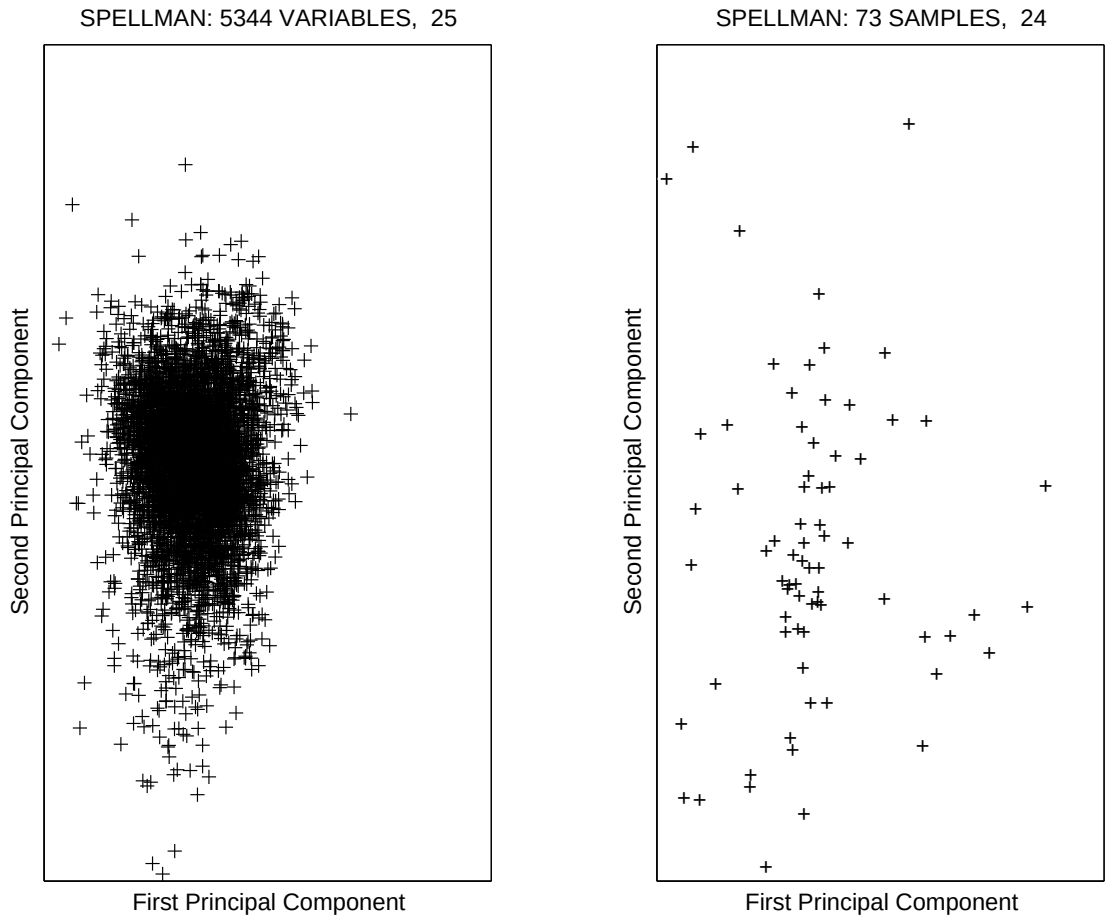


Figure A.36: PCA Projection of the SPELLMAN dataset

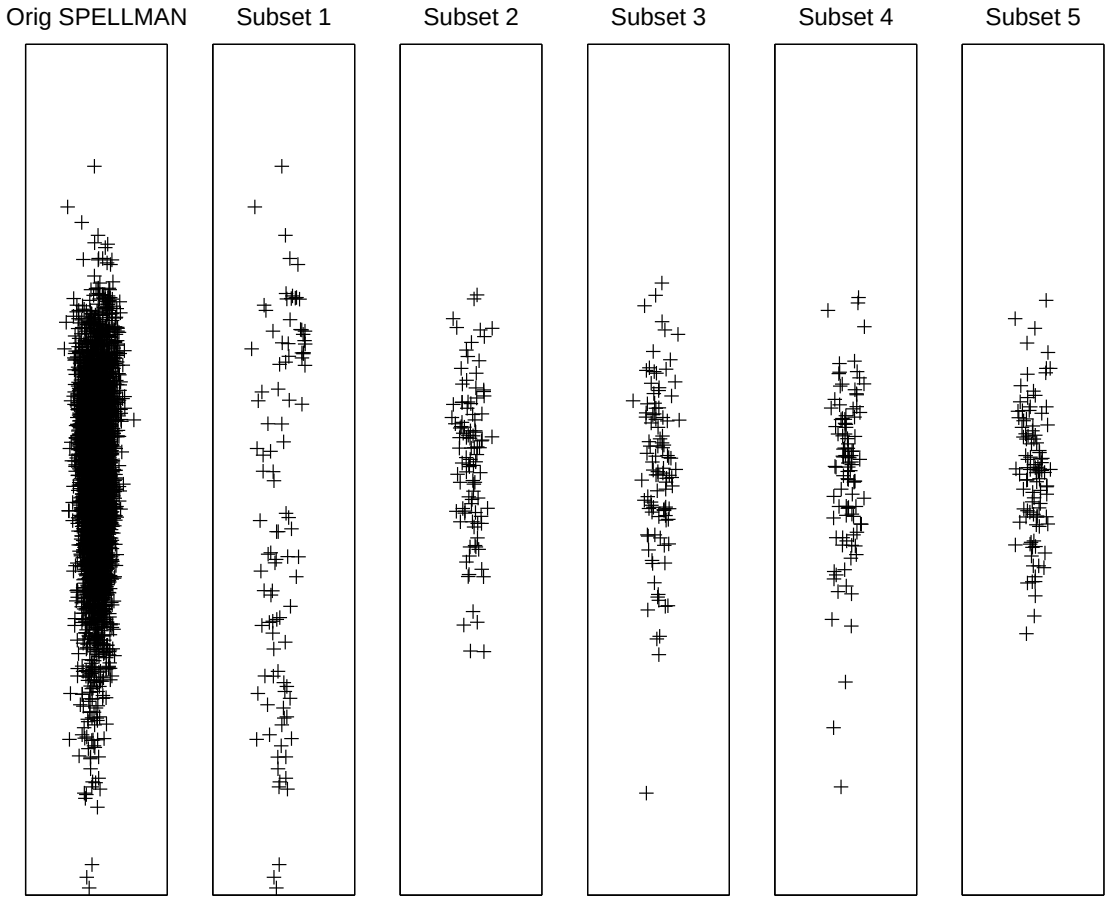


Figure A.37: PCA Projection of the five SPELLMAN subsets

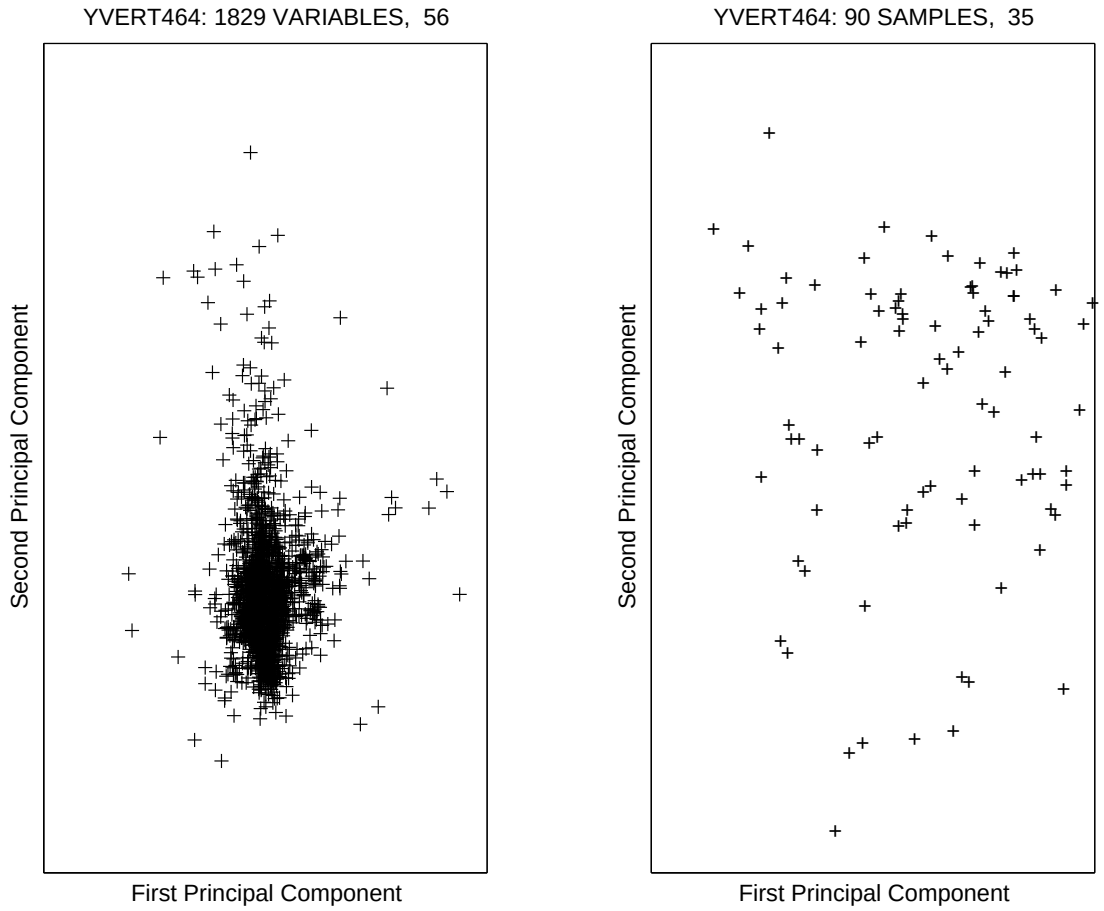


Figure A.38: PCA Projection of the YVERT464 dataset

Yeast Trans–acting Regulatory Variation cDNA Data (YVERT464 and YVERT465)

The *YVERT464* and *YVERT465* datasets derive from a study of the genetic variation which affects gene regulation [YBW+03]. Each dataset contains 90 samples hybridized to cDNA microarrays, where *YVERT464* measures 1829 genes while *YVERT465* measures 2222 genes. The data are the $\log_2$ ratio of expression relative to a reference sample and YVERT465 represents the *dye-swap* version of YVERT464, a technique which switches whether the reference sample appears as the numerator or the denominator in the ratio. Dye-swapping is often done to overcome biases in the technology of cDNA microarrays. We include this example as a demonstration of dye-swaps as an alternative form of (essentially) replicated data. PCA projections are shown in Figure A.38 through Figure A.41. Note that typical of ratio data, the variables form a cloud of points.

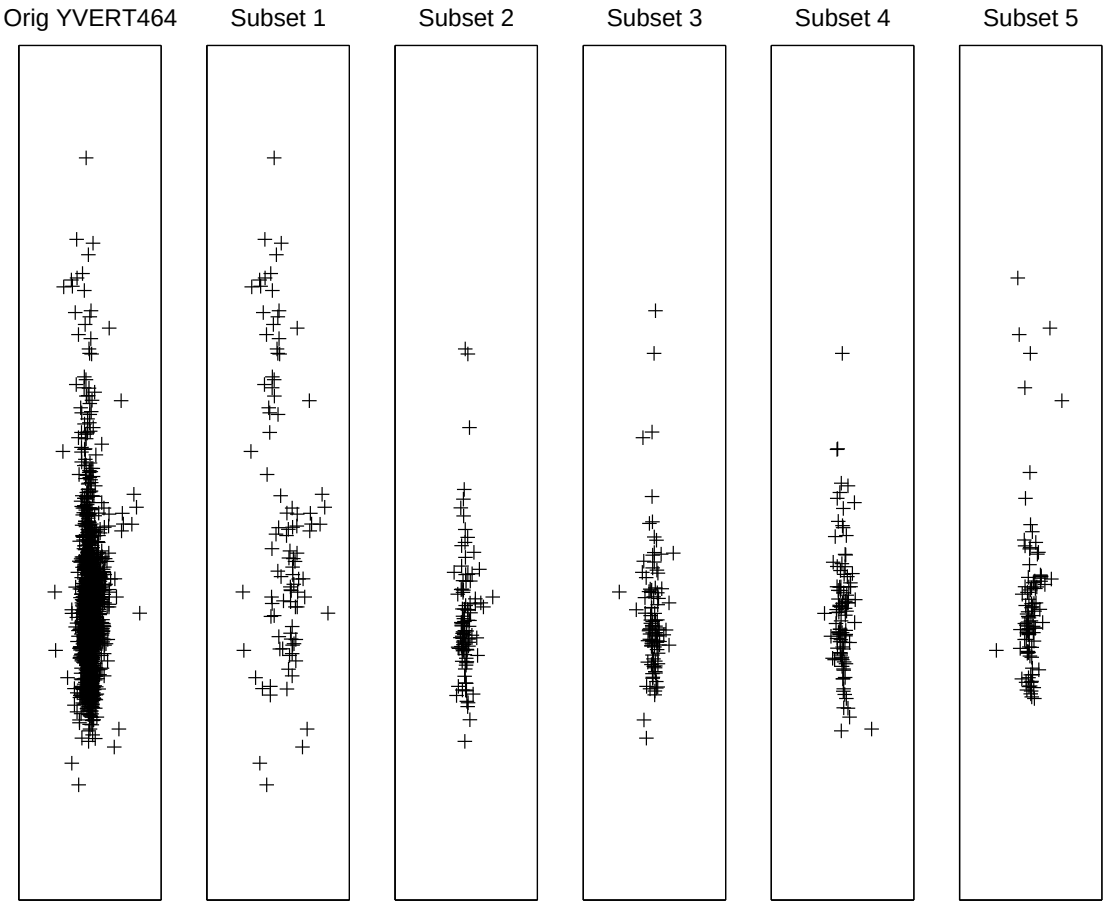


Figure A.39: PCA Projection of the five YVERT464 subsets

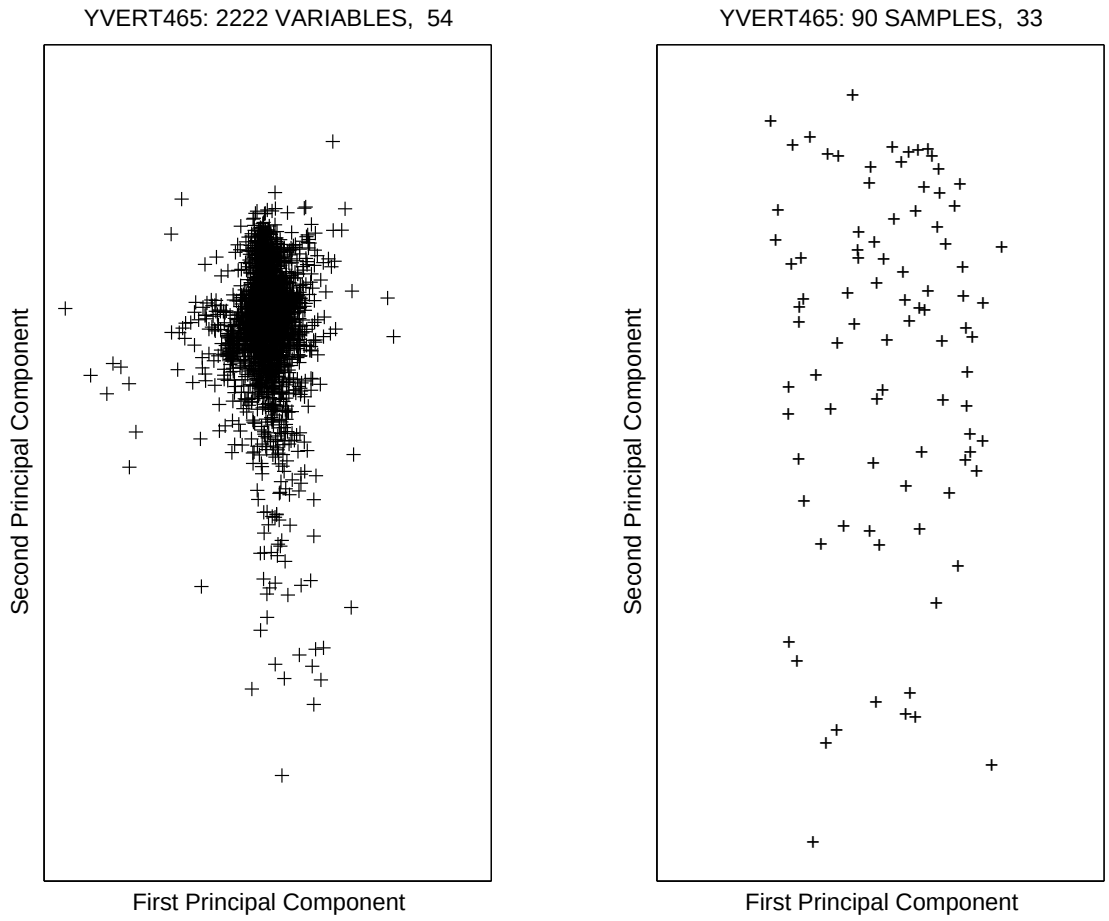


Figure A.40: PCA Projection of the YVERT465 dataset

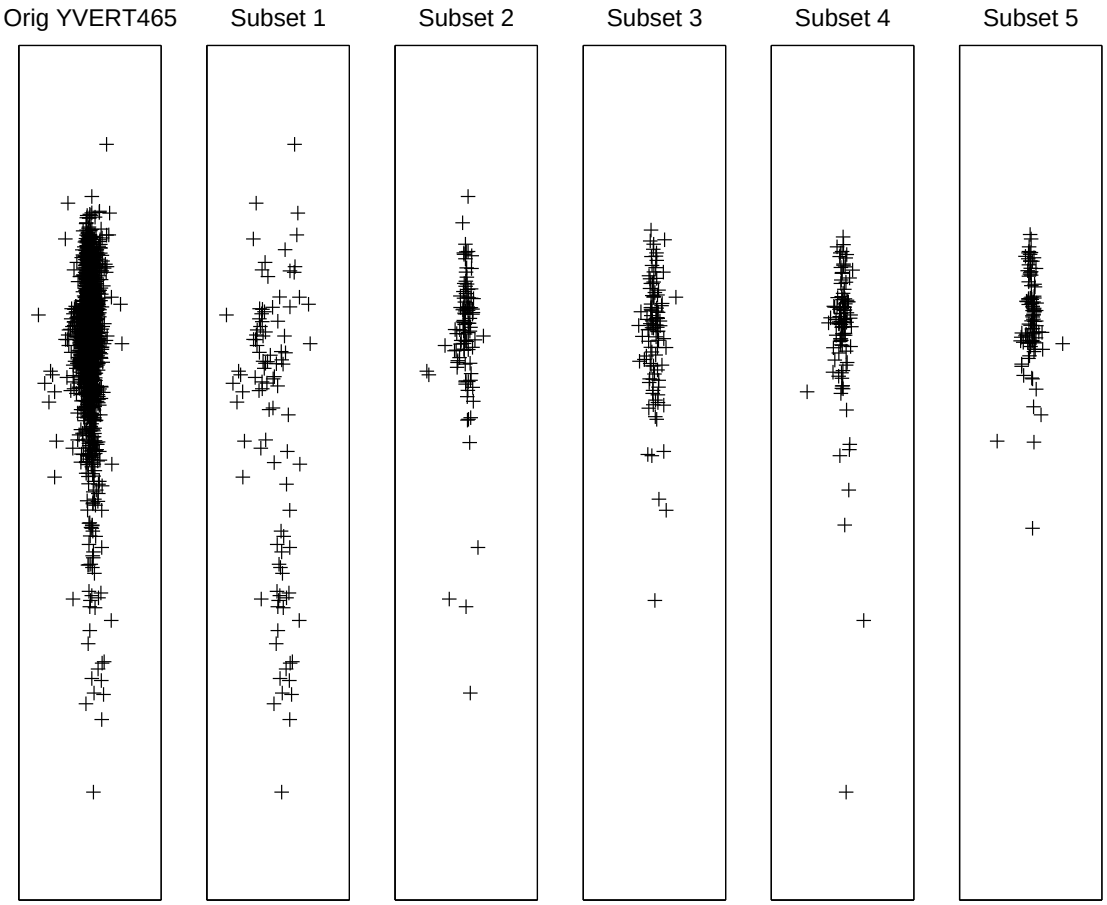


Figure A.41: PCA Projection of the five YVERT465 subsets

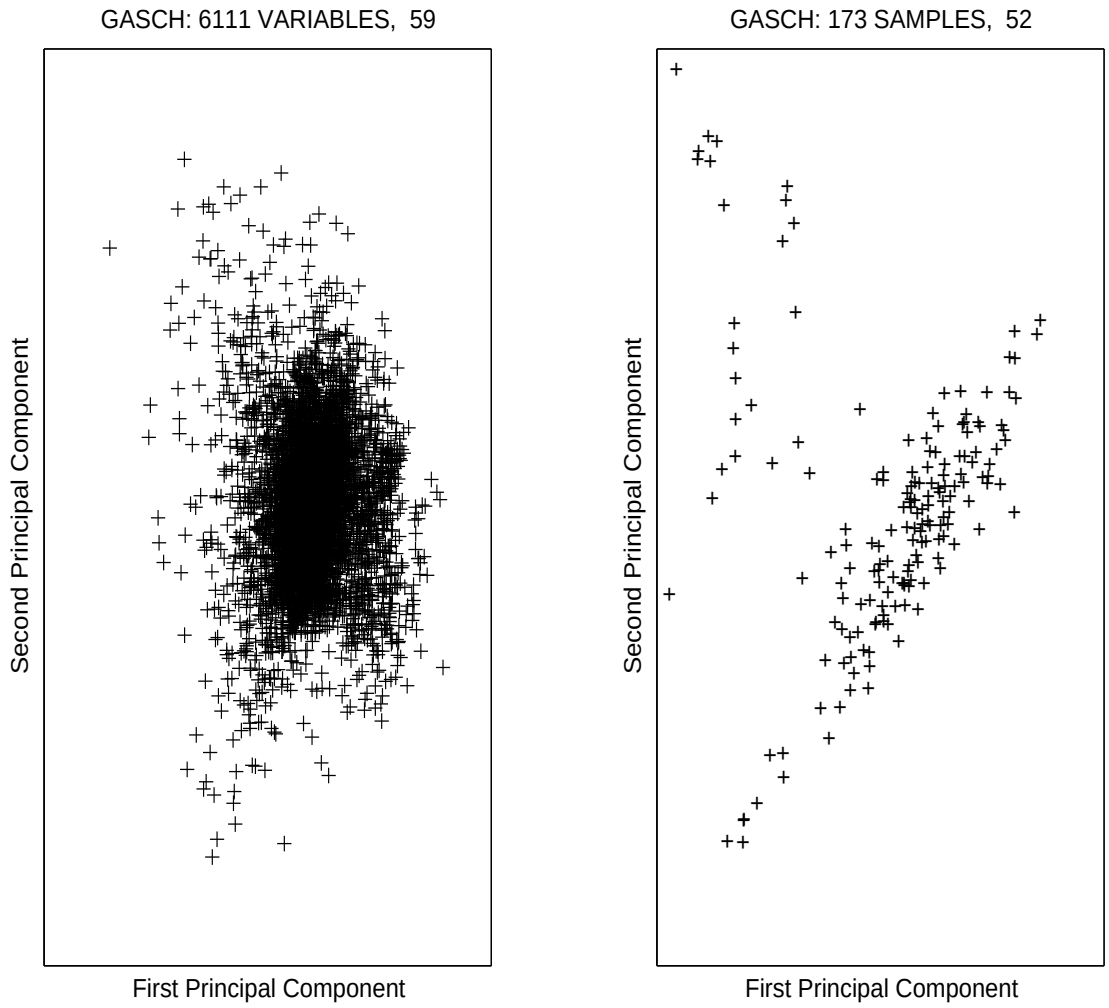


Figure A.42: PCA Projection of the GASCH dataset

Yeast Stress Response cDNA Data (GASCH)

The *GASCH* dataset derives from a study of the response of yeast over time to a large host of environmental changes, including temperature changes, nutrient deprivation, and drug dosages among others [GSK+00]. Data is measured for 6111 genes over 173 conditions using cDNA microarrays. We refer to this dataset as the *GASCH* dataset and include it as an example of a larger yeast gene expression dataset comprised of several time-series datasets. Figures A.42 and A.43 show PCA projections for the dataset. Note the elongated cloud of samples in the right-side plot in Figure A.42, perhaps indicative of the time-series nature of the data.

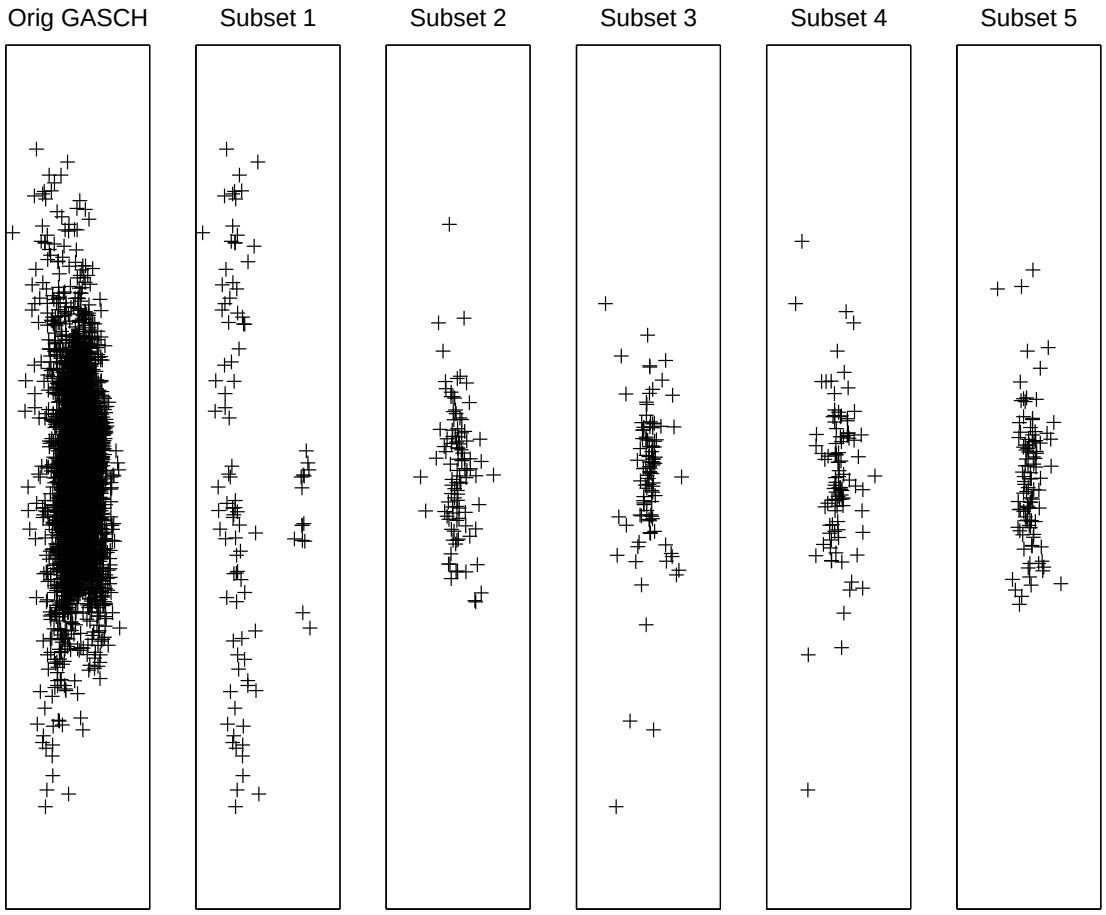


Figure A.43: PCA Projection of the five GASCH subsets

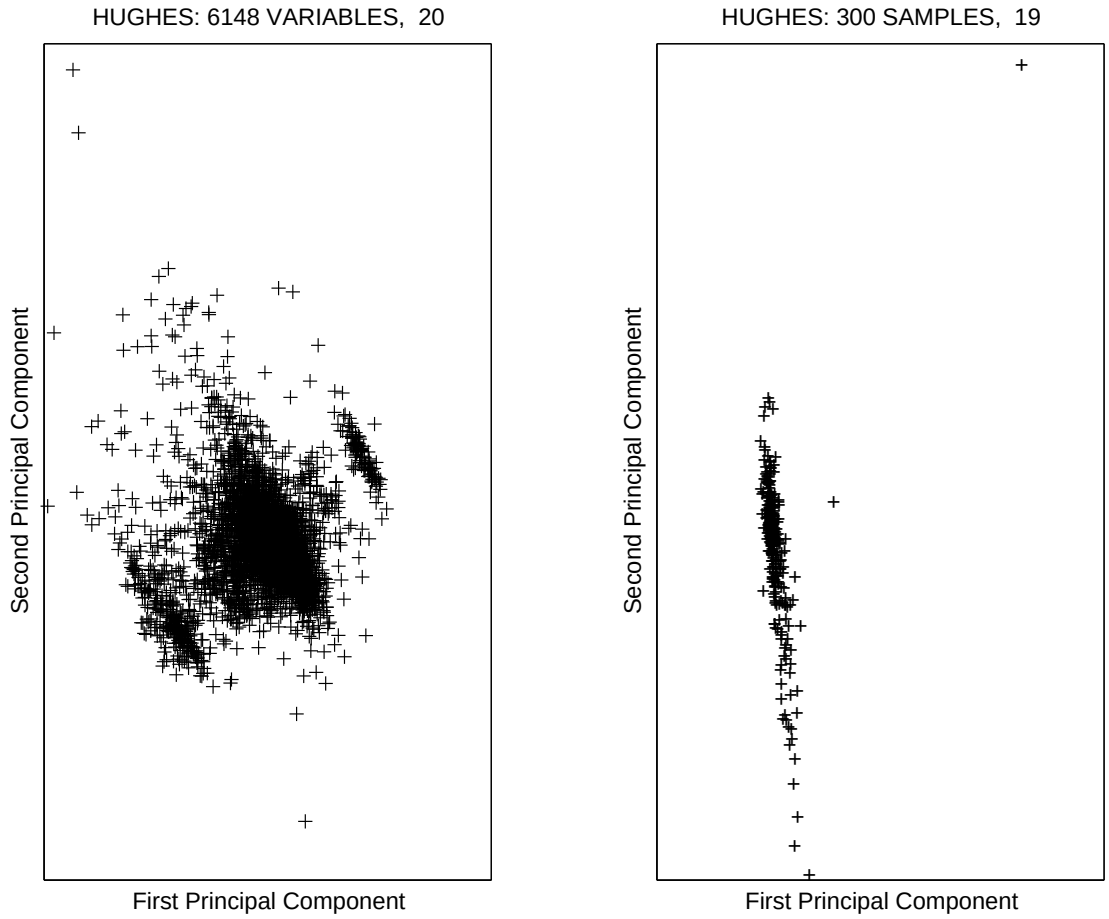


Figure A.44: PCA Projection of the HUGHES dataset

Yeast Deletion and Perturbation Compendium cDNA Data (HUGHES)

The *HUGHES* dataset derives from a study of 300 diverse gene mutations and chemical treatments in yeast [HMJ+00]. The dataset measures expression levels for 6148 genes by cDNA microarrays. We include this example as a larger example of yeast expression data; note that unlike the GASCH dataset, the samples are not from time-series data. Figure A.44 and Figure A.45 contain the PCA projections of this data. Note the very low values for the percent variance explained, indicative of the diversity of the environmental perturbations applied in the samples.

Waste Incinerator Emissions Bayesian Network Data (CG1)

The *CG1* dataset derives from a fictitious Bayesian network modelling the emissions from a waste incinerator shown in Figure A.46 [Lau92]. The variables correspond to the burning regimen B, the state of the electrofilter F, the composition of waste W, the carbon dioxide concentration C, the filter efficiency E, the metal concentration of incoming waste M_{in}, the penetrability of light L,

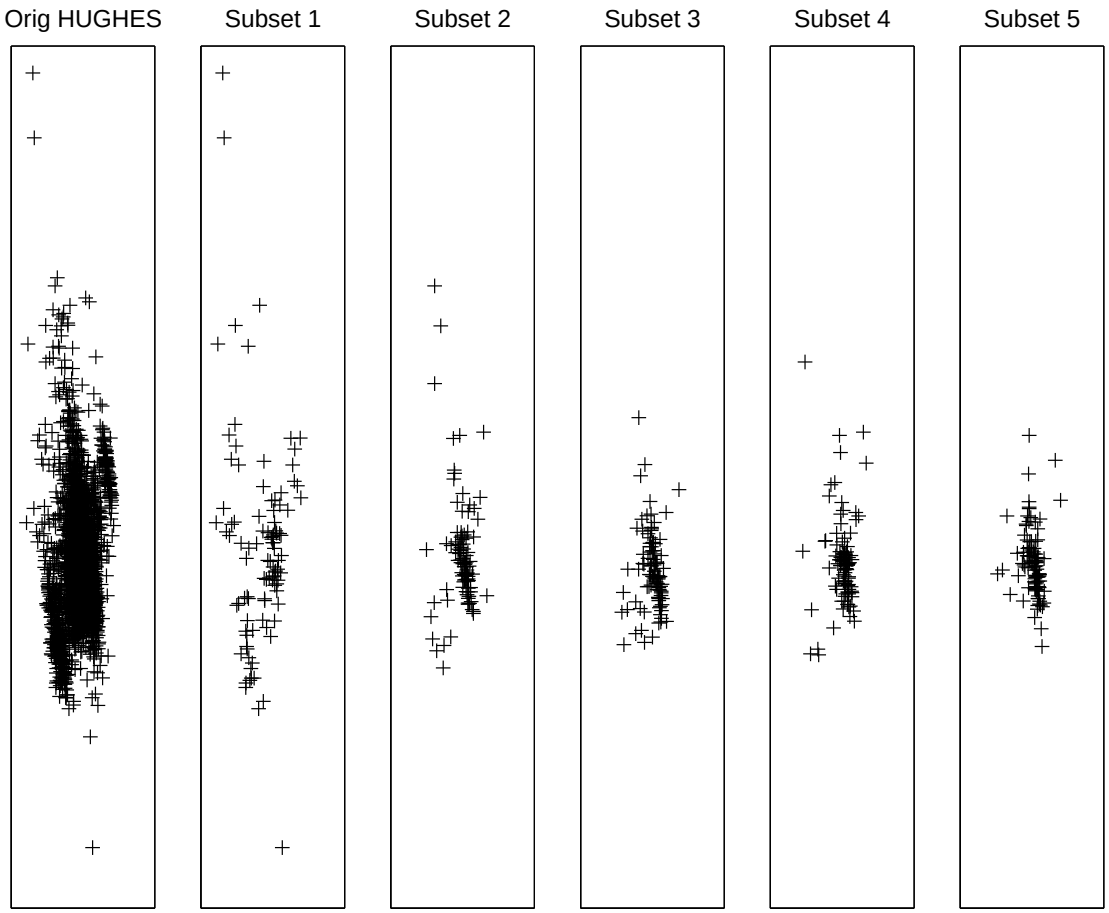


Figure A.45: PCA Projection of the five HUGHES subsets

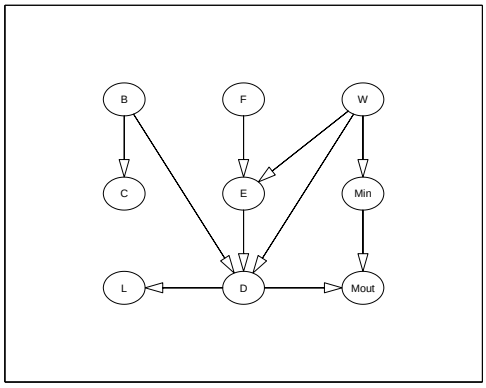


Figure A.46: CG1 Bayesian Network

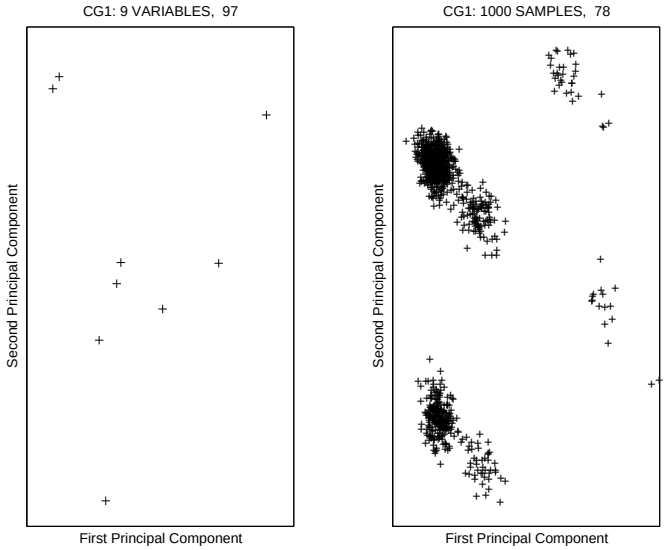


Figure A.47: PCA Projection of the CG1 dataset

the emission of dust D, and the metal concentration of emissions M_out. The variables F, W and B are discrete while the others are samples from a conditional gaussian distribution. During the discretization process, the discrete variables are left as is. We sample 1000 values for each of the 9 variables according the joint probability distribution encoded by the Bayesian network. We refer to this dataset at *CG1* and include it as an example of a typical learning problem where there are a larger number of samples over a small set of variables.

Figure A.47 illustrates the PCA projections for this dataset. Note in the left-side plot that 97 percent of the variance over the nine variables is explained by two principal components. The clustering of the samples apparent from the right-side plot is perhaps due in part to the finite number of configurations of the system depending on the setting of the discrete variables.

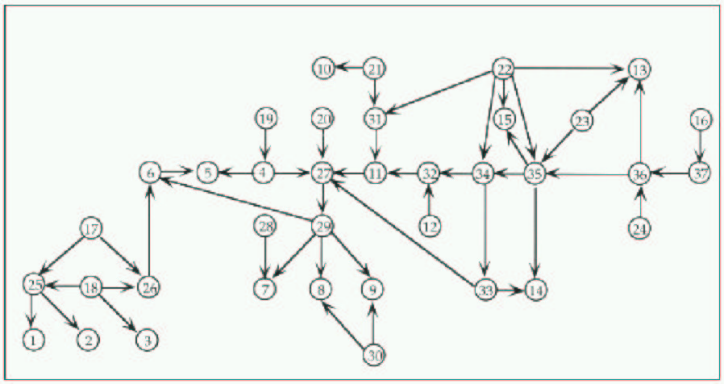


Figure A.48: ALARM Bayesian Network

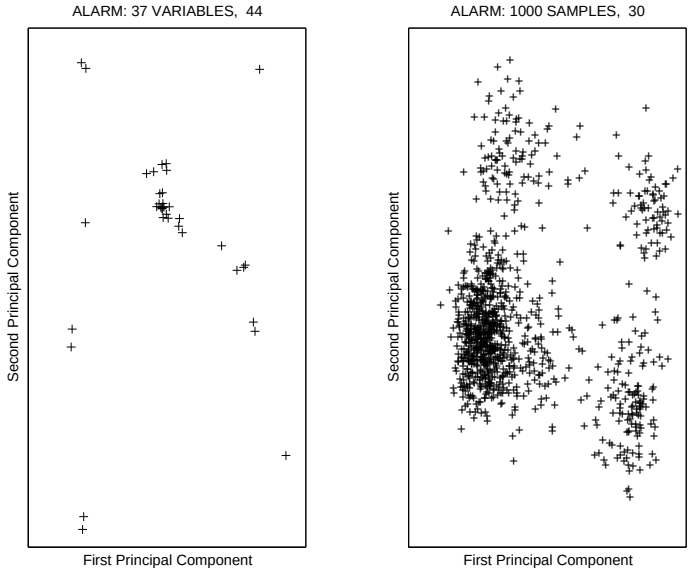


Figure A.49: PCA Projection of the ALARM dataset

Patient Monitoring Bayesian Network Data (ALARM)

The *ALARM* dataset is generated from a real-world example of a Bayesian network designed to model a medical diagnostic alarm message system for ICU patient monitoring [BSCC89]. The Bayesian network over 37 variables is shown in Figure A.48. The variables in the original network are discrete. To obtain a continuous dataset for discretization, we generate 1000 samples of the variables from the joint probability distribution and then add a small amount of noise to the discrete values. Figure A.49 shows the PCA projection of the data. Note the clustering of samples in the right-side plot, indicative of the noise added to the discrete values from the dataset sampled from the network. We include this dataset as an example due to its popularity as a benchmark in many Bayesian network learning tasks.

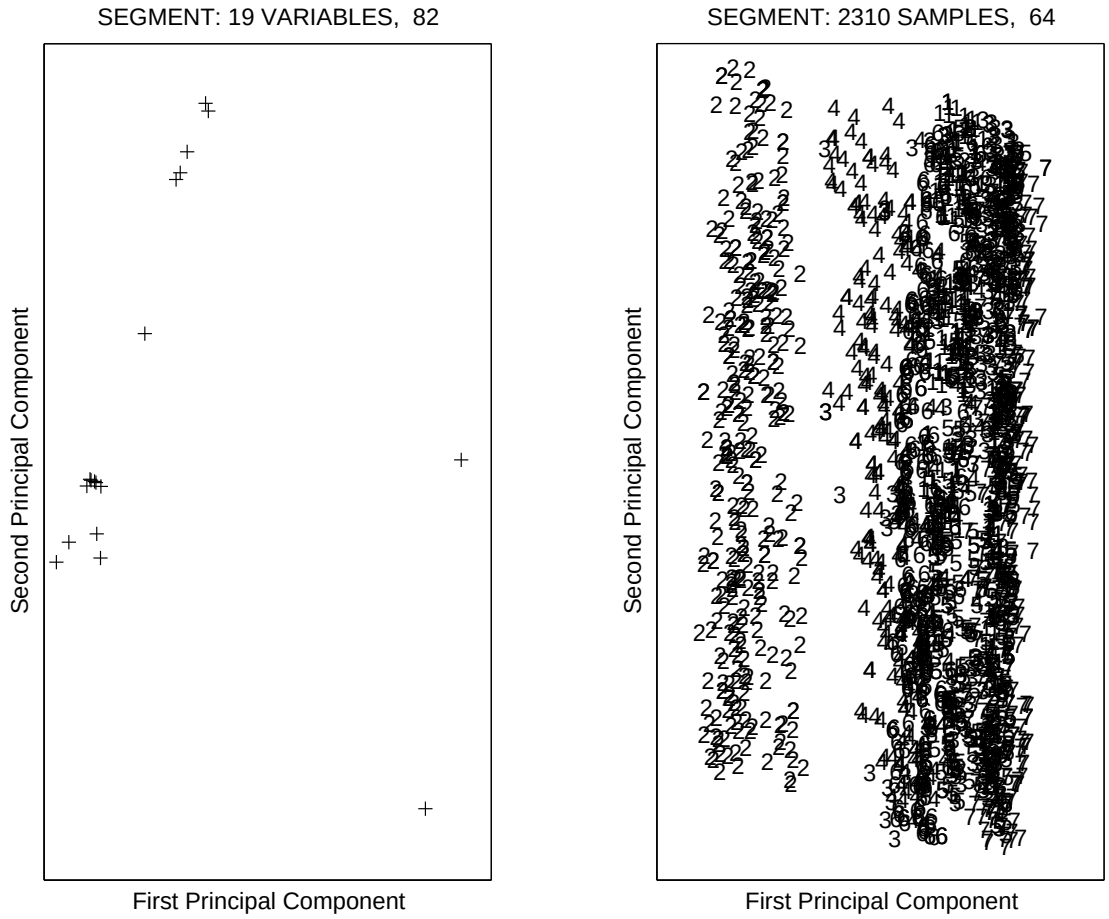


Figure A.50: PCA Projection of the SEGMENT dataset

Image Segmentation Data (SEGMENT)

The *SEGMENT* dataset derives from the task of classifying pixels in an image as either brickface, sky, foliage, cement, window, path, or grass based on information about pixel location, color, hue, value and other attributes. This dataset is available from the UC-Irvine Machine Learning Datasets Repository [BM98]. The dataset is comprised of 2310 hand-classified samples over 17 continuous attributes and 3 discrete attributes (including the classification variable). The PCA projection of the data is shown in Figure A.50. As can be seen from the right-side plot, the samples have been labelled by the integers 1 through 7, with the samples having value 2=sky forming a distinct vertical cluster to the left. This example is included to demonstrate a typical multi-class machine learning dataset.

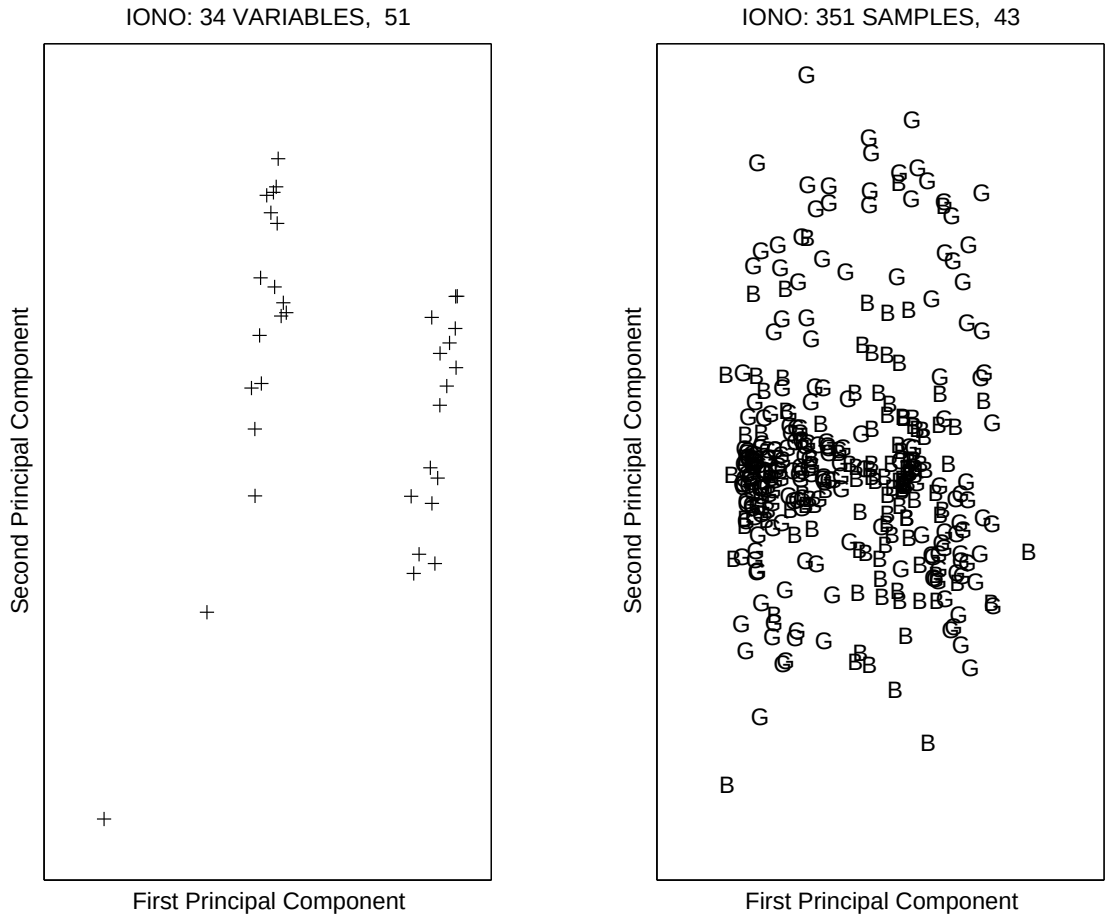


Figure A.51: PCA Projection of the IONO dataset

Radar Returns from the Ionosphere Dataset (IONO)

The *IONO* dataset derives from radar data transmitted into the ionosphere for the purpose of identifying 'Good' and 'Bad' signals based on whether they passed through the ionosphere or not. This dataset can be found in the UC-Irvine Machine Learning Repository [BM98]. There are 351 samples over 33 continuous attributes plus one discrete classification attribute. The original dataset had 34 continuous attributes which represented a two dimensional representation of 17 distinct radar pulses; one attribute was removed because its value was constant across samples. The PCA projection is shown in Figure A.51. In the left-side plot of variables, there are two distinct vertical clouds, perhaps indicative of the two-dimensional representation of the radar pulses. The samples, shown in the righ-side plot, are labelled with 'G' and 'B' for 'Good' and 'Bad' respectively. Note that the classification is not readily separable as in previous examples of classified samples. We include this dataset as a typical example of a binary classification problem.

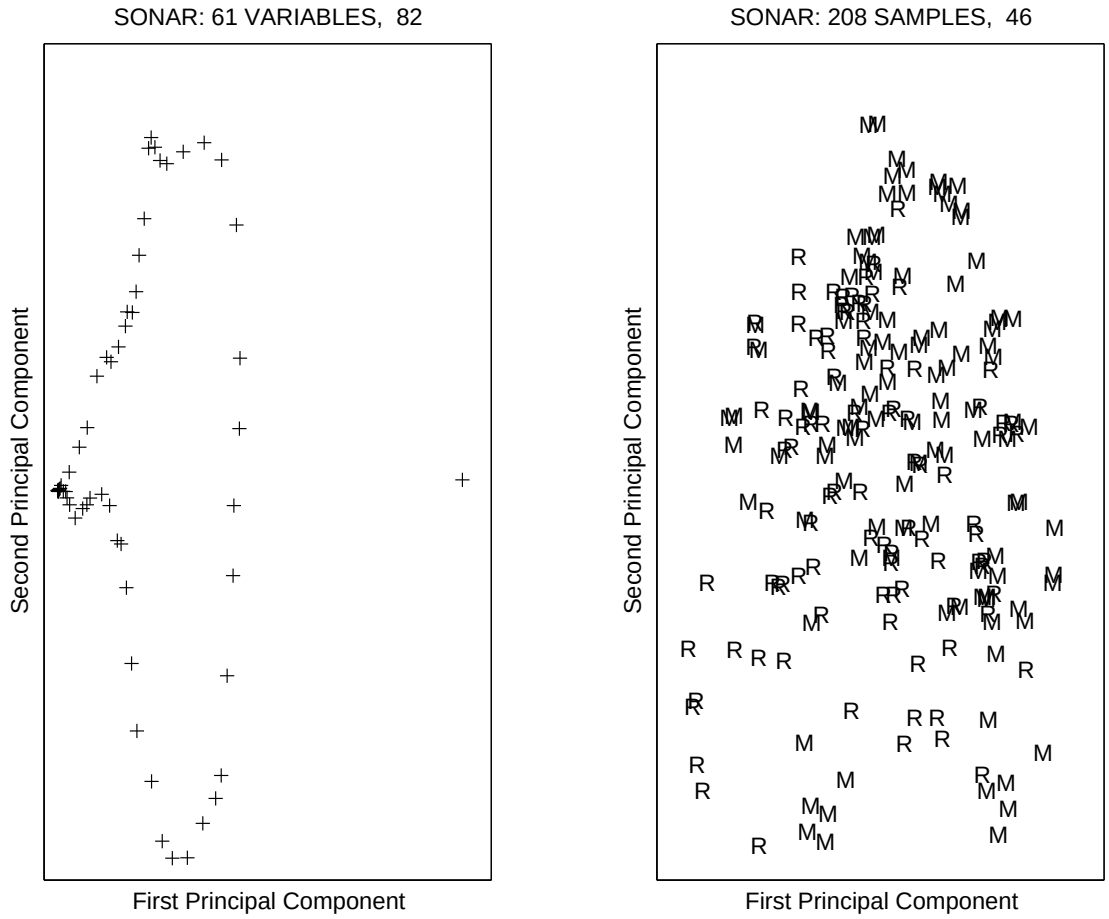


Figure A.52: PCA Projection of the SONAR dataset

Sonar Classification Dataset (SONAR)

The *SONAR* dataset derives from the task of classifying an object as metal or rock by detecting deflection of sonar signals for the purpose of identifying landmines. The dataset is available from the UC-Irvine Machine Learning Repository [BM98]. The data consists 60 continuous valued attributes and a binary classification attribute, measured for 208 samples, distributed as 111 for metal and 97 for rocks. The 60 attributes are numbers in the range 0.0 to 1.0 and encode the value of the energy within a particular frequency band, integrated over a certain period of time. Figure A.52 depicts the PCA projection of the data, where the frequency nature of the variables is clearly distinguishable from the left-most plot. The 60 sonar attributes form a distorted circle to the left while the classification attribute rests as a single point on the right in the plot. In the right-side plot, the samples are labelled as 'M' and 'R' for metal and rocks respectively. Note that the sample identities are not clearly separated in this view. We include this example as a typical machine learning dataset with high correlation between variables.

A.3.3 Experimental Parameters

Information is lost by discretizing, hence an important consideration is choosing the level of granularity of discretization. An incorrect choice for the number of categories can obscure important relationships evident in the original data. Ideally, one would optimize the number of bins over all measures; this is clearly impractical for a study of this magnitude. Therefore we choose a representative sample of values for the target number of bins. Many of our datasets contain very few samples which limits the number of effective discretization levels, thus we choose a target of three and a target of five bins for comparing all datasets. An odd number of bins is chosen since many of our datasets derive from gene expression data where the datapoints are ratio values, which when discretized have intuitive meanings of under-expression, average expression, over-expression and the like. For datasets with more than 80 samples, we also compare discretization with a target of 10 bins.

For many of the methods, several parameters must be specified. Let m be the number of samples in the dataset. For the methods TMIDLC_Q, TMIDLC_I, IDRDLC, SIDRDLC, PDRDLC, and SPDRDLC which use the Discretization Level Coalescence (DLC) algorithm of Section A.1.1, the number of bins in the initial fine-grained discretization must be specified. We choose to use $\min(50, m)$ as the initial granularity. Also, the DLC methods must specify which algorithm to use for the initial discretization. The TMIDLC methods use either quantile (*i.e.,* EqFreqInt) or interval (*i.e.,* EqIntWid) depending on the suffix (Q or I) while the (S)IDR and (S)PDR variants use EqIntWid. The RUDE procedures, RUDE_F and RUDE_C, use EqIntWid as an initial discretization procedure, where the target number of bins is set to $\min(100, m/2)$. The MVD discretization procedure uses an initial discretization of $\min(100, m)$ together with EqIntWid, unless the result is degenerate for a variable, in which case EqFreqInt is used for that variable. The significance level $\alpha = 0.05$ and the minimum deviation `mindev`$= 0.01$ for MVD. For the Restart procedures (IDRRestart, SIDRRestart, PDRRestart and SPDRRestart), we use 10 random restarts, where the bin boundaries of the first restart are set to those for EqIntWid, unless degenerate in which case all 10 restarts begin at random bin boundary assignments.

For the comparisons, the Matlab©programming environment (`http://www.mathworks.com`) was used to implement all discretization techniques except IDRFull, IDRSplit, PDRFull and PDRSplit which instead were implemented in C++ using the Standard Template Library (STL) from Silicon Graphics Incorporated (`http://www.sgi.com/tech/stl/`). Code for computing the wavelet evaluation metric used the WaveLab toolbox (`http://www-stat.stanford.edu/ wavelab`) for Matlab.

A.4 Results

In this section, we discuss the results of applying eight evaluation metrics for 31 discretization techniques on 14 datasets. Relative performance of all metrics was fairly consistent across all datasets and the graphs for $N = 5$ are nearly the same as the $N = 3$ equivalents. For this reason, we include a representative in Figure A.53 for $N = 3$ and defer the full set of results to Appendix B and C.

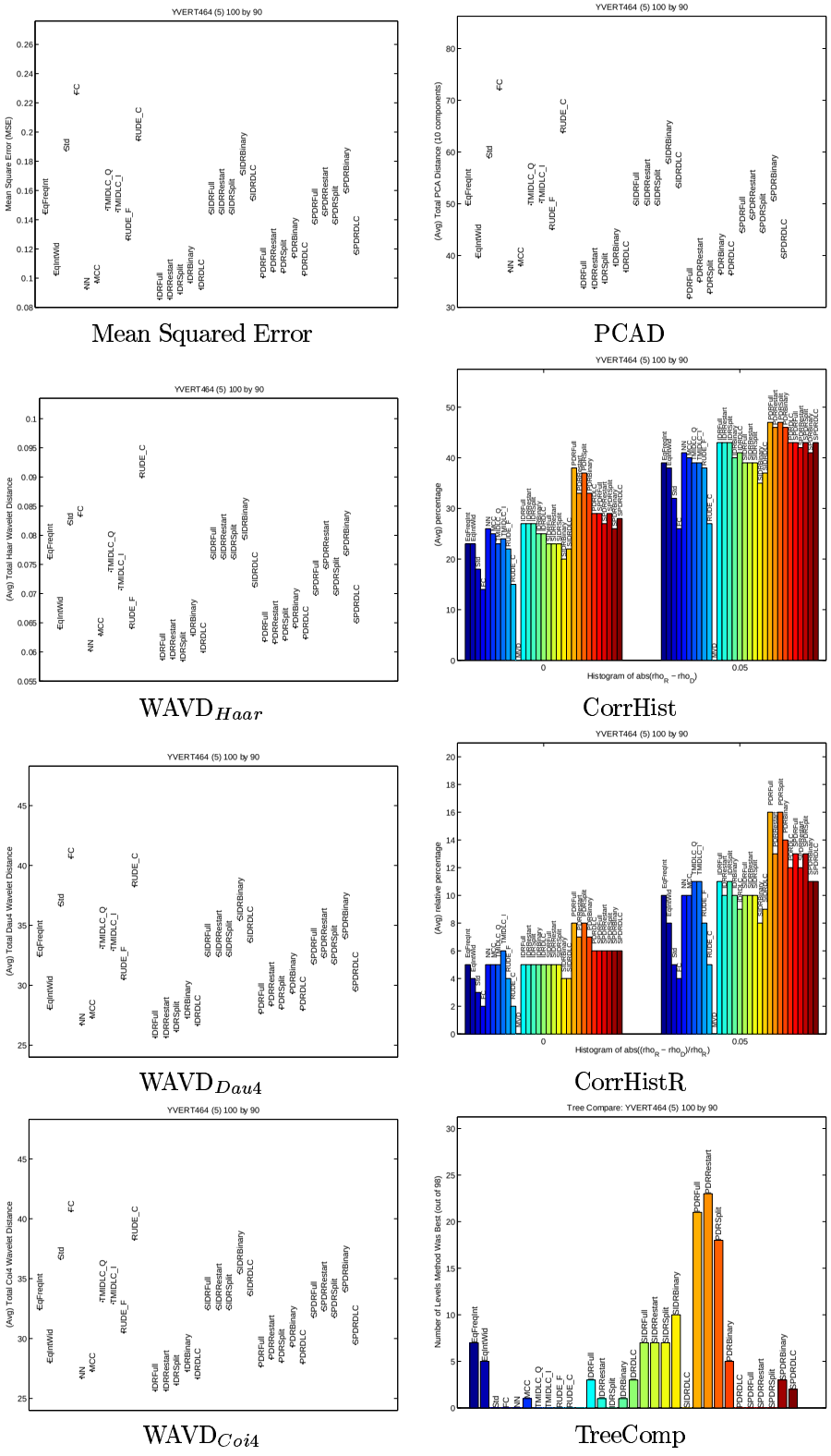


Figure A.53: Example of Evaluation Metrics using YVERT464 Dataset

Figure A.53 provides an example to refer to for all the discussion points which follow. The figure shows each metric in turn, namely Mean Squared Error, WAVD, PCAD, CorrHist, CorrHistR, and TreeComp. For the Mean Squared Error, WAVD, and PCAD metrics, better performance is indicated by values lower on the vertical scale. For CorrHist, CorrHistR and TreeComp, better performance is indicated by higher bars. For CorrHist and CorrHistR, we are particularly interested in the height of the bars over value 0 on the horizontal axis. The bars at 0.05 is used to break ties. For ease of comparison, the ordering of the 31 methods, from left to right, remains the same in each plot. Also, for CorrHist, CorrHistR, and TreeComp, the coloration of the bars remains the same across plots. For example, EqFreqInt is always shown on the extreme left, with dark blue bars where appropriate, while SPDRDLC is always shown on the extreme right with burgundy bars where appropriate. The labelling order and color of the TreeComp plot in the lower right of the figure can generally be used as a guide.

For the Mean Squared Error (MSE) and three WAVD metrics, the results generally show the IDR variants as the strongest performers, followed by NN, then MCC, with EqIntWid and the PDR variants vying for fourth place. The Spearman variants of IDR and PDR are generally among the worst performers, with the Spearman Binary methods consistently showing the lowest performance even relative to the other Spearman methods. RUDE_C and Std also perform poorly on these metrics.

For the MSE, the operations on the datamatrix are performed independently on each cell of the matrix, while for the WAVD metrics, the datamatrix is convolved in the time and frequency domain before computing distances. The strength of IDR over PDR suggests that these metrics are highly sensitive to preserving the correlation of the independent rows of the datamatrix. For the WAVD metrics, this may in part be an artifact of the process of computing the wavelets first on the rows of the matrix, then on the columns. In contrast, for the PCAD where the datamatrix is transformed instead in geometric space, the PDR variants outperform the other metrics, followed closely by the IDR variants, then NN, with MCC and EqIntWid generally sharing fourth place. The objective of preserving the joint correlational structure among the rows of the datamatrix using PDR appears to offer an advantage over IDR for this metric. Again, the Spearman variants, RUDE_C and Std are the worst performers.

The three metrics CorrHist, CorrHistR and TreeCompare rely heavily on computing pairwise correlations, thus it is not surprising that the PDR variants emerge as the strongest performers. For CorrHist and CorrHistR, the order of performance is similar to the PCAD metric: PDR followed by IDR, closely met with NN, MCC and EqIntWid. The Spearman IDR variants correspondingly perform worse than the Spearman PDR variants.

Many interesting dimensions are investigated in this study. We mentioned above the results for using Spearman versus Pearson correlation for the IDR and PDR variants. Several other questions may be asked, as summarized below.

- **Spearman versus Pearson Correlation**: The Pearson variants of IDR and PDR consistently outperform their Spearman counterparts on all datasets, under all metrics.

- **Ratio vs Non-ratio Datasets**: Though generally the Spearman variants perform worse than Pearson, there are larger margins between their performance and the next worst performers in the non-ratio datasets (CF, ADENO, CG1, ALARM, IONO, SEGMENT and SONAR) relative to the ratio datasets. In fact, the Spearman methods are not among the worst for WAVD_{Haar} on the non-ratio datasets, rather they are comparable to NN, MCC and PDR (with the exception of the Spearman Binary versions). The non-parametric nature of Spearman correlation, combined with the square, almost discontinuous waveform of the Haar wavelet appears to compliment non-ratio data. Spearman methods also see better performance under the CorrHist and CorrHistR on non-ratio data relative to the ratio datasets. Interestingly, the DLC variants (TMIDLC, IDRDLC, PDRDLC), which generally perform poorly, also see increased performance in the TreeComp metric for the non-ratio datasets versus the ratio datasets.

- **Variations of the WAVD Metric**: For the ratio datasets, the relative performance of all 31 methods is identical for the Haar, Dau4 and Coi4 metrics, thus proving invariant to the choice of wavelet. For the non-ratio datasets, as mentioned above, the Haar wavelet stands out with respect to the Spearman variants. The Dau4 and Coi4 remain identical.

- **EqFreqInt versus EqIntWid**: Comparing EqFreqInt to EqIntWid for all eight metrics, generally EqIntWid shows better performance. The exception is WAVD_{Haar} for non-ratio datasets where more often EqFreqInt slightly outperforms EqIntWid. Recall that TMIDLC_I uses EqIntWid for the pre-discretization step and TMIDLC_Q uses EqFreqInt. Generally, the range-based TMIDLC_I outperforms the quantile-based TMIDLC_Q, though there is at least one dataset in which TMIDLC_Q beats TMIDLC_F in the non-ratio datasets. For the ratio datasets, when TMIDLC_Q performs better than TMIDLC_I for the WAVD_{Haar} metric, it also does for the CorrHist, CorrHistR and TreeComp metrics on the same dataset. From these we conclude that generally the range-based versions are better than the quantile based versions, whether used alone or in conjunction with further processing.

- **RUDE_C versus RUDE_F**: In the ratio datasets, RUDE_F consistently outperforms RUDE_C, and in non-ratio datasets, RUDE_F frequently does better than RUDE_C. However, we find that very frequently RUDE_F fails to find bin-boundaries, in which case the attribute is discretized according to EqIntWid. Thus the increased performance of RUDE_F over RUDE_C is likely due to the observed high performance for EqIntWid discussed above, rather than demonstrating a preference for Fayyad and Irani's technique over CART.

- **CorrHist versus CorrHistR**: For the non-ratio datasets, the distributions of CorrHist and CorrHistR are nearly identical. In the ratio datasets, measuring relative difference of correlation through CorrHistR homogenizes across the different IDR or PDR variants, causing almost

all of the variations to have nearly identical performance. This may be due in part to the fact that lower percentages are reported along the vertical axis, meaning there are fewer small differences (0 or 0.05). If there are fewer items in a bin, there are fewer chances to differ among the variants, thereby causing all alternatives to appear to have roughly the same performance.

- **Full, Restart, Split, Binary or DLC**: We consider the difference in performance in two ways: how well the variants chooses a discretization strategy that matches the one found using the Full version; and how well the variants perform on the eight metrics.

    1. Table A.1 through Table A.4 investigate the ability of the different variants of IDR and PDR to recreate the discretizations for each variable found using the Full version. Included are the Pearson and Spearman variants for both. Each entry in the tables shows the percentage of the number of variables discretized using some variant using exactly the same cutpoints found using the Full version of (S)IDR or (S)PDR. Each row corresponds to a dataset, where the last row shows the average over all datasets. In Table A.1, for IDR we see that the Restart version achieves near perfect accuracy on most datasets (95.3% on average) while the Binary version demonstrates poor performance (20.4% on average). This mirrors our earlier discussion about the smoothness of the objective function landscape where we expect that Restart will almost always obtain a local maxima. In contrast are the results for PDR in Table A.2, where the Split version outperforms the Restart version. This again confirms our earlier investigation of the fragmentation of the objective function landscape where we would expect the Restart version to be frequently trapped at local minima. The accuracy of PDRSplit relative to PDRFull (73.3% on average) is comparable to that of IDRSplit relative to IDRFull (75.1%). The high values for Split are a consequence of the fact that these variants, while they explore a reduced space relative to Full, they consider much more of the space than the other variants. For the Spearman versions of IDR and PDR shown in Table A.3 and Table A.4, respectively, we see that the Split version is most accurate, followed by the Restart version. The Binary and DLC versions show exceptionally poor performance in all tables.
    2. On the eight evaluation metrics, the IDR and PDR variants for non-ratio data are generally ranked as Full, Restart, DLC, Split, and Binary, in decreasing order, where Full and Restart often share rank despite the low fidelity of PDRRestart to discretize exactly as PDRFull. For ratio data, Split outperforms DLC. For PDR, generally DLC performs the best on the MSE and WAVD metrics for non-ratio data, followed by Full, Restart, Split then Binary. This is surprising given that PDRDLC showed only 10.4% correspondence to PDRFull in Table A.2. Perhaps the differences in binning using PDRDLC preserve the Euclidean distance to the real-valued data with respect to each attribute, causing the discretized vectors to look more like their IDR equivalents for which these metrics showed high performance for IDR. For ratio data with IDR or PDR, DLC performance falls below Full, Restart, and Split. On the last four metrics, whether ratio or non-ratio data,

DATASET	Subsets	IDRRestart	IDRSplit	IDRBinary	IDRDLC
CF	5	99.8	72.0	55.0	58.4
CFRatio	5	100.0	82.0	54.0	59.2
ADENO	5	98.4	54.2	27.0	33.2
ADENORatio	5	98.2	75.2	21.6	23.4
SPELLMAN	5	96.2	95.8	5.4	9.4
YVERT464	5	94.2	84.4	7.8	12.2
YVERT465	5	97.4	82.8	8.4	9.4
GASCH	5	89.4	83.6	3.6	2.2
HUGHES	5	83.0	82.0	34.0	12.0
IONO	1	100.0	58.8	5.9	8.8
SONAR	1	91.8	55.7	1.6	3.3
AVERAGE		95.3	75.1	20.4	21.0

Table A.1: Comparison of approximations to IDRFull (N=3) by (average) percentage of variables discretized exactly as by IDRFull.

Split performs better than Restart. Curiously, for the TreeComp metric, the GASCH and HUGHES datasets show PDRSplit outperforming even PDRFull. Regardless of whether the metric is PDR or IDR, or whether the correlation is Spearman or Pearson, the Binary variants perform the worst.

From these two observations, we can conclude that IDRRestart is a nearly equivalent alternative to IDRFull. For PDR, either Split or Restart can be used depending on whether or not the task requires preserving the joint correlational structure, as measured by the last four metrics.

A.5 Conclusions

Based on all metrics tested over a variety of datasets, the Binary variants, the Spearman variants, TMIDLC, RUDE, Std and FC do not perform well. MVD is intractable for all but three datasets and its performance is poor for those examples. In cases where it is impractical to incur the computational expense associated with IDR or PDR, the NN or EqIntWid methods are tractable alternatives, with some loss of performance. However, the computationally inexpensive Restart versions of IDR and PDR show comparable results to the Full versions. Overall, our novel technique IDR emerges as the best performing discretization method for the MSE and WAVD metrics, and the second best for the remaining four metrics, in which case our PDR proves most effective.

DATASET	Subsets	PDRRestart	PDRSplit	PDRBinary	PDRDLC
CF	5	97.6	71.2	47.4	39.0
CFRatio	5	97.6	83.2	42.6	41.4
ADENO	5	61.8	58.0	20.0	12.6
ADENORatio	5	59.8	72.8	16.8	7.0
SPELLMAN	5	21.2	91.2	15.4	1.0
YVERT464	5	16.0	83.0	9.0	0.6
YVERT465	5	13.0	81.2	11.6	1.4
GASCH	5	5.2	81.8	4.4	0.4
HUGHES	5	12.6	78.8	11.6	4.0
IONO	1	14.7	58.8	5.9	5.9
SONAR	1	4.9	45.9	4.9	1.6
AVERAGE		36.8	73.3	17.2	10.4

Table A.2: Comparison of approximations to PDRFull (N=3) by (average) percentage of variables discretized exactly as by PDRFull.

DATASET	Subsets	SIDRRestart	SIDRSplit	SIDRBinary	SIDRDLC
CF	5	0.6	100.0	0.0	9.6
CFRatio	5	1.0	100.0	0.0	11.0
ADENO	5	93.8	100.0	0.0	1.6
ADENORatio	5	92.0	100.0	0.0	1.6
SPELLMAN	5	50.6	100.0	0.0	0.4
YVERT464	5	100.0	100.0	0.0	0.0
YVERT465	5	100.0	100.0	0.0	0.0
GASCH	5	68.0	100.0	0.0	0.2
HUGHES	5	82.0	100.0	0.0	0.0
IONO	1	91.2	100.0	5.9	5.9
SONAR	1	41.0	100.0	1.6	1.6
AVERAGE		65.5	100.0	0.7	2.9

Table A.3: Comparison of approximations to SIDRFull (N=3) by (average) percentage of variables discretized exactly as by SIDRFull.

DATASET	Subsets	SPDRRestart	SPDRSplit	SPDRBinary	SPDRDLC
CF	5	98.2	100.0	26.4	2.4
CFRatio	5	98.6	100.0	23.6	5.8
ADENO	5	63.4	99.4	7.6	1.2
ADENORatio	5	64.2	99.6	9.4	0.6
SPELLMAN	5	28.8	99.8	10.0	0.4
YVERT464	5	20.0	100.0	5.4	0.2
YVERT465	5	21.0	99.4	9.2	0.0
GASCH	5	4.4	99.0	4.4	0.0
HUGHES	5	2.4	99.8	0.6	0.0
IONO	1	8.8	100.0	5.9	5.9
SONAR	1	3.3	100.0	1.6	1.6
AVERAGE		37.6	99.7	9.5	1.6

Table A.4: Comparison of approximations to SPDRFull (N=3) by (average) percentage of variables discretized exactly as by SPDRFull.

Appendix B

Complete Discretization Results (N=3)

This chapter presents the complete discretization results for $N = 3$ bins using eight evaluation metrics for 31 discretization techniques on 14 datasets described in Appendix A. All datasets are shown for each metric in turn, namely Mean Squared Error, WAVD, PCAD, CorrHist, CorrHistR, and TreeComp. Mean Squared Error (MSE) measures the average squared Euclidean Distance between the discrete version of a vector and the real-valued, continuous version. The wavelet distance functions WAVD_{Haar}, WAVD_{Dau4}, and WAVD_{Coi4} measure the Euclidean Distance between the wavelet coefficients of the discrete datamatrix and the real datamatrix using Haar, Daubechies 4 and Coiflet 4 wavelets respectively. PCAD measures the Euclidean Distance between the top 10 principal components of the discrete datamatrix and the real datamatrix. CorrHist provides a histogram of the Euclidean Distance between the vector of pairwise correlations calculated on discrete vectors and the vector of corresponding pairwise correlations calculated using real vectors. CorrHistR is the same as CorrHist except that the difference is scaled by the magnitude of the correlation to provide relative distances. Finally, TreeComp measures the number of levels in the dendrogram computed on discrete data in which a particular method achieves the best performance with respect to preserving the same pairs found in the dendrogram computed on the real data.

For the MSE, WAVD, and PCAD metrics, better performance is indicated by values lower on the vertical scale. For CorrHist, CorrHistR, and TreeComp, better performance is indicated by higher bars. For CorrHist and CorrHistR, we are particularly interested in the height of the bars over value 0 on the horizontal axis. The 0.05 is used to break ties. For ease of comparison, the ordering of the 31 methods, from left to right, remains the same from plot to plot. Also, for CorrHist, CorrHistR, and TreeComp, the coloration of the bars remains the same across plots. For example, EqFreqInt is always shown on the extreme left, with dark blue bars where appropriate, while SPDRDLC is always shown on the extreme right with burgundy bars where appropriate. The labelling order and color of the TreeComp plot in the lower right of each page can generally be used as a guide.

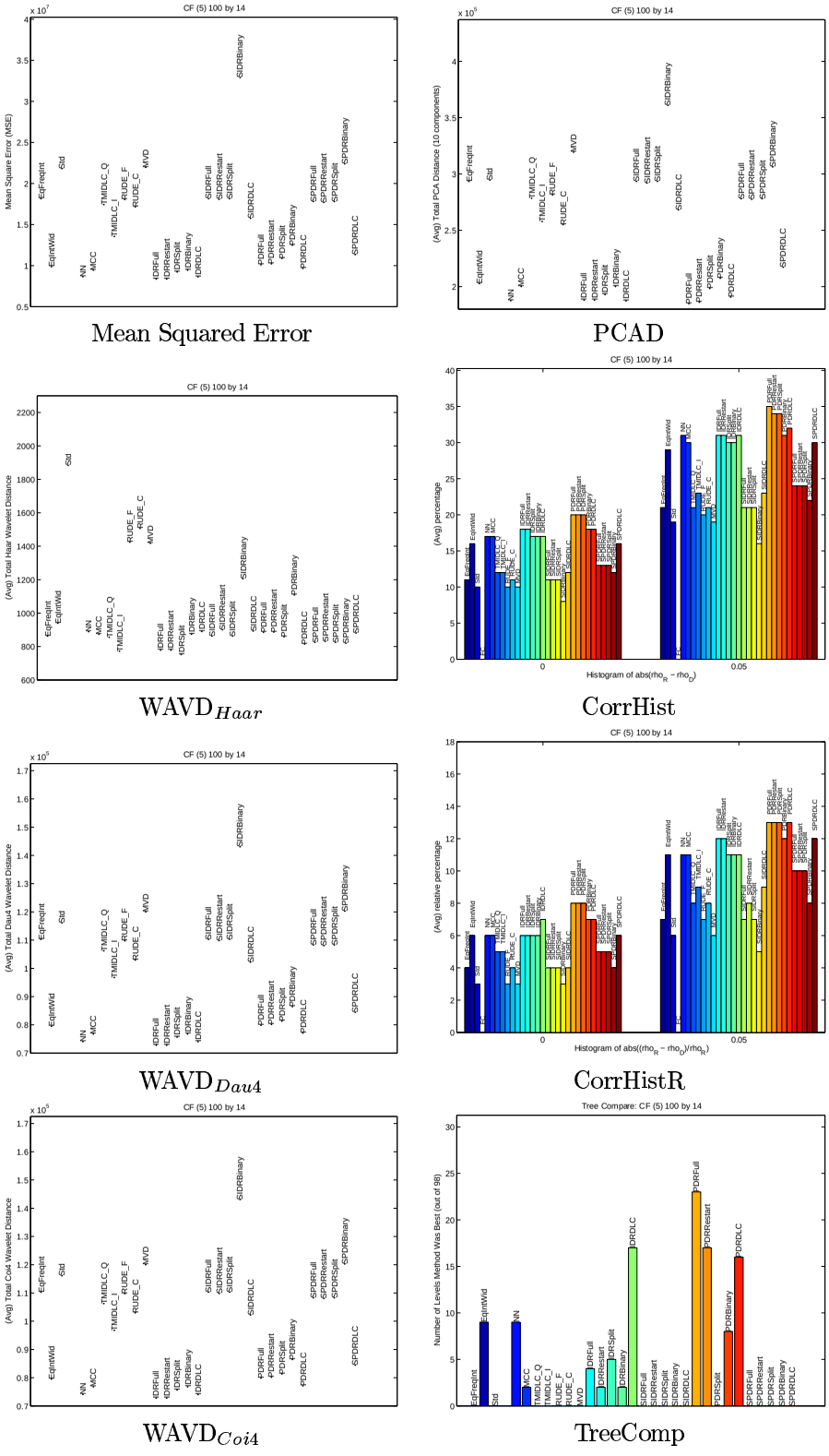


Figure B.1: Metrics on the CF Dataset (N=3)

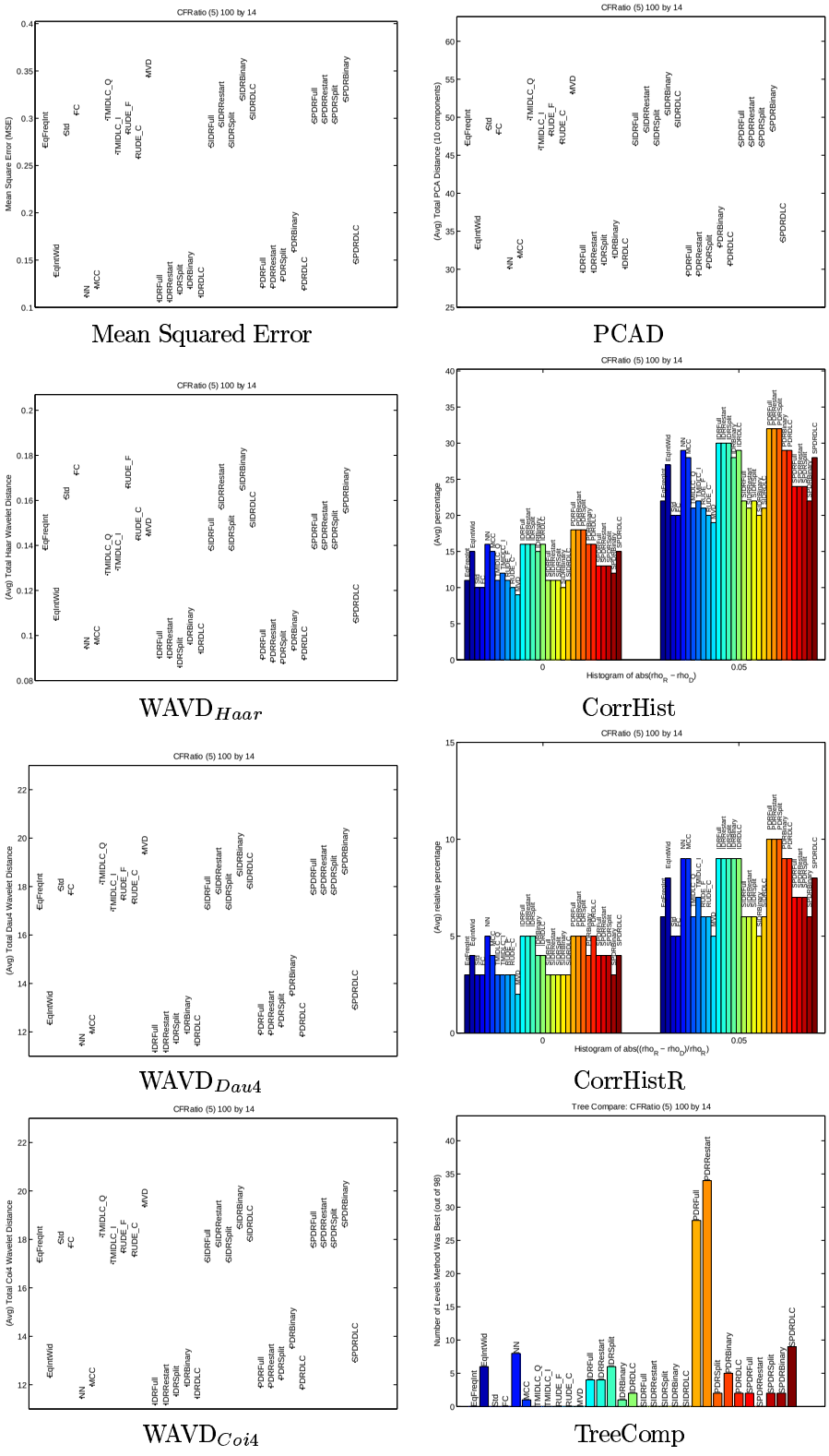


Figure B.2: Metrics on the CFRatio Dataset (N=3)

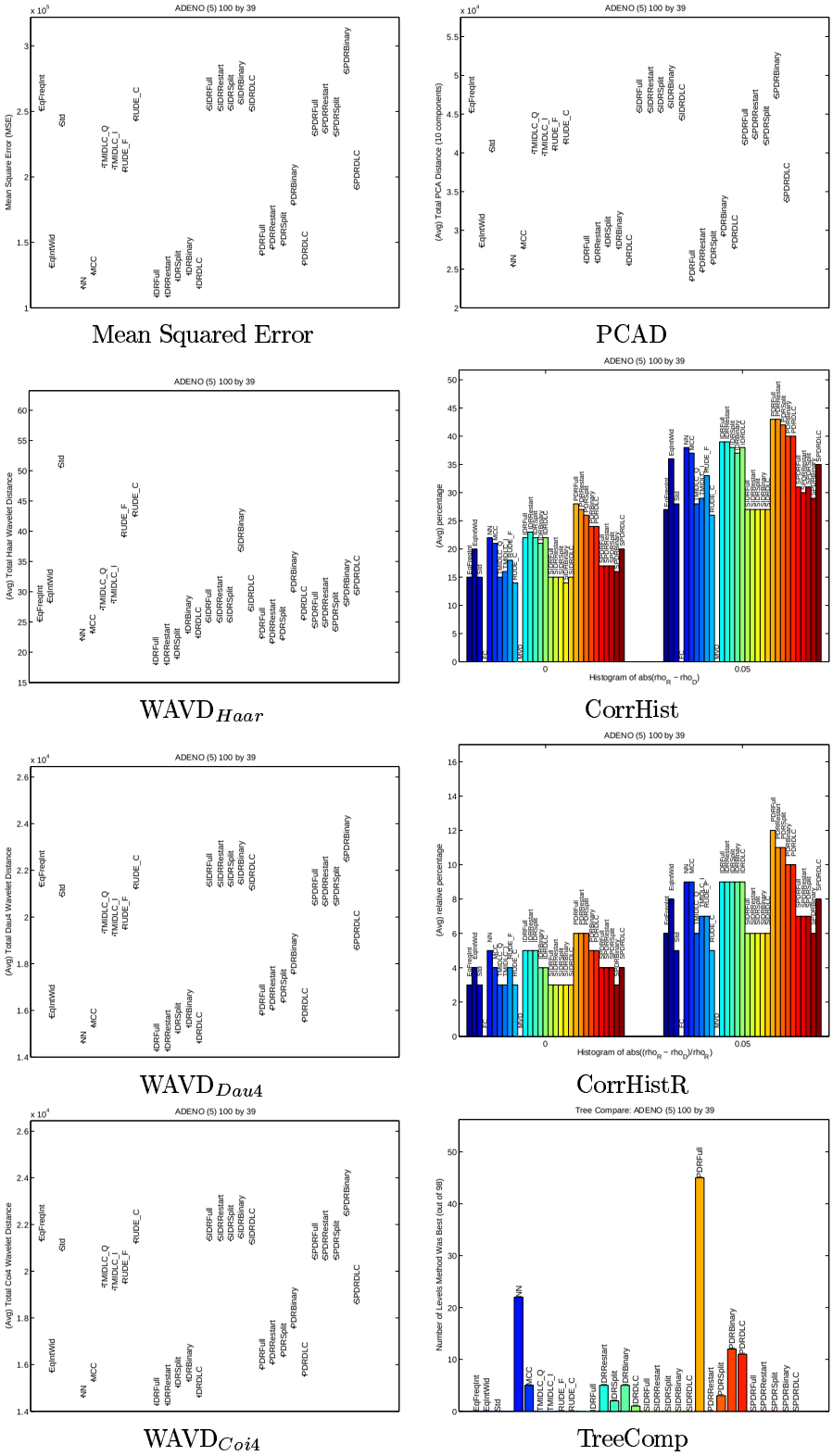


Figure B.3: Metrics on the ADENO Dataset (N=3)

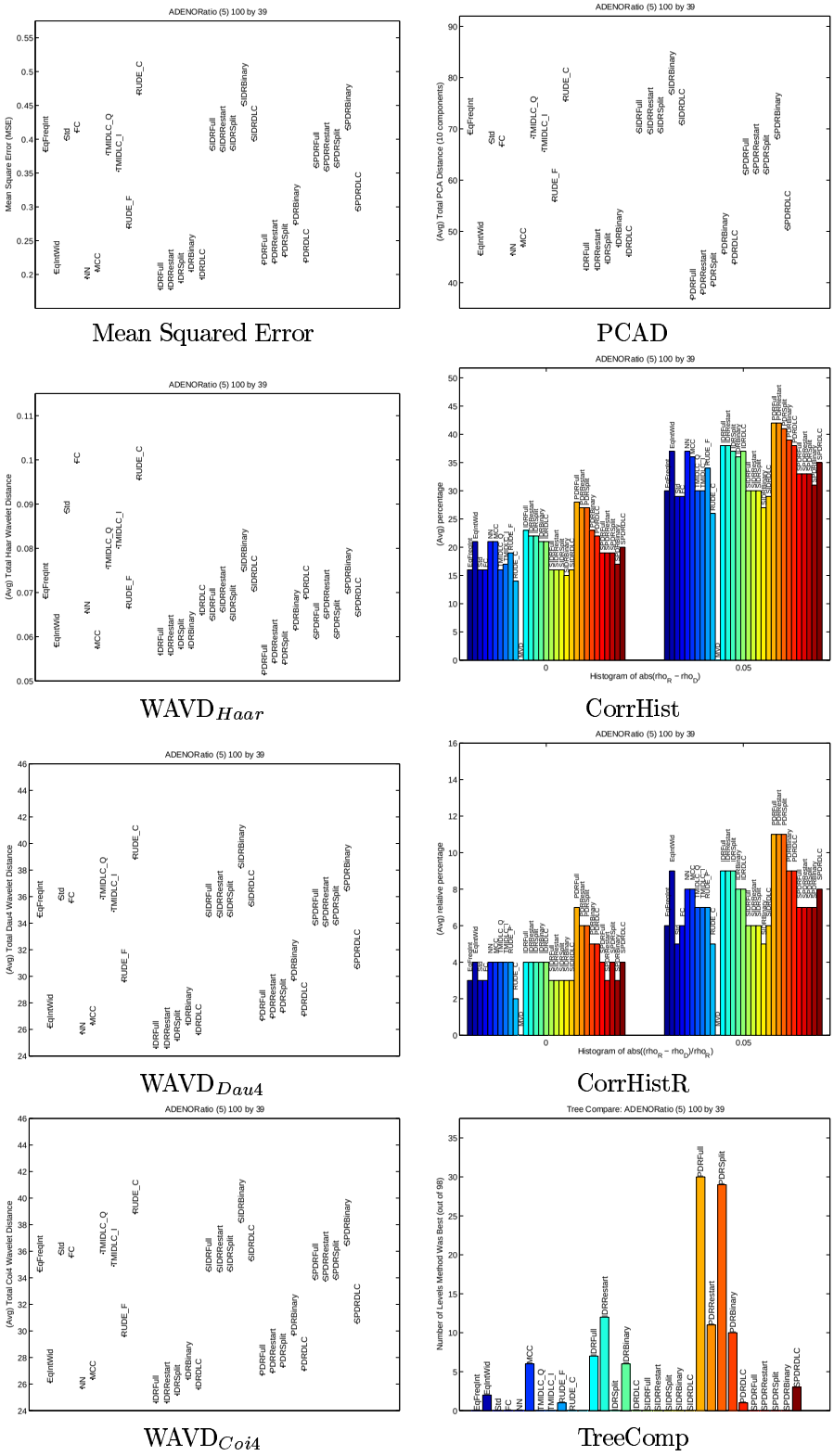


Figure B.4: Metrics on the ADENORatio Dataset (N=3)

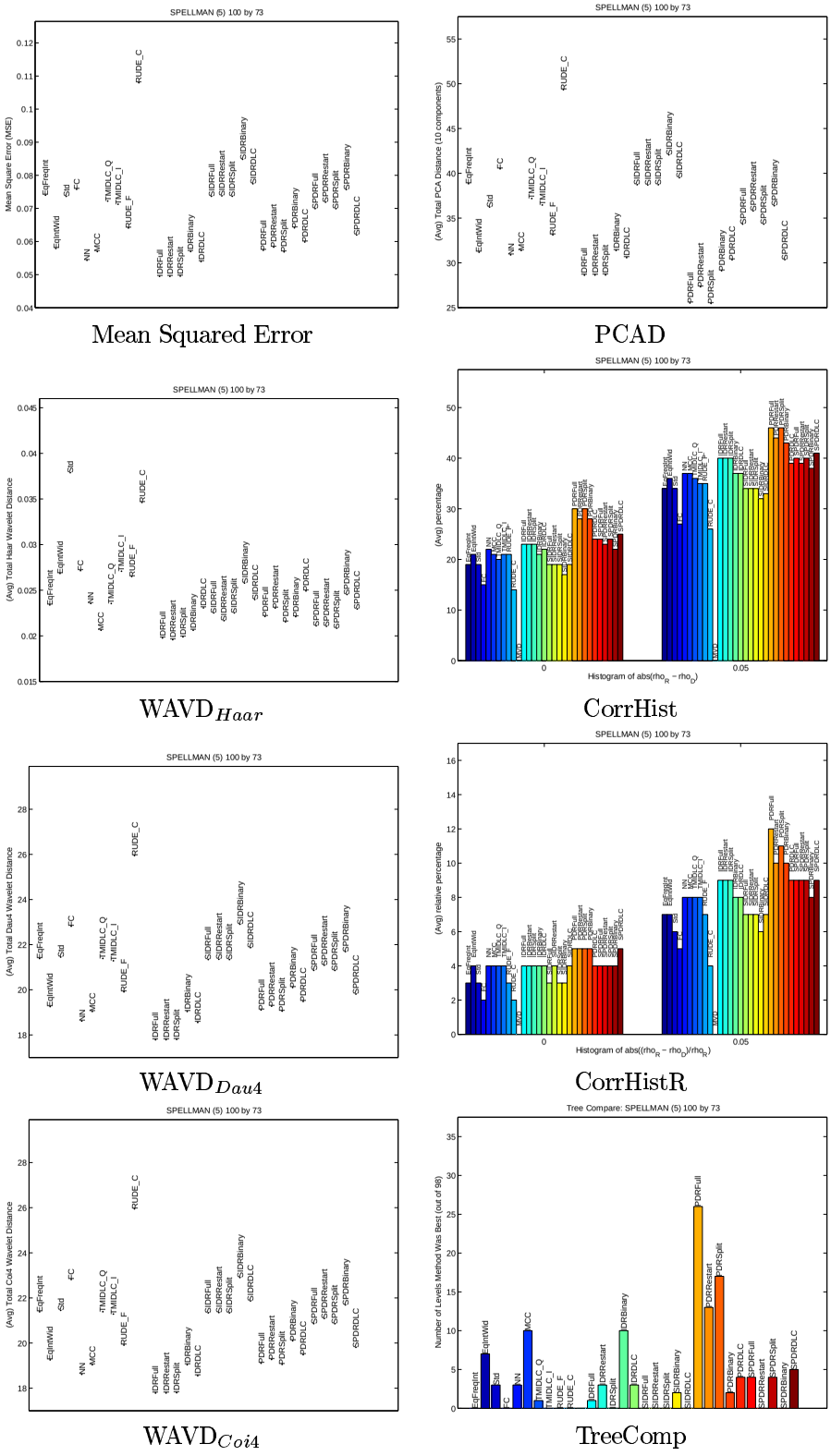


Figure B.5: Metrics on the SPELLMAN Dataset (N=3)

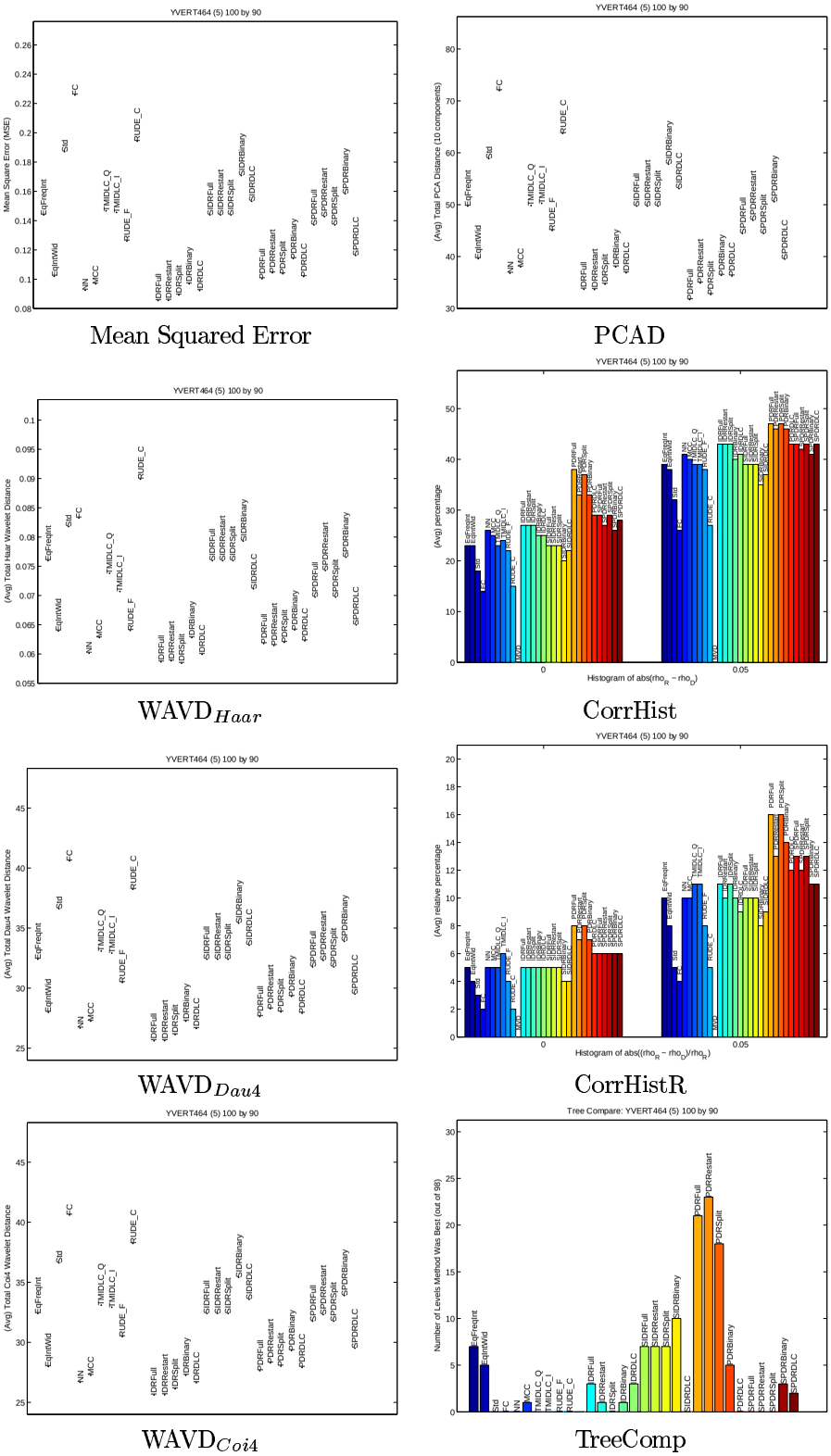


Figure B.6: Metrics on the YVERT464 Dataset (N=3)

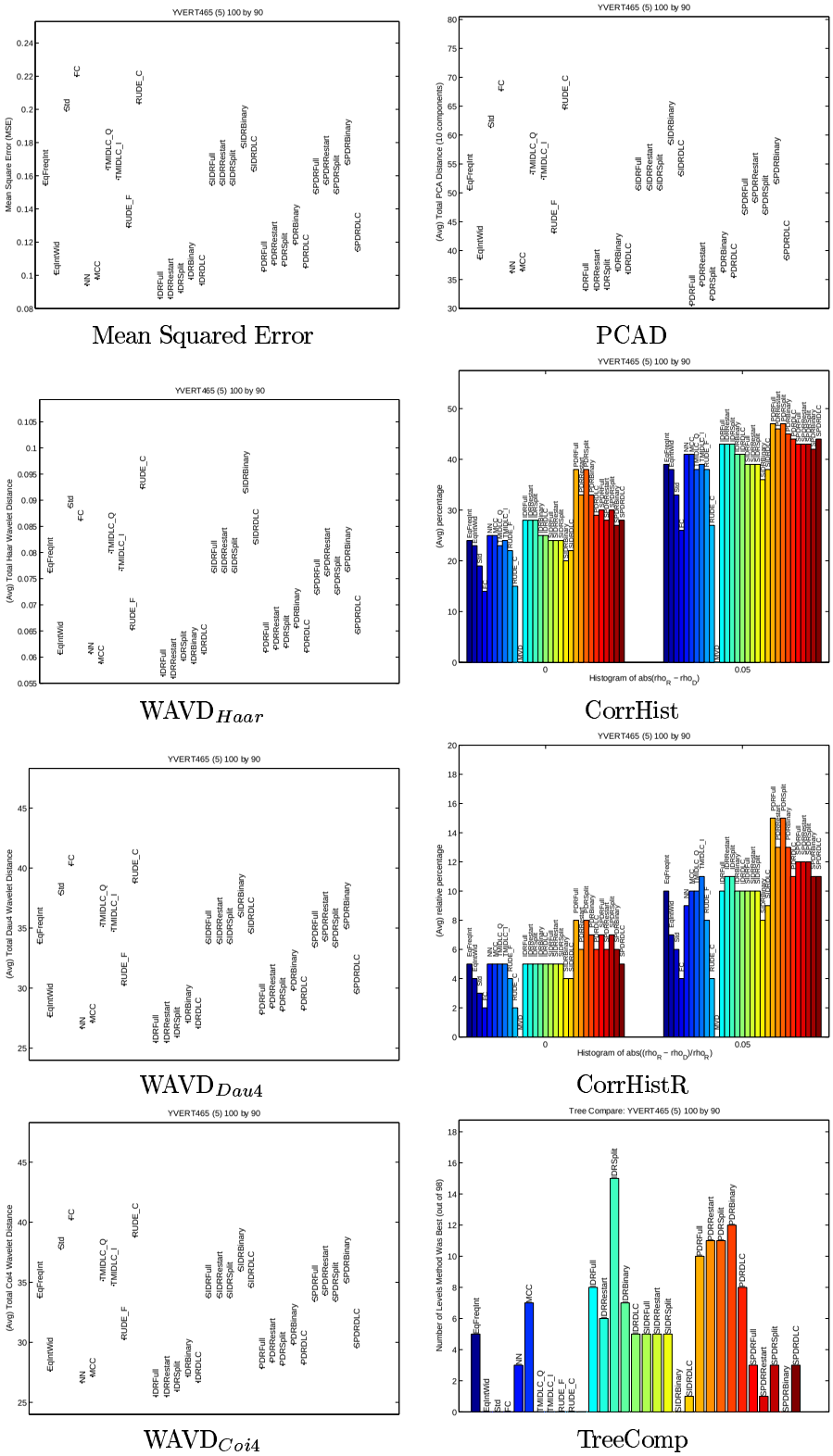


Figure B.7: Metrics on the YVERT465 Dataset (N=3)

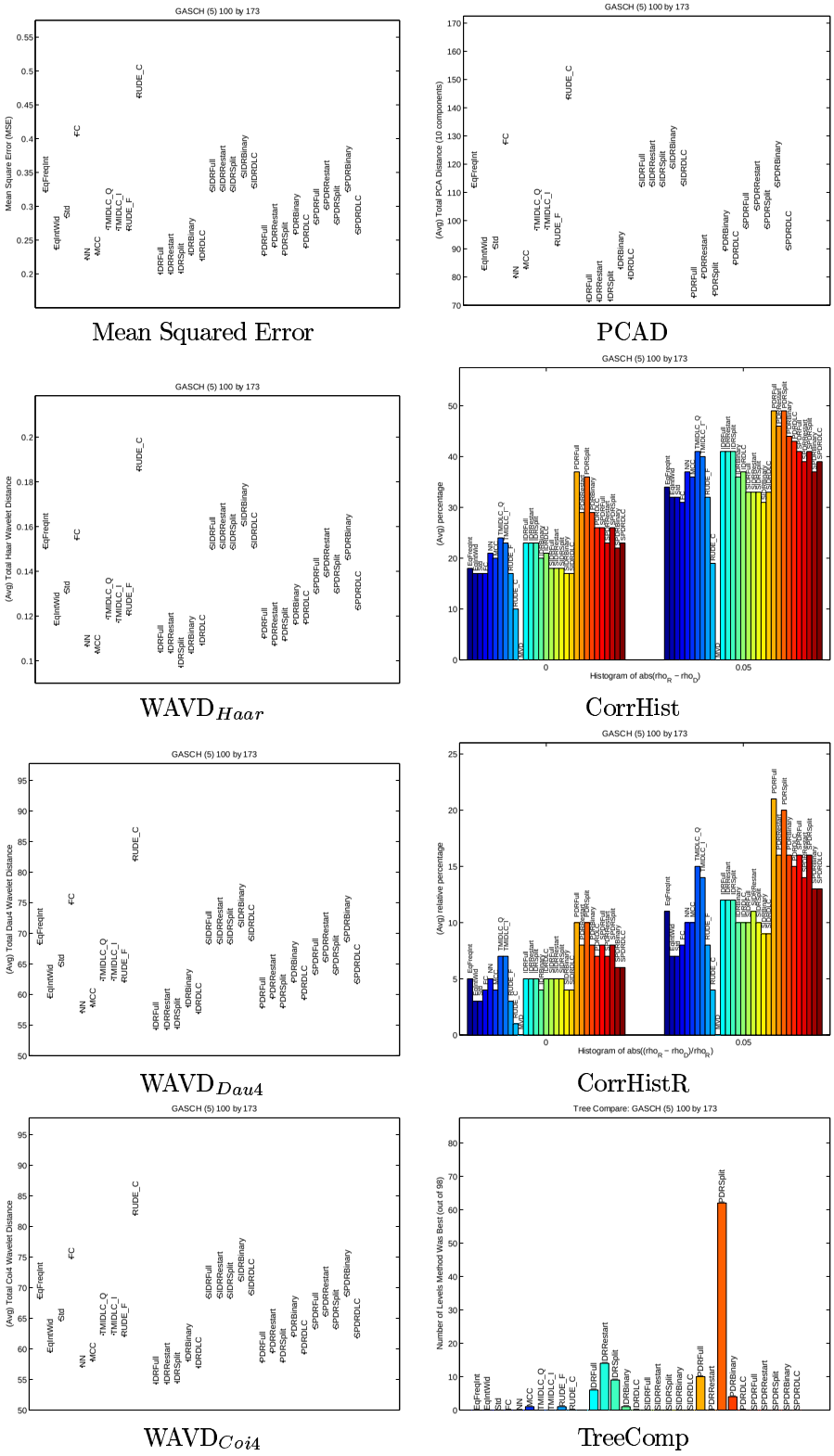


Figure B.8: Metrics on the GASCH Dataset (N=3)

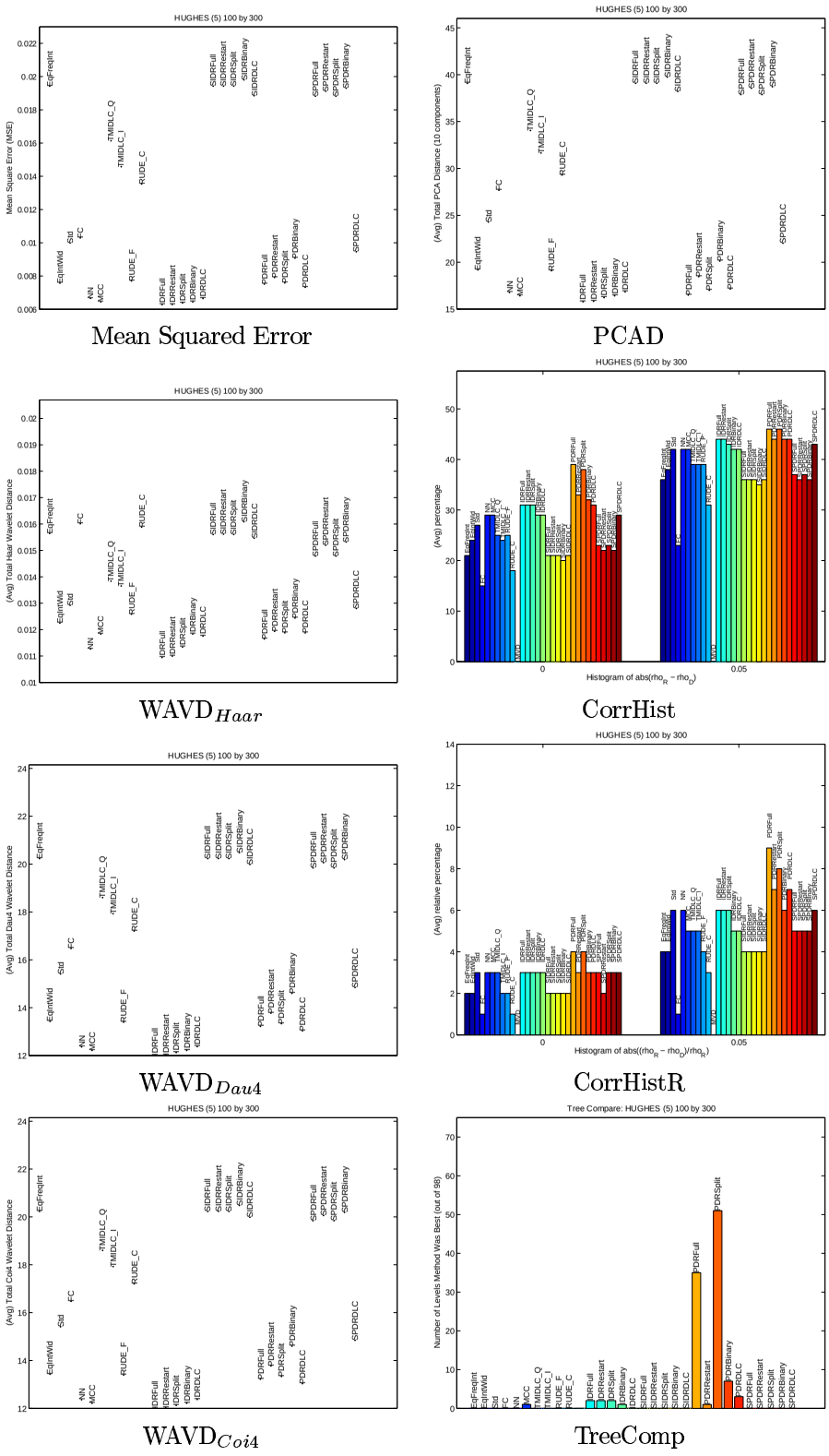


Figure B.9: Metrics on the HUGHES Dataset (N=3)

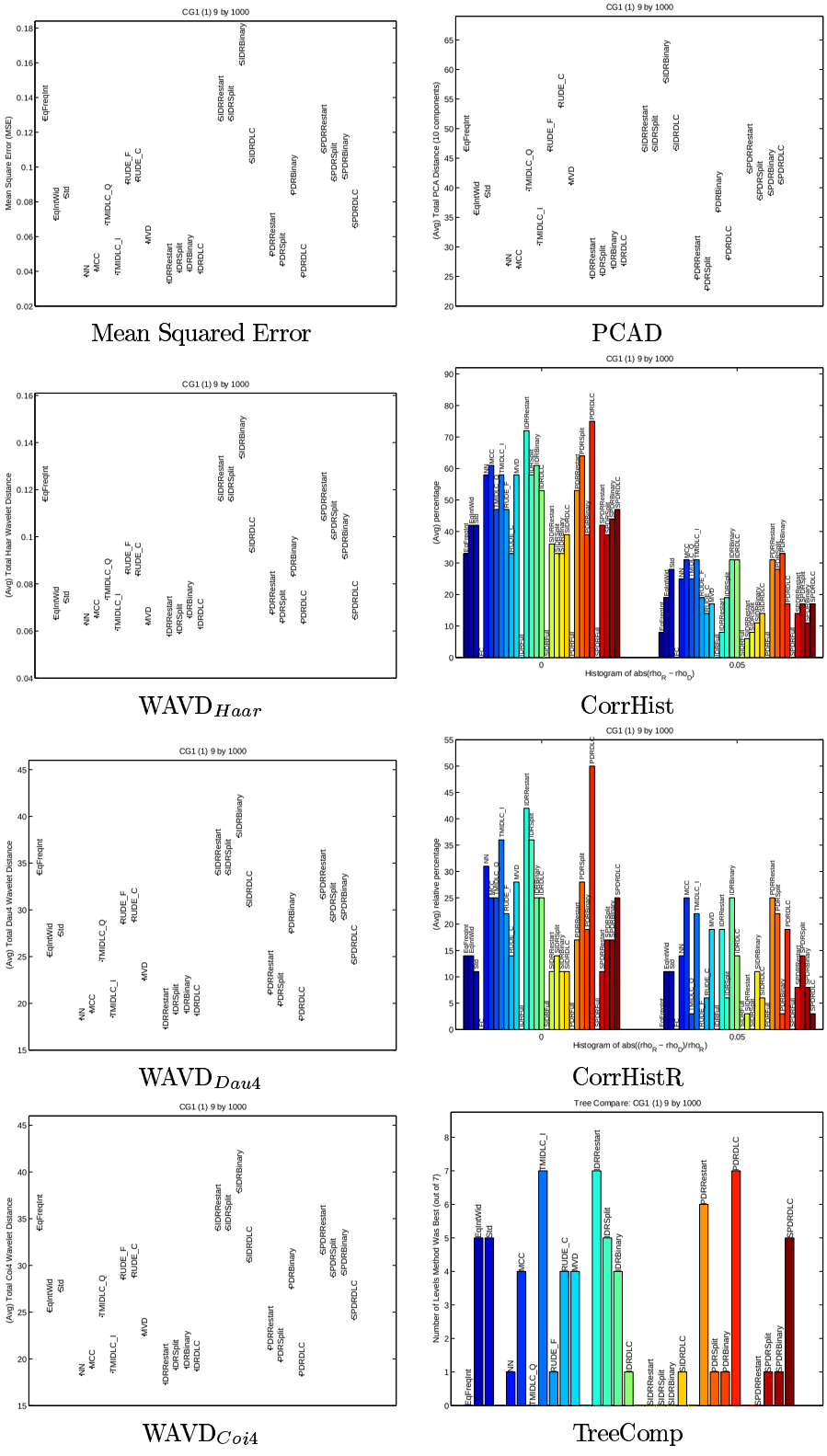


Figure B.10: Metrics on the CG1 Dataset (N=3)

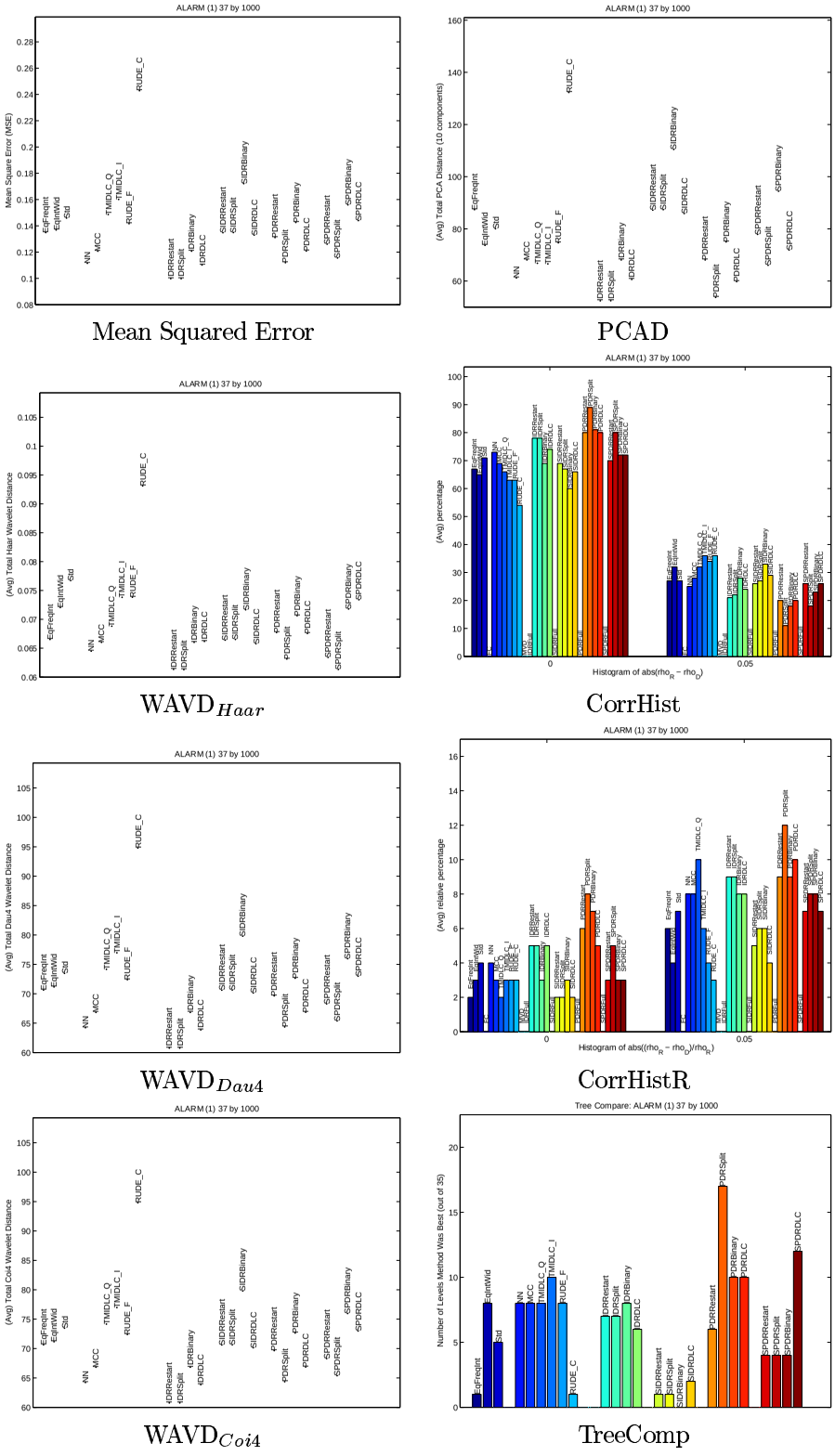


Figure B.11: Metrics on the ALARM Dataset (N=3)

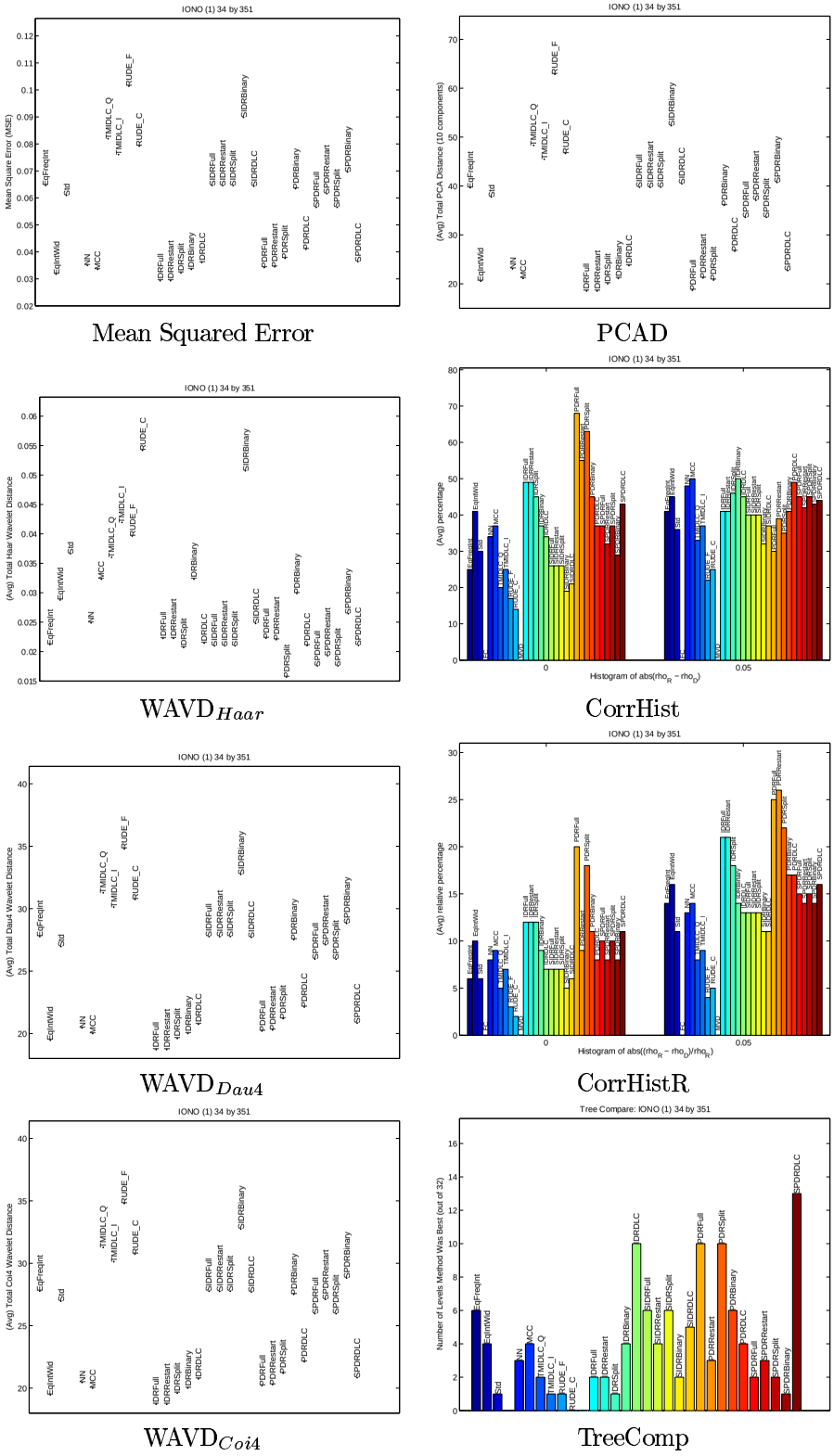


Figure B.12: Metrics on the IONO Dataset (N=3)

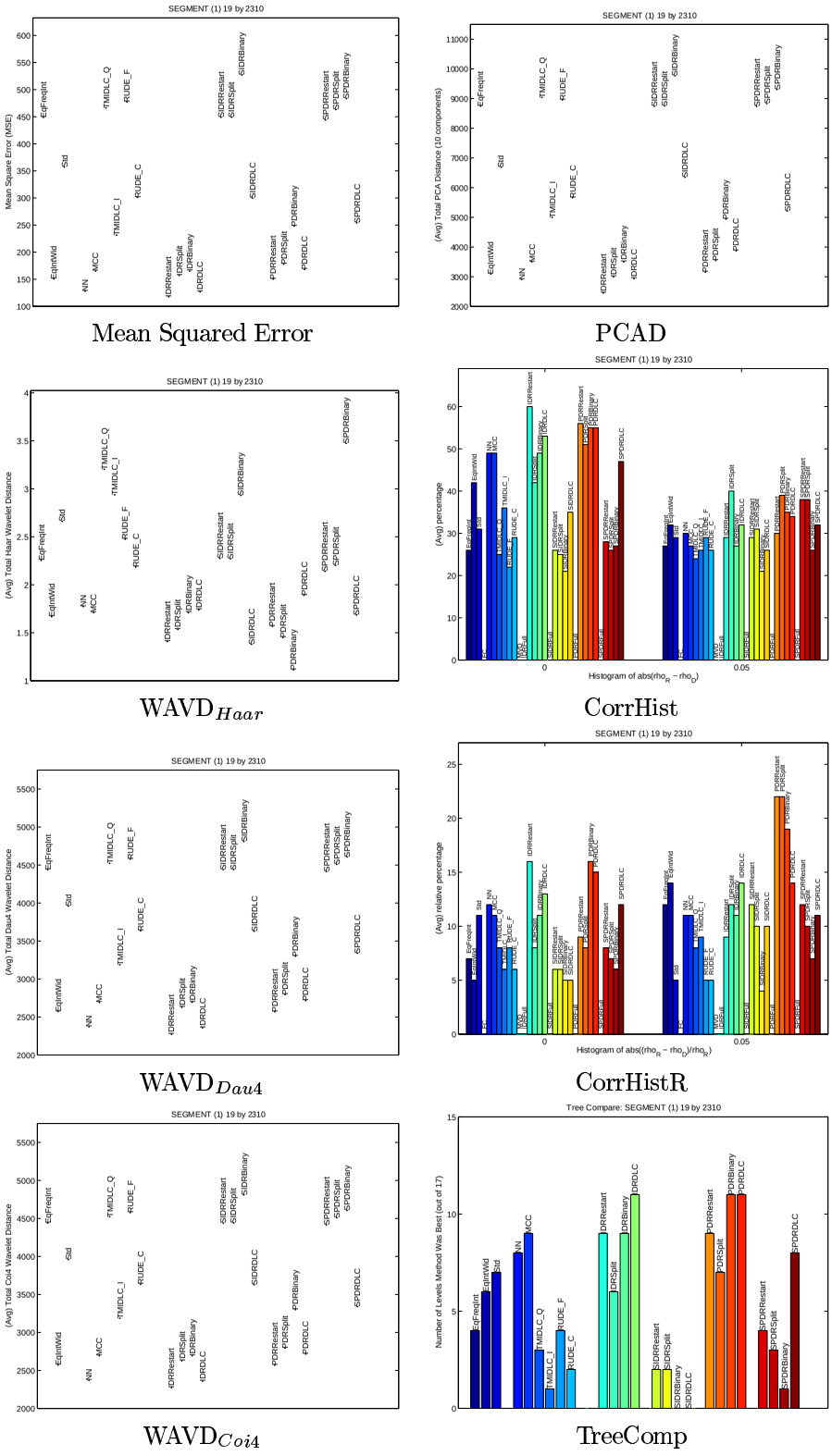


Figure B.13: Metrics on the SEGMENT Dataset (N=3)

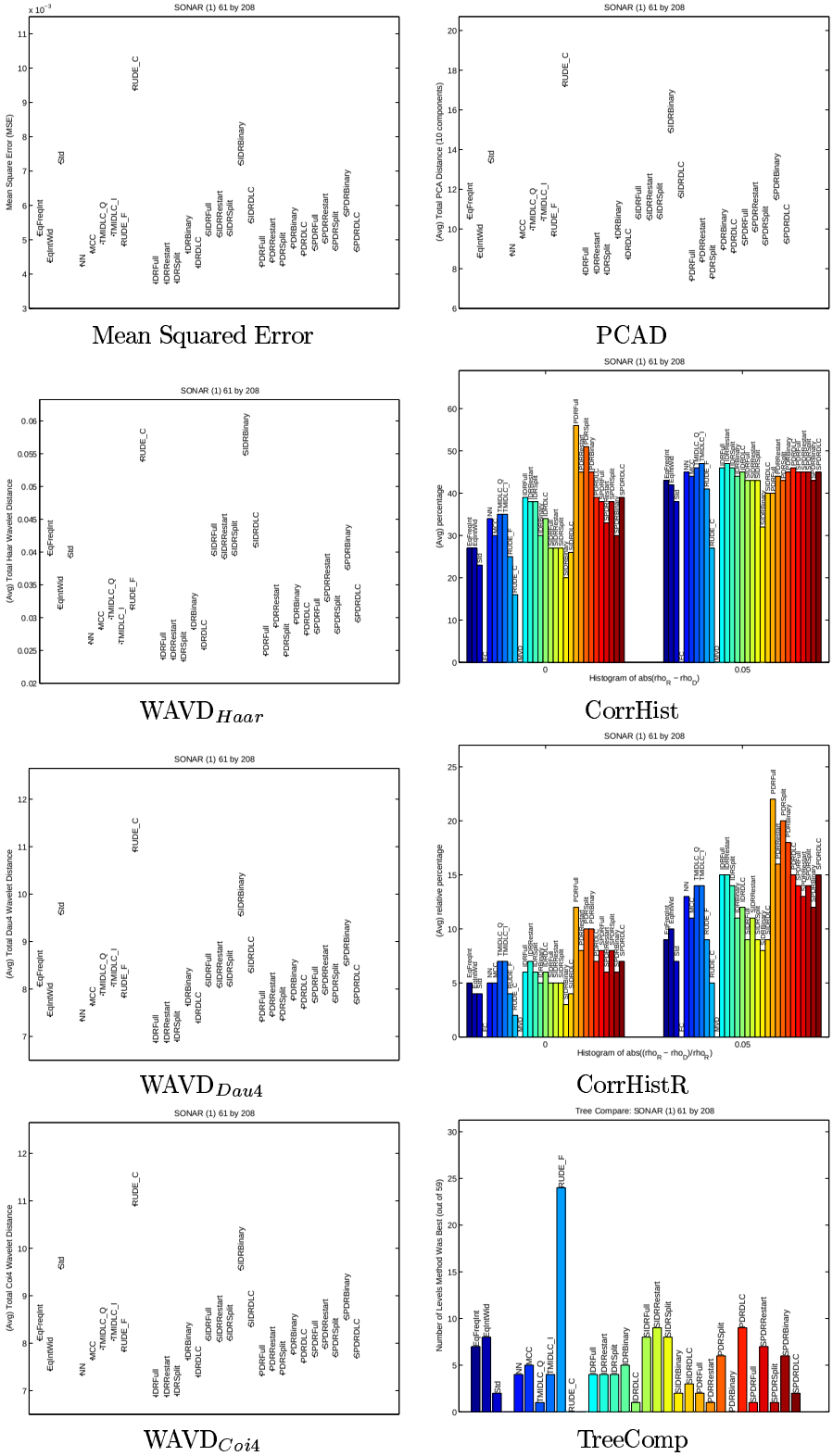


Figure B.14: Metrics on the SONAR Dataset (N=3)

Appendix C

Complete Discretization Results (N=5)

This chapter presents the complete discretization results for $N = 5$ bins using eight evaluation metrics for 31 discretization techniques on 14 datasets described in Appendix A. All datasets are shown for each metric in turn, namely Mean Squared Error, WAVD, PCAD, CorrHist, CorrHistR, and TreeComp. Mean Squared Error (MSE) measures the average squared Euclidean Distance between the discrete version of a vector and the real-valued, continuous version. The wavelet distance functions WAVD_{Haar}, WAVD_{Dau4}, and WAVD_{Coi4} measure the Euclidean Distance between the wavelet coefficients of the discrete datamatrix and the real datamatrix using Haar, Daubechies 4 and Coiflet 4 wavelets respectively. PCAD measures the Euclidean Distance between the top 10 principal components of the discrete datamatrix and the real datamatrix. CorrHist provides a histogram of the Euclidean Distance between the vector of pairwise correlations calculated on discrete vectors and the vector of corresponding pairwise correlations calculated using real vectors. CorrHistR is the same as CorrHist except that the difference is scaled by the magnitude of the correlation to provide relative distances. Finally, TreeComp measures the number of levels in the dendrogram computed on discrete data in which a particular method achieves the best performance with respect to preserving the same pairs found in the dendrogram computed on the real data.

For the MSE, WAVD, and PCAD metrics, better performance is indicated by values lower on the vertical scale. For CorrHist, CorrHistR, and TreeComp, better performance is indicated by higher bars. For CorrHist and CorrHistR, we are particularly interested in the height of the bars over value 0 on the horizontal axis. The 0.05 is used to break ties. For ease of comparison, the ordering of the 31 methods, from left to right, remains the same from plot to plot. Also, for CorrHist, CorrHistR, and TreeComp, the coloration of the bars remains the same across plots. For example, EqFreqInt is always shown on the extreme left, with dark blue bars where appropriate, while SPDRDLC is always shown on the extreme right with burgundy bars where appropriate. The labelling order and color of the TreeComp plot in the lower right of each page can generally be used as a guide.

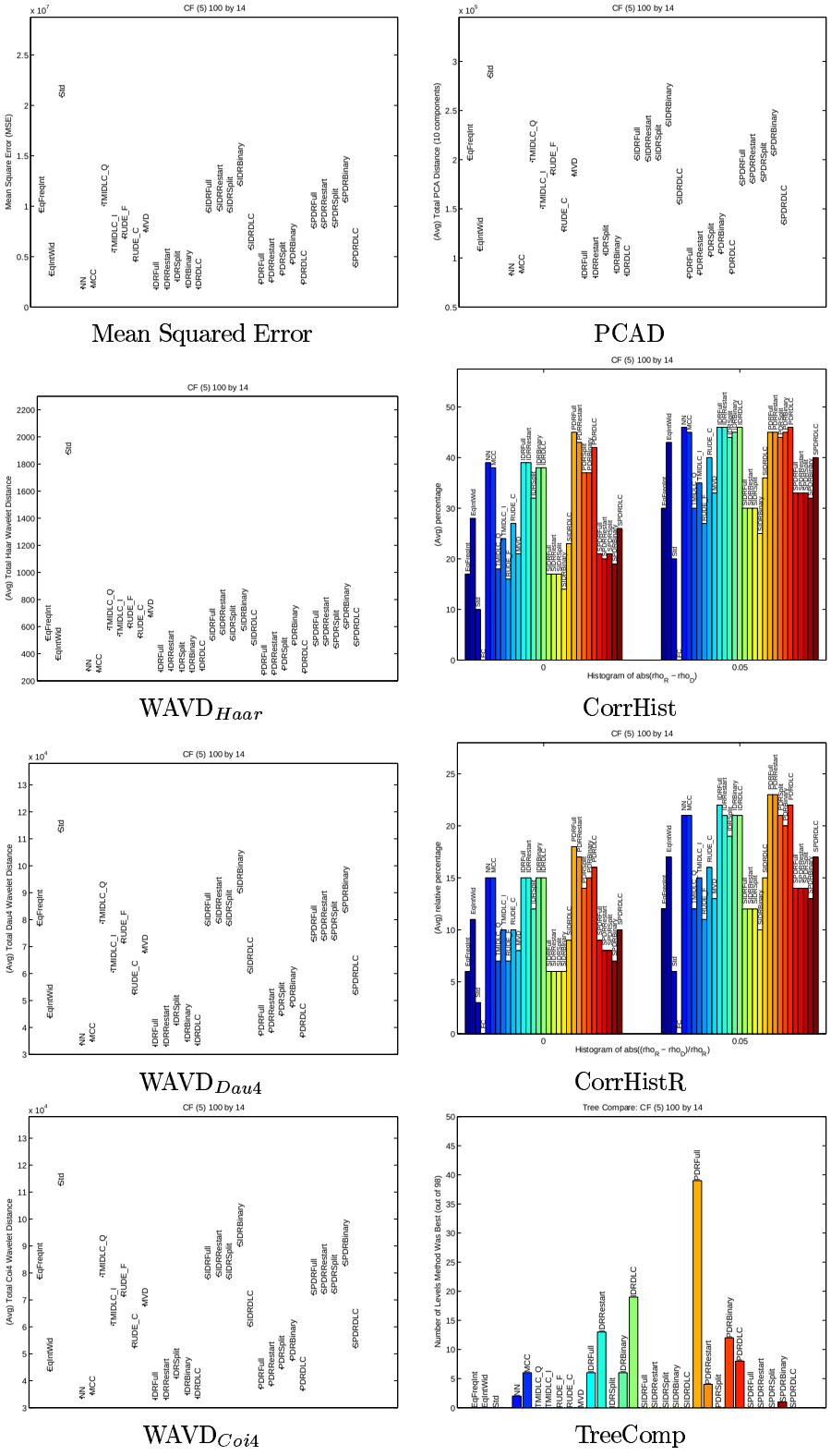
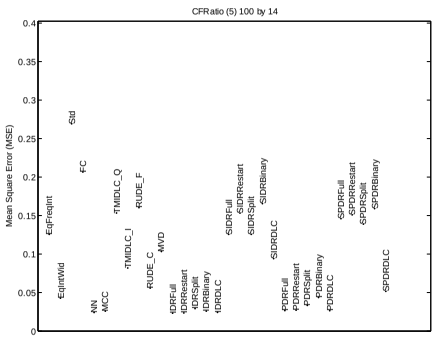
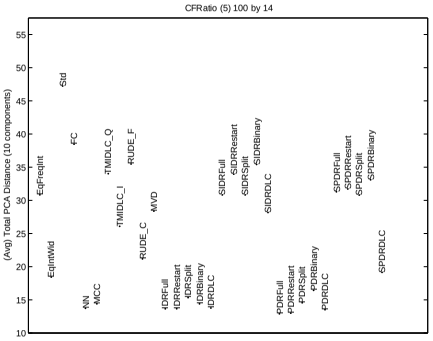


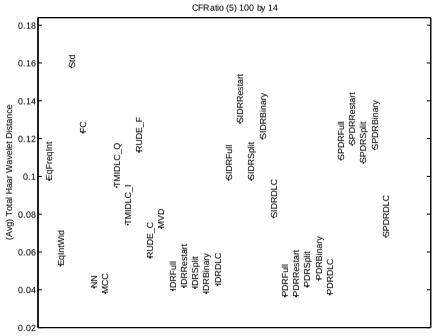
Figure C.1: Metrics on the CF Dataset (N=5)



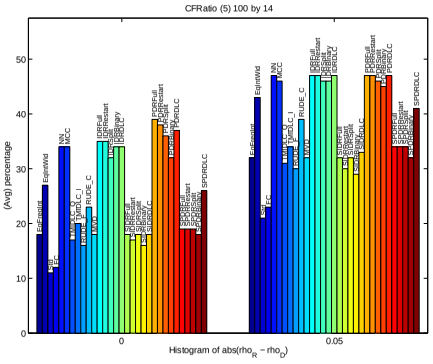
Mean Squared Error



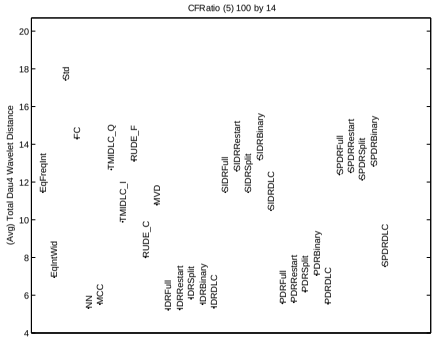
PCAD



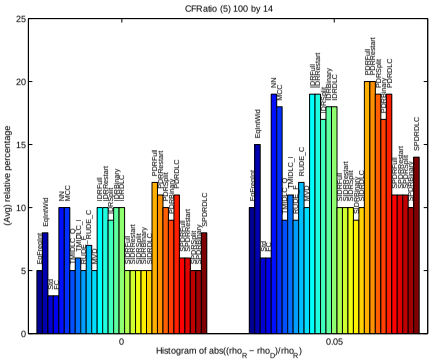
WAVD_{Haar}



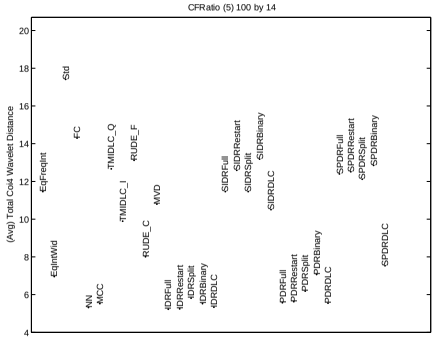
CorrHist



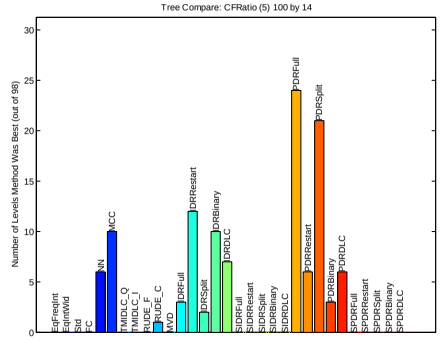
WAVD_{Dau4}



CorrHistR



WAVD_{Coi4}



TreeComp

Figure C.2: Metrics on the CFRatio Dataset (N=5)

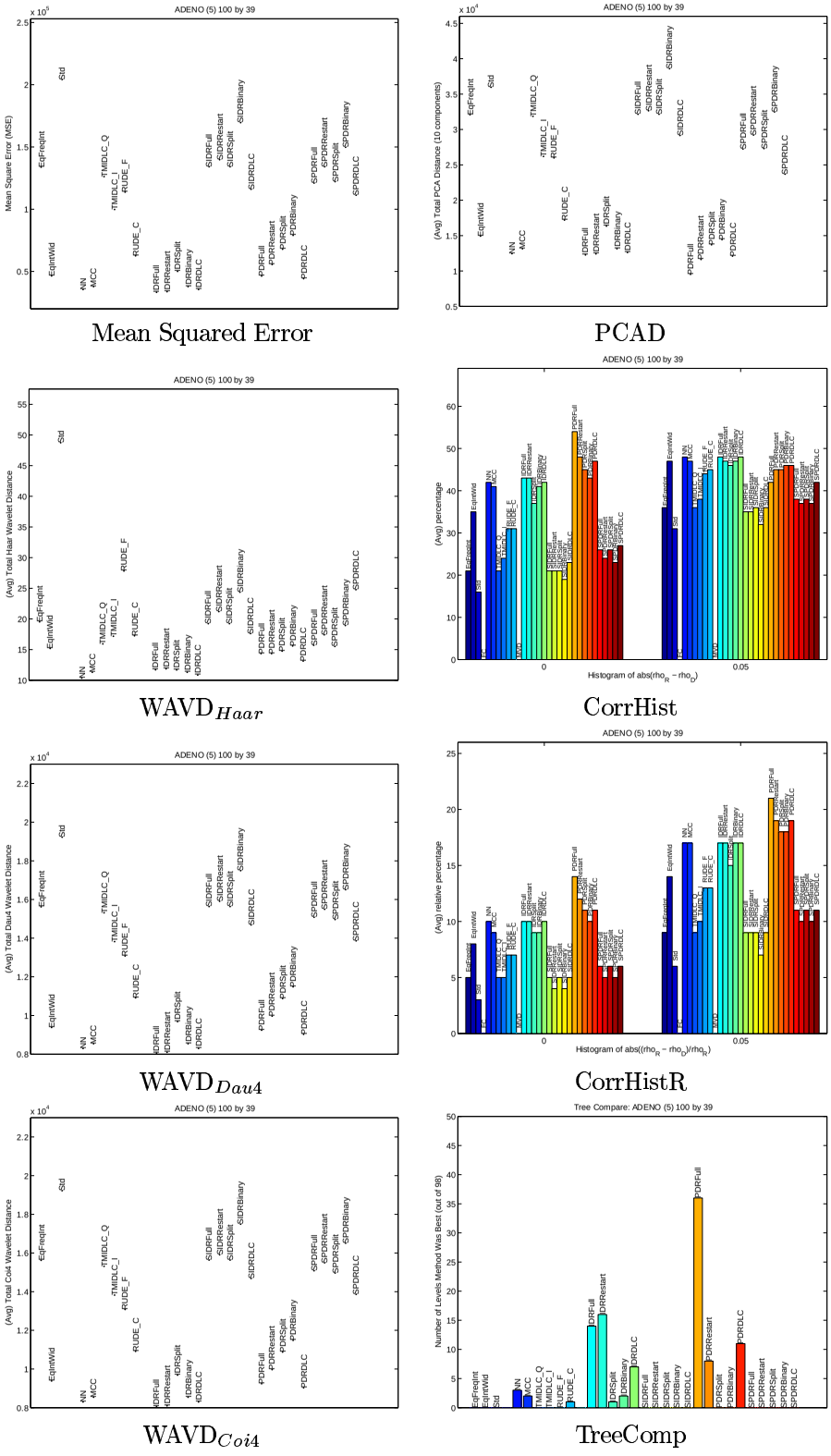


Figure C.3: Metrics on the ADENO Dataset (N=5)

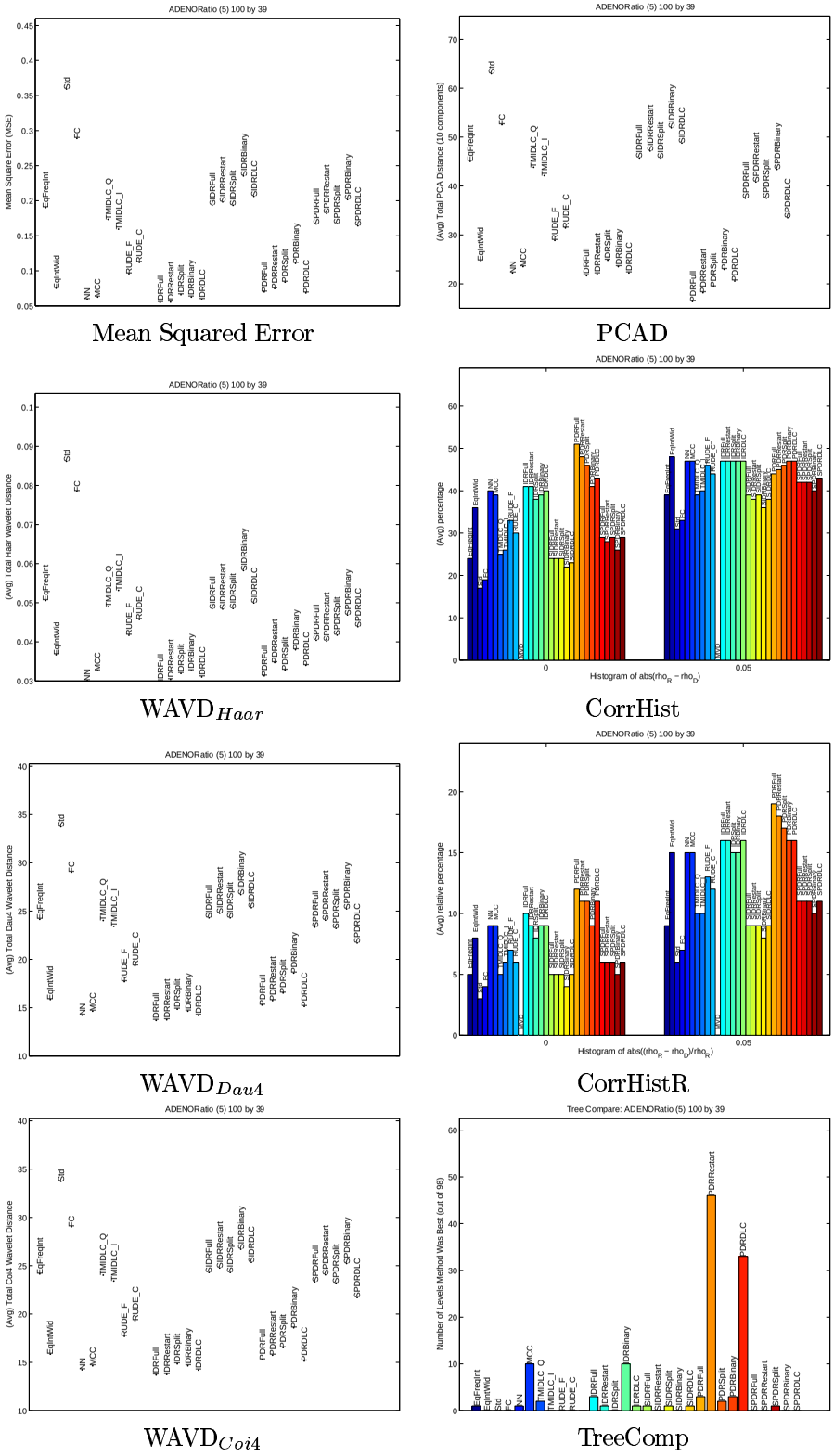


Figure C.4: Metrics on the ADENORatio Dataset (N=5)

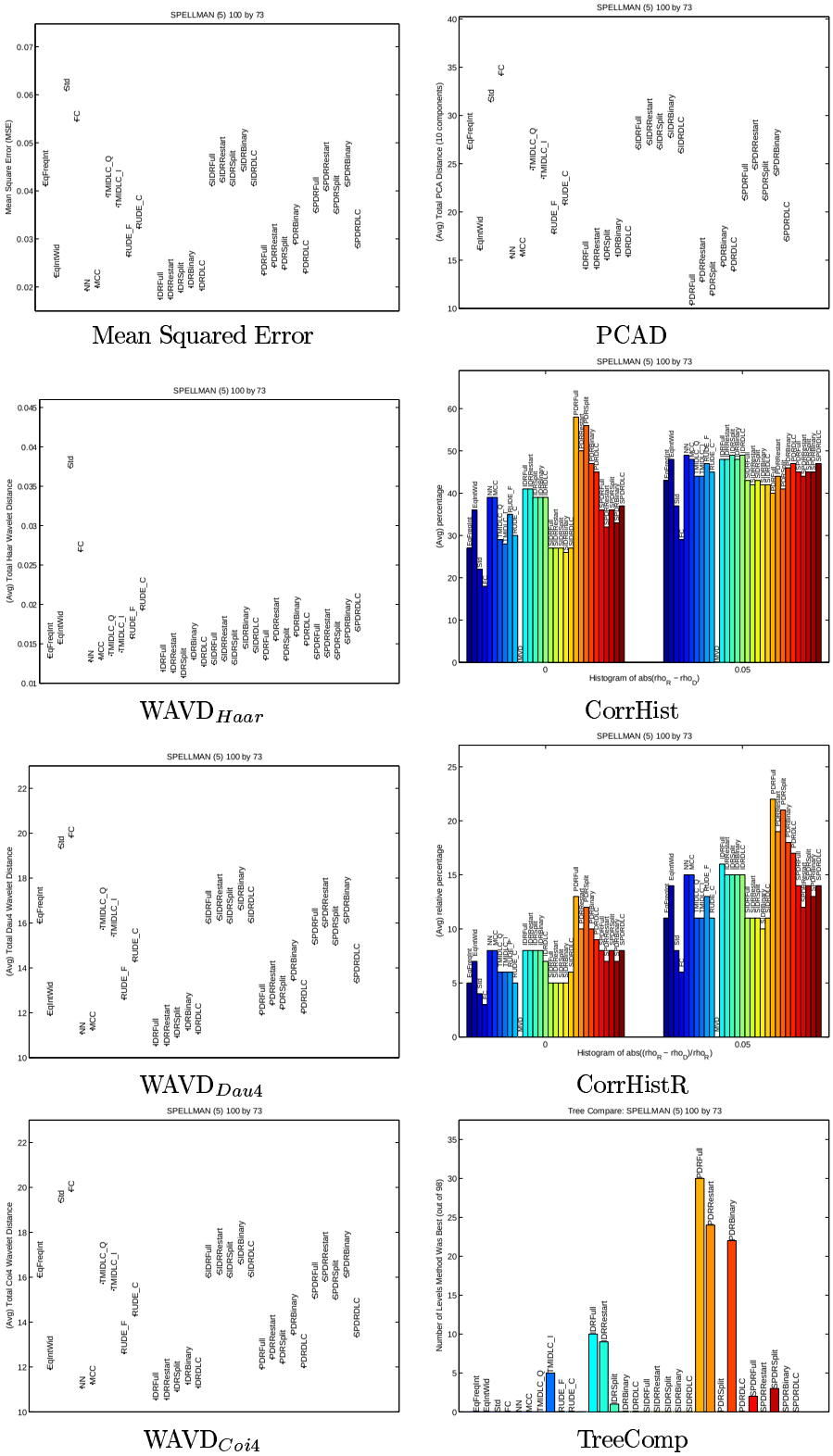


Figure C.5: Metrics on the SPELLMAN Dataset (N=5)

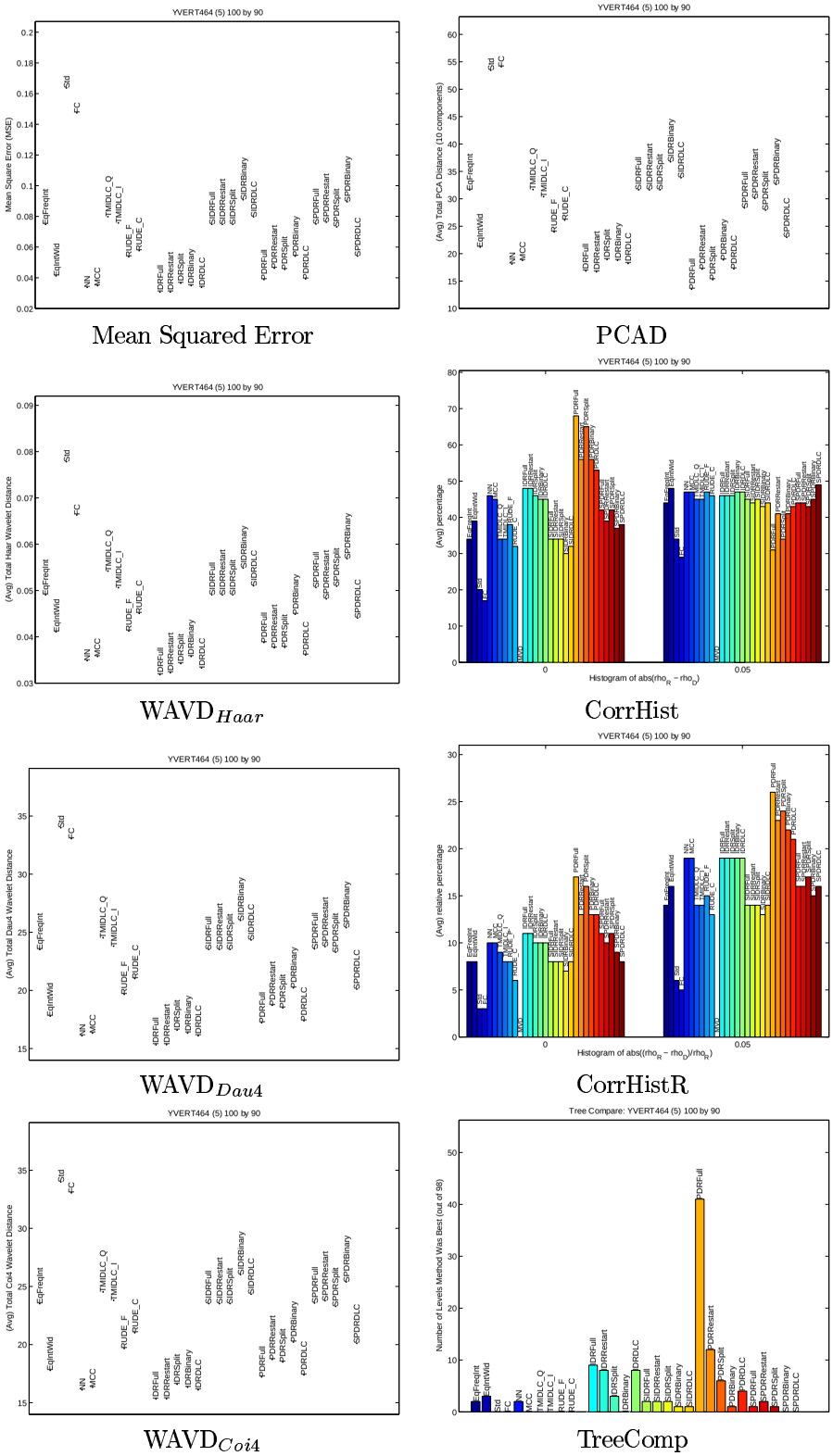


Figure C.6: Metrics on the YVERT464 Dataset (N=5)

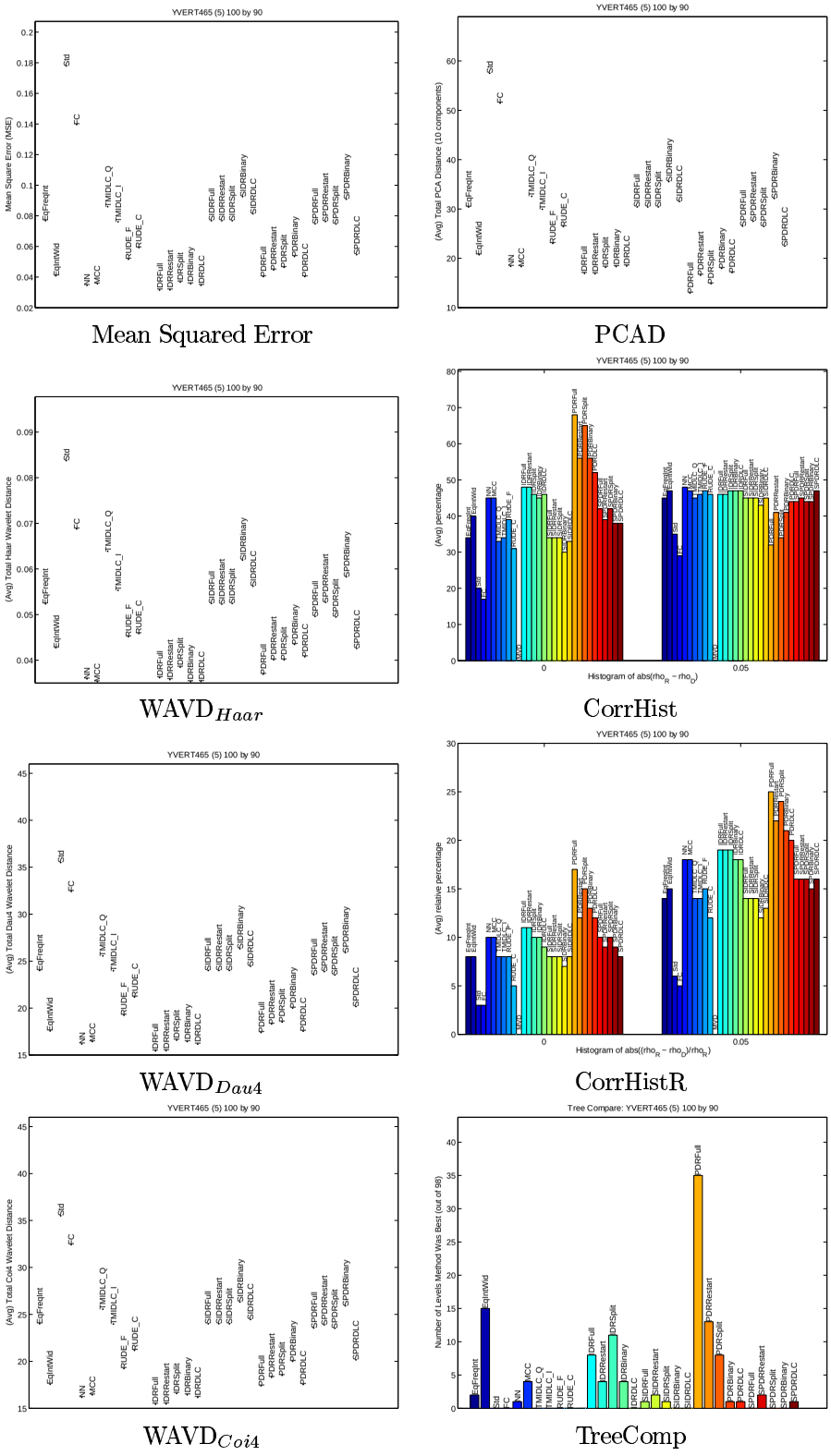


Figure C.7: Metrics on the YVERT465 Dataset (N=5)

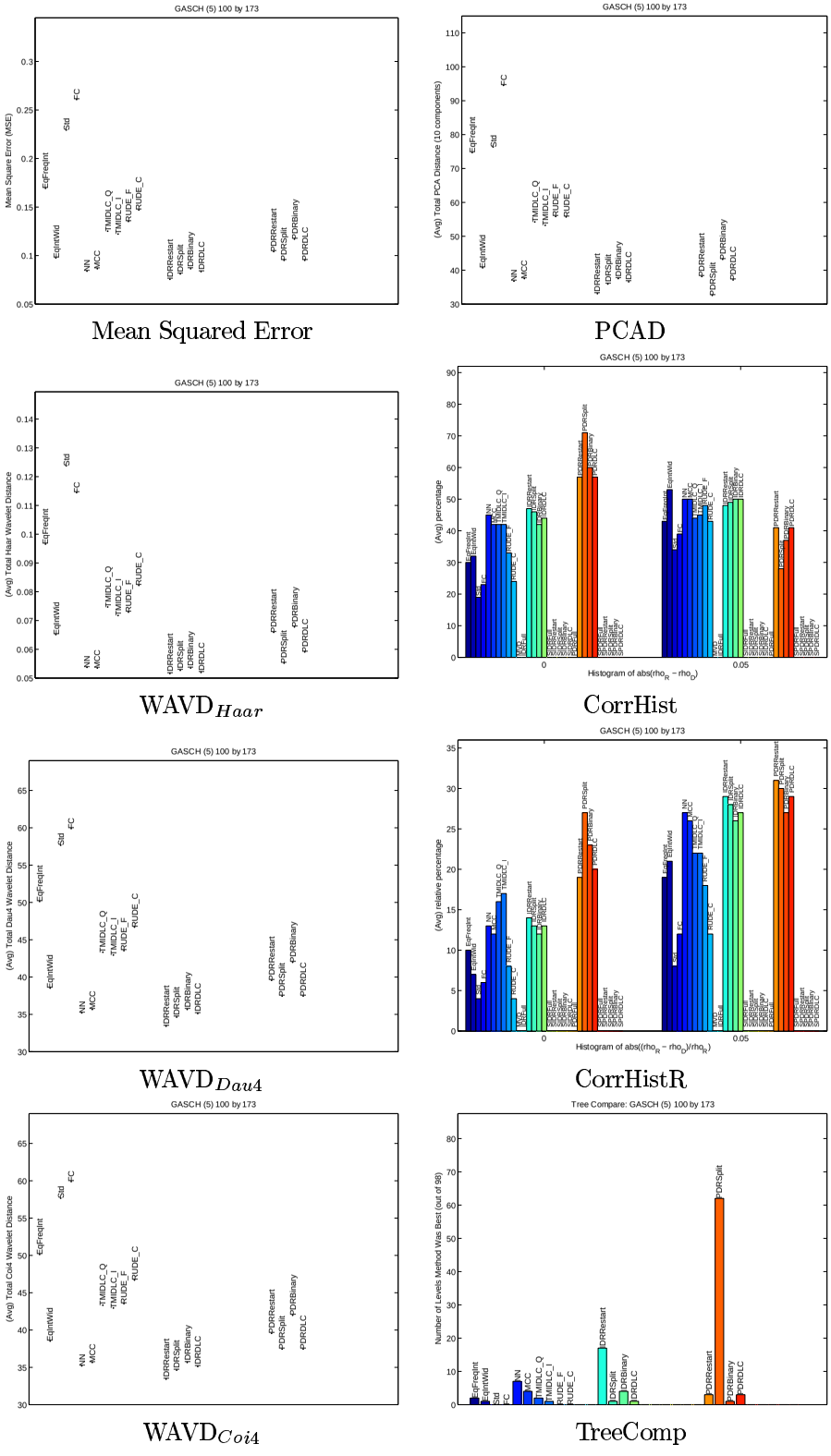


Figure C.8: Metrics on the GASCH Dataset (N=5)

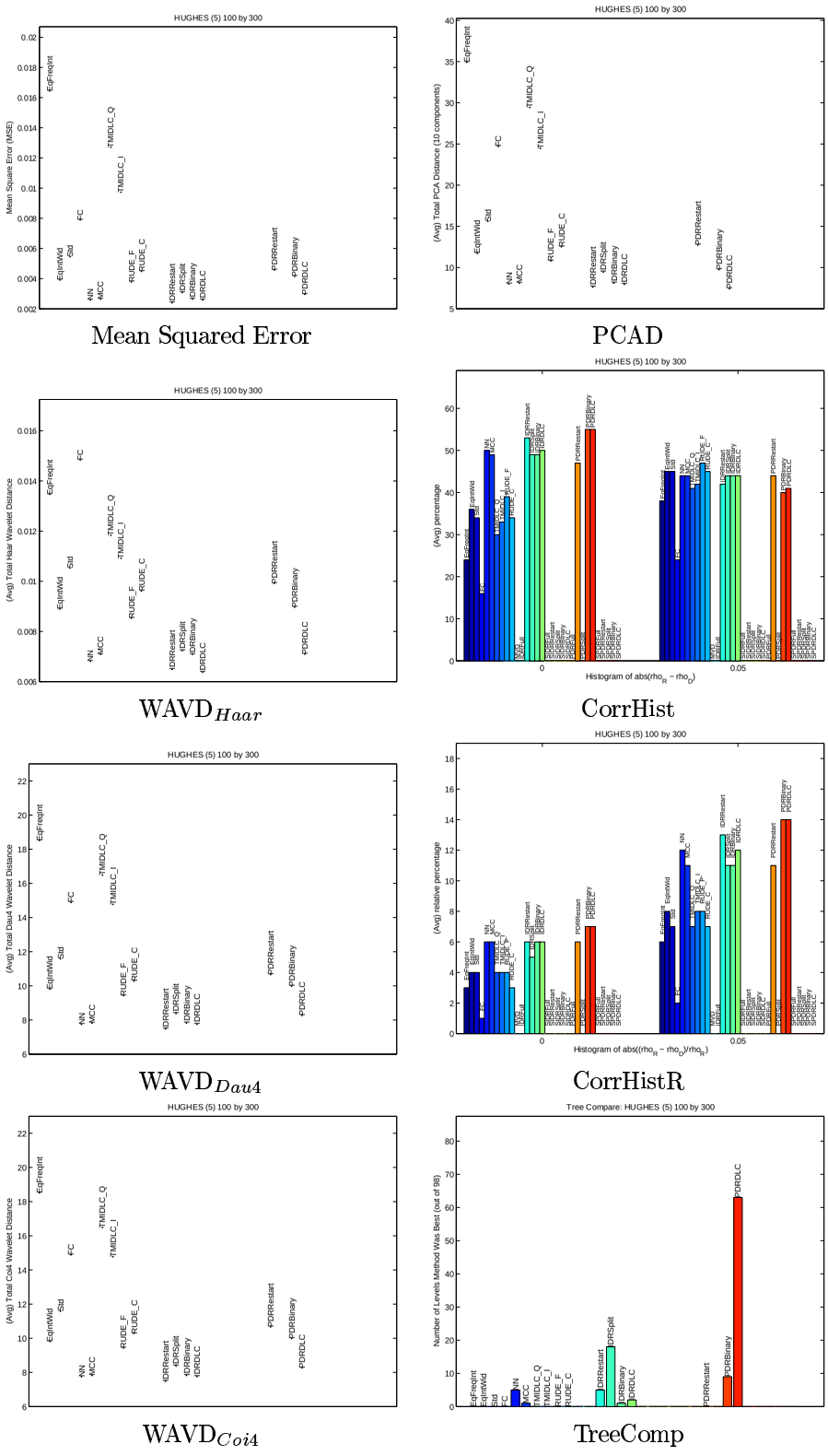


Figure C.9: Metrics on the HUGHES Dataset (N=5)

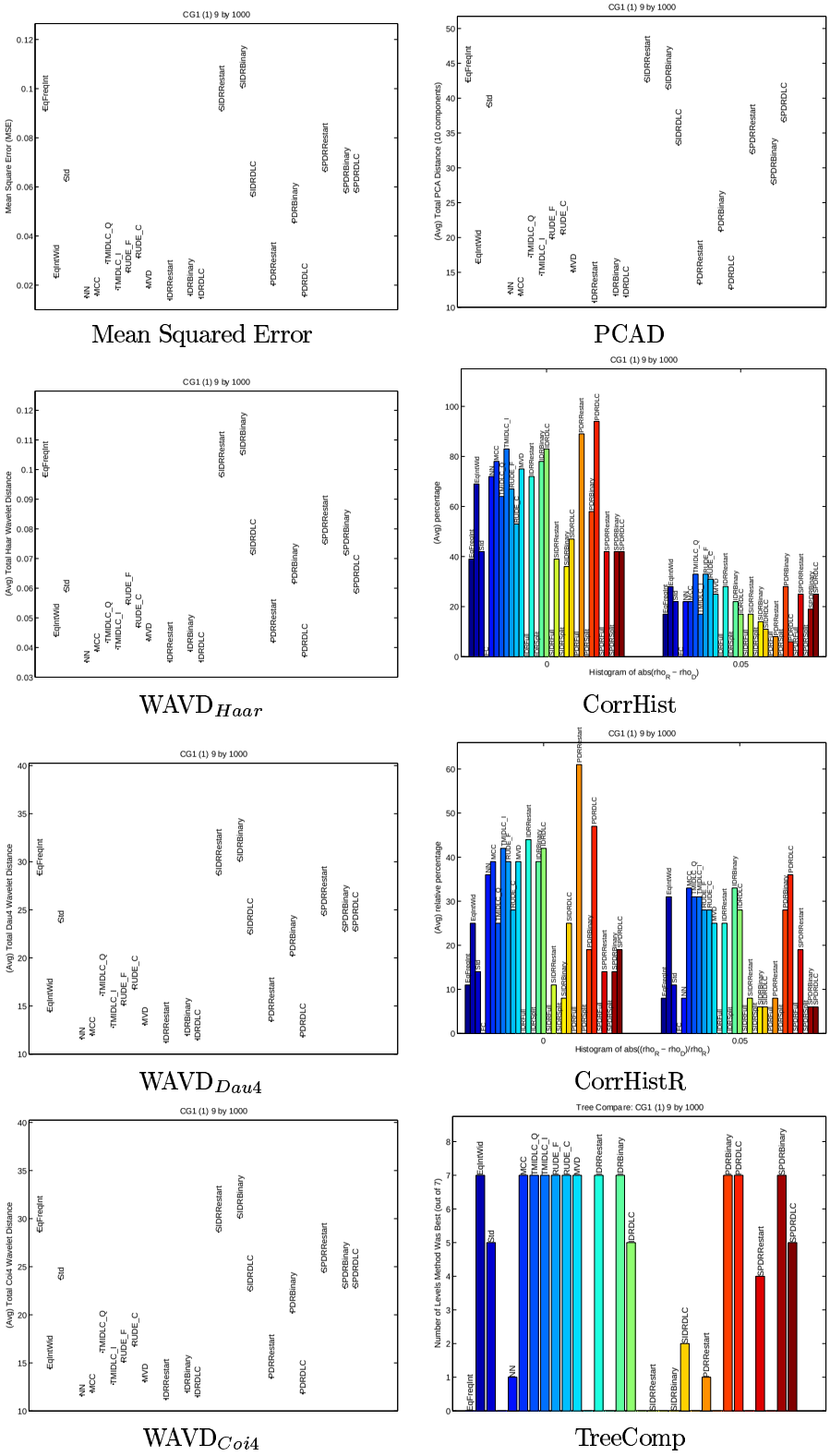


Figure C.10: Metrics on the CG1 Dataset (N=5)

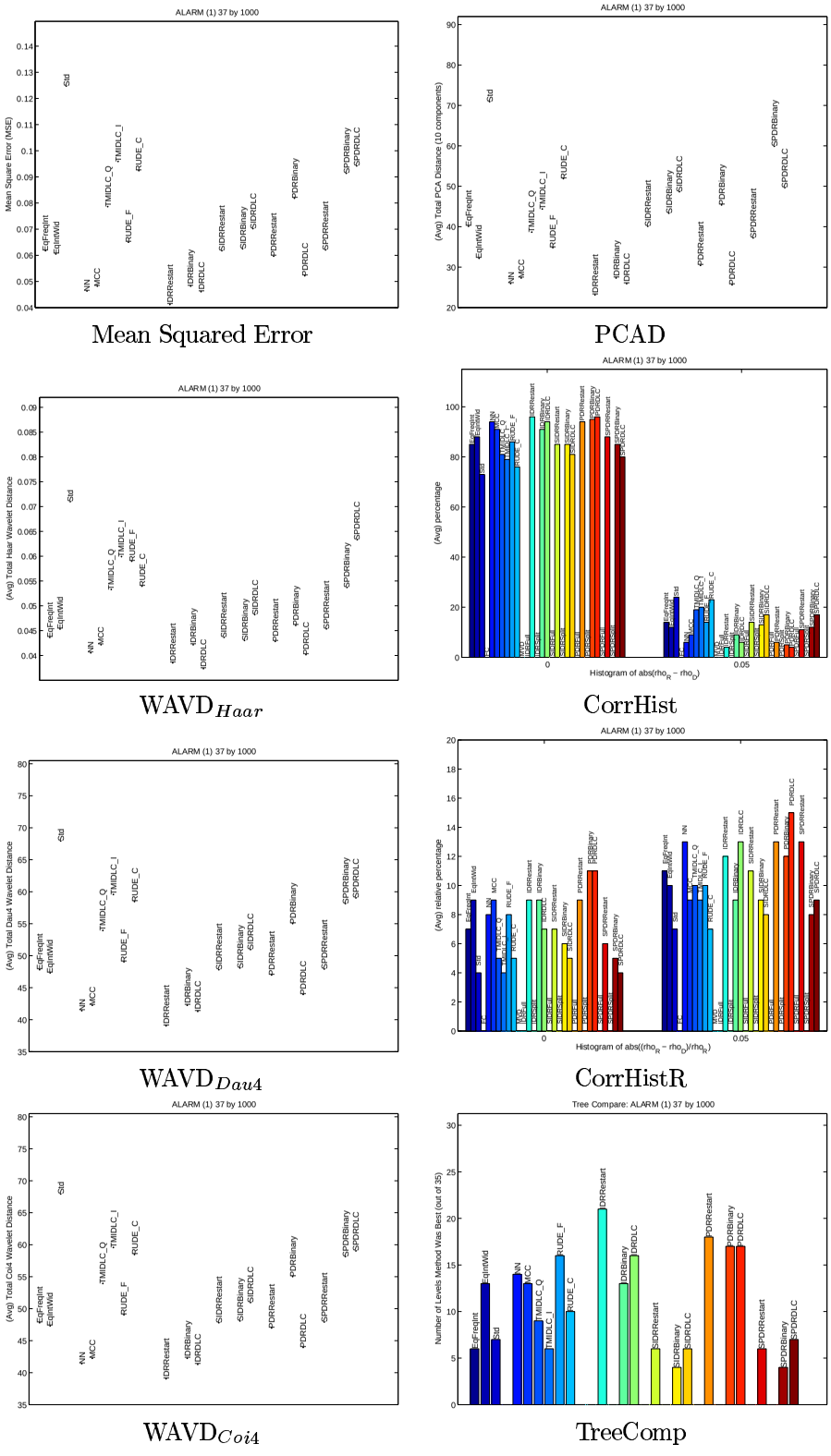


Figure C.11: Metrics on the ALARM Dataset (N=5)

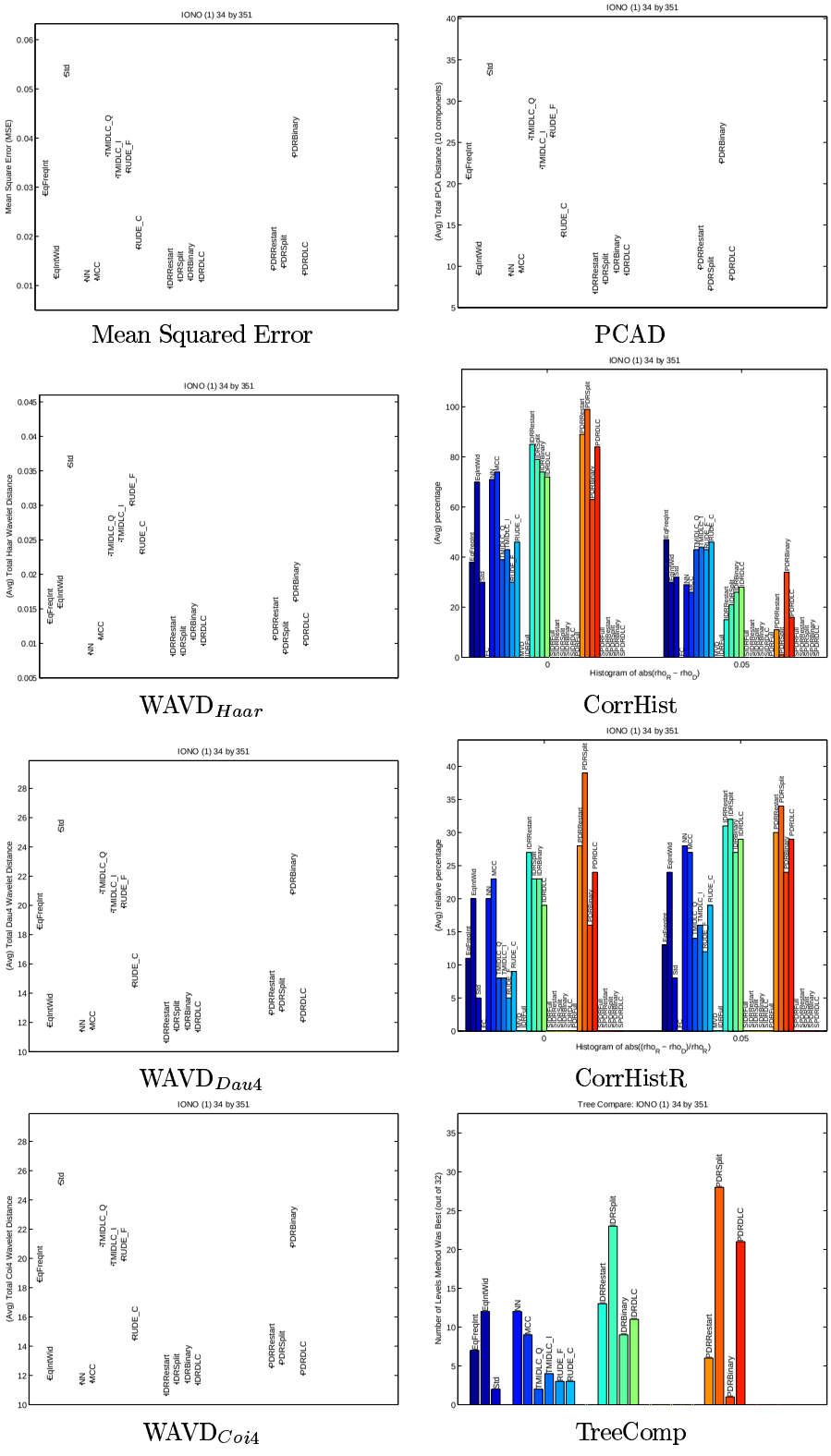


Figure C.12: Metrics on the IONO Dataset (N=5)

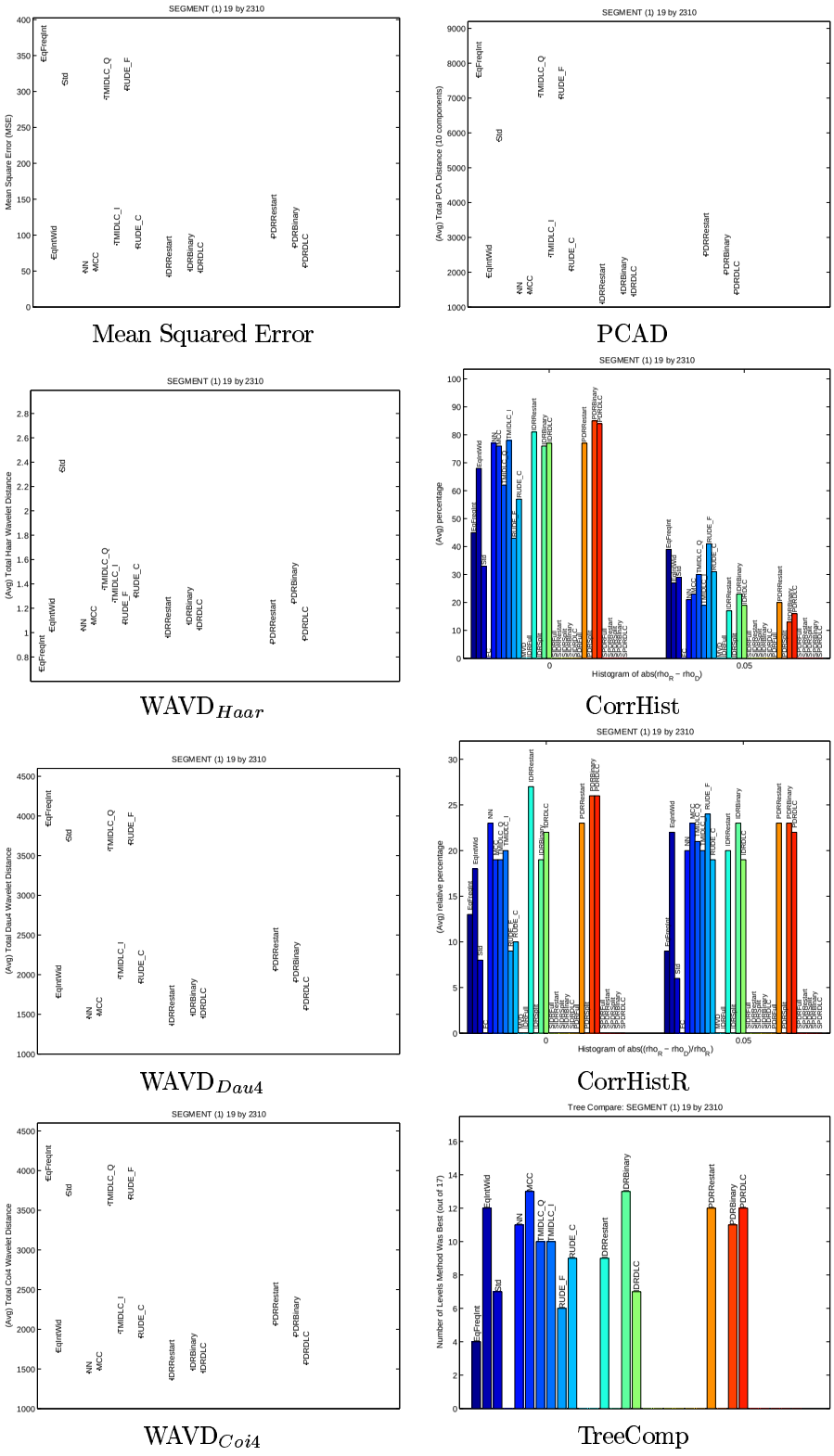


Figure C.13: Metrics on the SEGMENT Dataset (N=5)

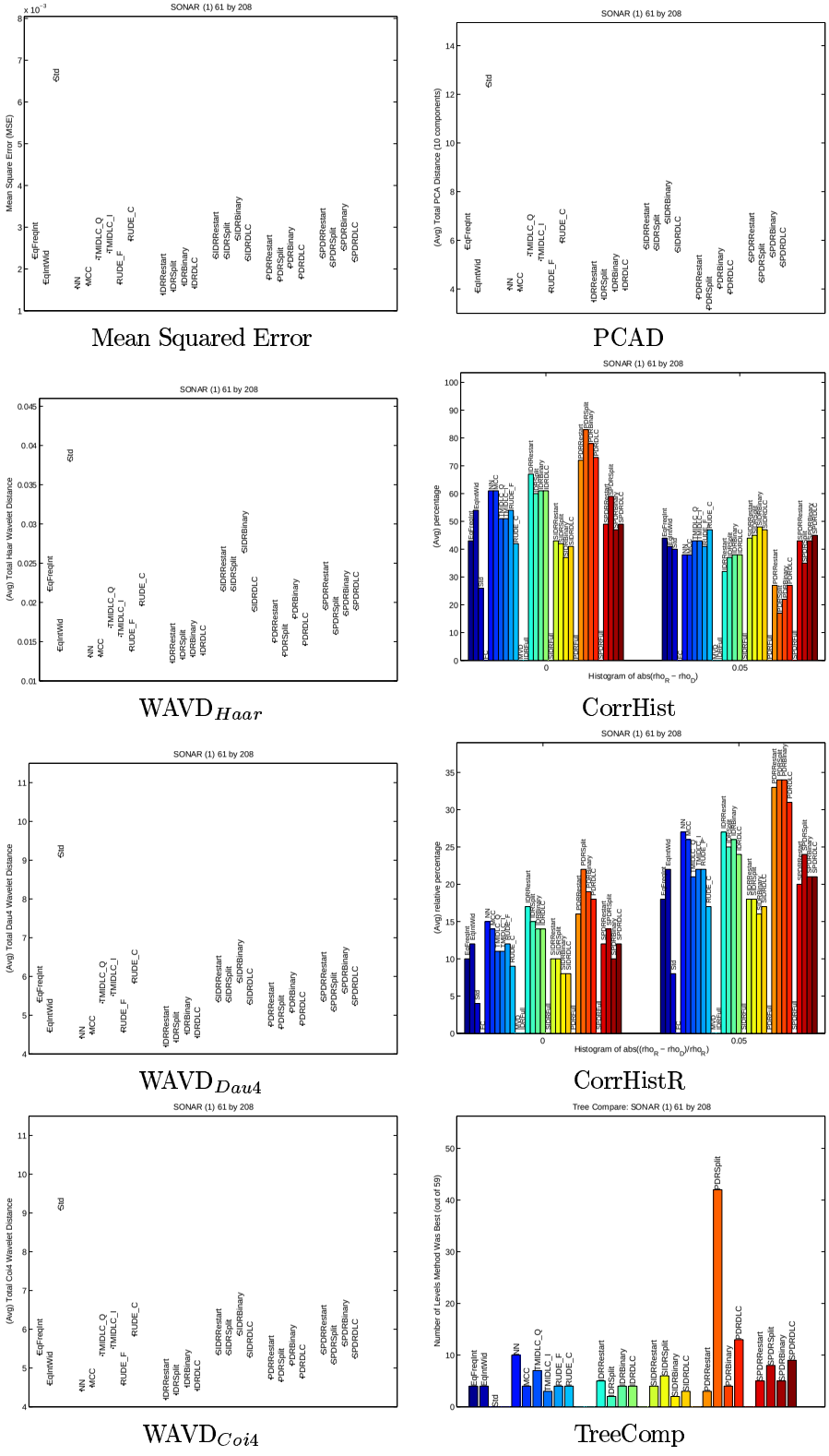


Figure C.14: Metrics on the SONAR Dataset (N=5)

Appendix D

Semantic Similarity using the Gene Ontology

The Gene Ontology™ (GO) consists of a controlled vocabulary of terms for the annotation of molecular attributes in model organisms, where the relationships between terms are captured in a directed acyclic graph structure through arcs denoting *is_a* or *part_of* relationships. An excerpt of the ontology is given in Figure D.1 where the root term branches into three distinct ontologies describing cellular component, molecular function, and biological process, respectively.

Terms are associated to the genes of a model organism manually by expert curators within each community. The reliability of the source of each annotation is indicated by an *evidence code* from the list below.

IC	Inferred by Curator
IDA	Inferred from Direct Assay
IEA	Inferred from Electronic Annotation
IEP	Inferred from Expression Pattern
IGI	Inferred from Genetic Interaction
IMP	Inferred from Mutant Phenotype
IPI	Inferred from Physical Interaction
ISS	Inferred from Sequence or Structural Similarity
NAS	Non-traceable Author Statement
ND	No biological Data available
RCA	inferred from Reviewed Computational Analysis
TAS	Traceable Author Statement
NR	Not Recorded

As a curated resource with a controlled vocabulary, GO has often been used as a gold standard for many applications. Recently, GO has been proposed as a tool for measuring similarity of genes. The *semantic similarity* between genes is calculated based on statistical and topographical information about GO terms and their relationships according to the ontology.

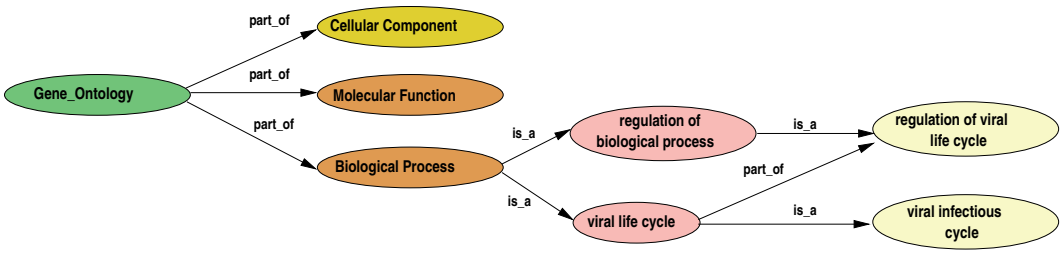


Figure D.1: Excerpt of the Gene Ontology (taken from [WABD04])

.

The work of Lord *et al.* [LSBG03] investigates the correlation of semantic similarity with protein sequence similarity. They used three different definitions of similarity considered in an earlier paper by Budanitsky and Hirst [BH01] who study semantic distance in WordNet [Fel98], an online lexical reference system with a structure similar to GO in that English nouns, verbs, adjectives and adverbs are organized into synonym sets and links exist between those sets. Using the same metrics as Lord and his colleagues, Wang *et al.* [WABD04] quantify the relationship between semantic similarity and values of Pearson correlation for gene expression profiles.

In this chapter, we investigate the correlation between GO semantic similarity and our sources of evidence of gene interaction for the genes of the yeast genome. We consider two key questions: 1) Given that a technique has measured interaction, what is the level of GO similarity; and 2) Given that a pair has high GO similarity, what is the strength of evidence for interaction. By interaction, we mean assertions of a relationship between the genes, such as by the set of explicit and implicit experts described in Chapter 2.

D.1 Semantic Similarity

To compute similarity between two genes using GO, we cannot simply calculate the minimum path-length between any pair of terms assigned to the genes. As argued by Lord *et al.* [LSBG03], the structure of the ontology is inconsistent across terms so that the depth of a node is a function of the peculiarities of biological knowledge rather than a comment about the specificity of a particular term. Instead, we employ the same three information theoretic measures of semantic similarity used in previous studies which capture the intuition that less frequently used terms are more informative. In the following, we use the notation of Lord *et al.* [LSBG03].

All three semantic similarity measures calculate the probability of a term based on corpus statistics which indicate the proportion of the number of items in the corpus assigned that term relative to the total number of items in the corpus. Probabilities are calculated separately for each of the three main branches of GO, namely Cellular Component, Molecular Function, and Biological Process. Since child terms inherit from their parents, a gene annotated with a particular term also is annotated with all parents of that term. For two genes with multiple terms under one GO branch, we consider all pairs of those terms and assign similarity as the maximum achieved by any pair.

Given the probability p_c of a term c calculated as the proportion of items assigned to c in the corpus, the value $-\log p_c$ is known as the information content of the term c. We can define $S(c_1, c_2)$ as the set of parental concepts shared by both c_1 and c_2. The graph structure of GO allows two terms to have multiple parents which means two terms can share the same parents by multiple paths. We take the parent with the minimum p_c which we call p_{ms} for the *probability of the minimum subsumer*:

$$p_{ms}(c_1, c_2) = \min_{c \in S(c_1, c_2)} \{p(c)\}$$

Using these quantities, the similarity metrics are defined as:

- **RESNIK** This metric by Resnik [Res95] uses only the information content of the shared parents and varies from 0 for no similarity to $-\log_2(1/t) = \log_2(t)$ for maximum similarity, where t is the total number of items in the corpus:

$$sim(c_1, c_2) = -\log_2 p_{ms}.$$

- **LIN** This metric by Lin [Lin98] uses both the probability of the parent term as well as the two query terms, and varies from 0 for dissimilar terms to 1 for maximum similarity:

$$sim(c_1, c_2) = \frac{2 \cdot [\log_2 p_{ms}(c_1, c_2)]}{\log_2 p(c_1) + \log_2 p(c_2)}.$$

- **JIANG** This metric by Jiang and Conrath [JC97] is actually a distance metric which we convert to a similarity metric using the knowledge that it achieves a maximum of $2\log_2(t)$:[1]

$$dist(c_1, c_2) = 2\log_2 p_{ms}(c_1, c_2) - [\log_2 p(c_1) + \log_2 p(c_2)]$$

$$sim(c_1, c_2) = 2\log_2(t) - dist(c_1, c_2).$$

For all metrics, if $\log_2 p_{ms}(c_1, c_2) = 0$ then we set $sim(c_1, c_2) = 0$.

D.2 Question 1: GO Similarity Distribution for Interactions

Our first question examines the level of GO similarity for each interaction suggested by expert information sources using a histogram of GO similarity scores. For each expert, we define three outcomes for an interaction, analogous to the creation of negative data discussed in Section 5.3. For the explicit experts, we have Positive assertions of interaction where the assay detected an interaction, otherwise the assertion is Unknown. For the implicit experts, we have a Positive assertion of interaction when the category of each gene is known under an expert and it is the same for both genes. When the category is known for both but different, we have a Negative assertion. Otherwise, the implicit expert asserts Unknown.

[1] The formula for JIANG in Lord *et al.* [LSBG03] has a typographical error, incorrectly including a minus sign in front of 2 $\log p_{ms}$.

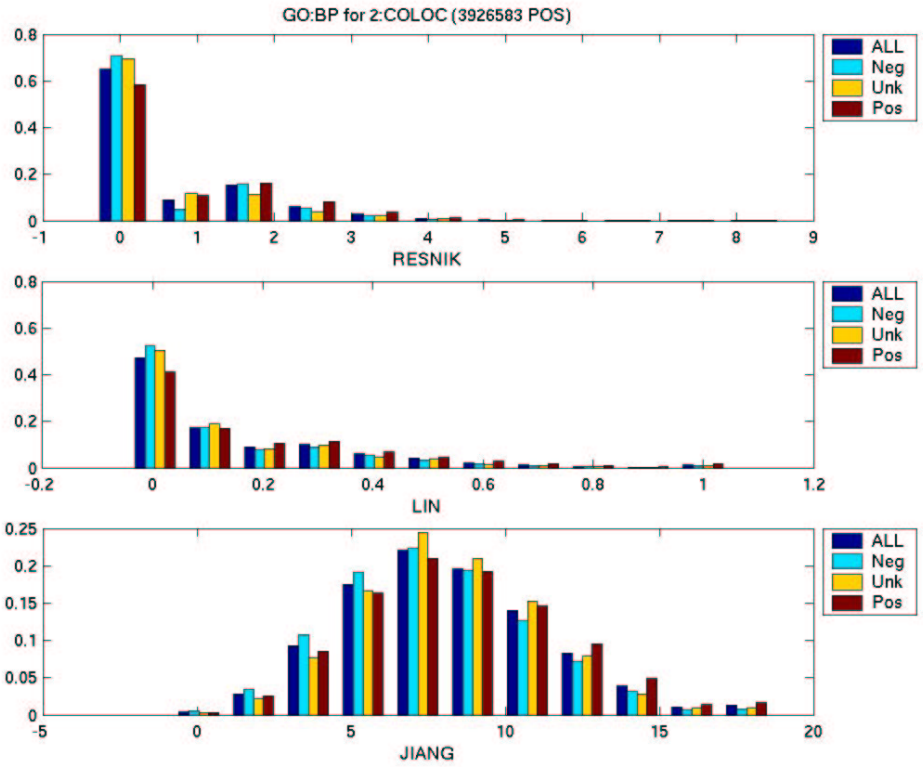
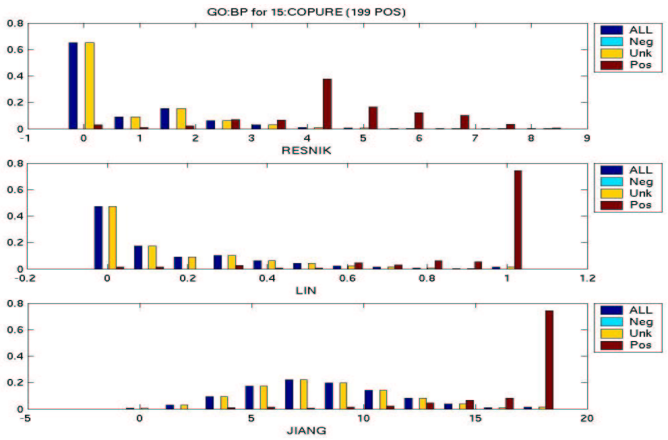


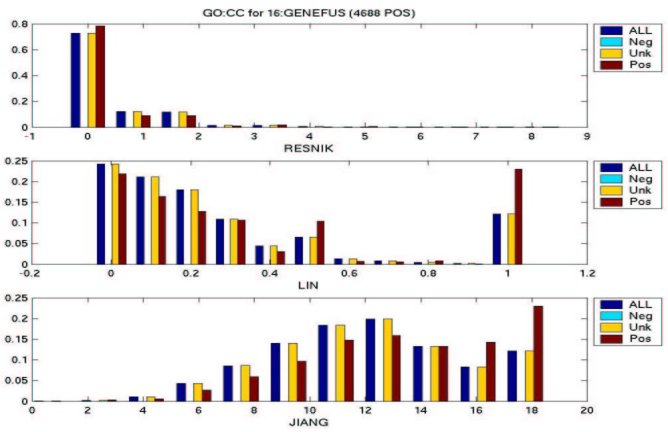
Figure D.2: Examples of GO Semantic Similarity Distributions

For each expert, we empirically find the distribution of semantic similarity scores, per branch of GO, using each metric for the Unknown, Positive and (when appropriate) Negative assertions. We can compare these to the distribution of semantic similarities for ALL pairs of genes under the same metric and same ontology branch. An example display is shown in Figure D.2 for the implicit expert COLOC which measures interaction by inferring co-location. The result is shown for all three metrics calculated using terms from the Biological Process branch (GO:BP). The histogram at the top of the plot shows the RESNIK distribution of semantic similarity scores, while the middle and bottom histograms show the LIN and JIANG scores respectively. In the title of the plot, the number in parenthesis indicates how many Positive assertions were made by the expert, 3926583 in the case of COLOC. As can be seen from the plot, the similarity scores for the Positive assertions from this expert are mainly centered around zero for the RESNIK and LIN metrics. On the other hand, the distribution for Positive assertions under the JIANG metric stretches further in the higher range of similarity values. This result is typical of all the combinations of experts and GO branch we considered, where JIANG proved to be more discriminative than RESNIK and LIN.

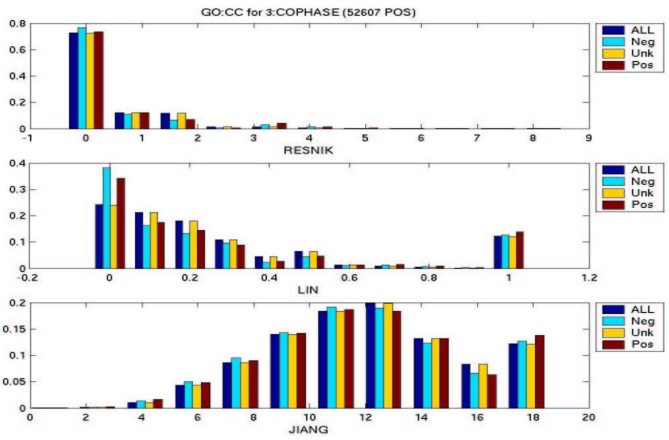
To determine the discrimination power of the JIANG measure, we examine the degree of separation of the Positive distribution from the ALL distribution for each combination of interaction



(a) Well-Separated



(b) Semi-Separated



(c) Poorly-Separated

Figure D.3: Examples Demonstrating Degrees of Separation for GO Semantic Similarity Distributions

expert and GO branch. By visual inspection, we categorize the correlation of JIANG scores under each GO branch for each type of interaction expert according to whether the Positive distribution is *well-separated*, *semi-separated* or *poorly-separated* from the ALL distribution. Figure D.3(a) shows an example of a well-separated distribution since scores for the majority of Positive assertions populate the higher range of JIANG. Figure D.3(b) shows an example of a semi-separated distribution, where the distribution of Positive assertions is generally shifted to the right of the ALL distribution. Figure D.3(c) shows an example of a poorly-separated distribution, where the Positive distribution is indistinguishable from the ALL distribution.

Using the above criterion of quality of separation, we can classify each of the explicit and implicit experts used for yeast. Figure D.4 shows a summary of those results for each branch of GO. The red indicates well-separated distributions using JIANG, the green indicates semi-separated distributions, and the darker blue indicates poorly-separated distributions. The additional color of turquoise appearing in the row labelled MIPS FUNC denotes that the distribution lay between poorly and semi-separated by visual inspection. In general, the implicit experts appear towards the top of the plot, followed by the explicit experts. From these results, we can see that the majority of the implicit experts are poorly separable (dark blue/turquoise). Among the explicit experts, distributions for the predictive experts such as gene fusion (GENEFUSION), phylogenetic profiles (PHYLOPROF) and transcription binding site location analysis (LOCANALY) are generally semi-separated (green). The majority of other explicit sources have well-separated distributions. For the GO:CC branch, the poor results are either because the expert ignores cellular location (Y2H, HiThru) or the expert involves transcriptional information (LOCANALY, TRANSFAC) where proteins necessarily must exist in multiple cellular compartments.

D.3 Question 2: Interactions for GO Similarity Ranges

In order to investigate the strength of interaction data for pairs of genes with high GO semantic similarity scores, we use two different methods. The first method examines the distribution of GO similarity scores for pairs with *any* source detecting Positive interaction versus an average of five random samples, of the same size, of similarity scores from all possible gene pairs. Figure D.5 shows the distributions from those pairs with evidence, denoted Evid, and the average from the random pair samples, denoted Avg Random. Again, the plots show that JIANG in more discriminative than LIN and RESNIK in favoring high GO similarity scores for pairs with evidence of interaction.

The second method counts the number of pairs with specific GO similarity values and a specific number of Positive assertions of interaction from our expert sources. The number of experts range from a minimum of 1 to a maximum of 16 in agreement for a particular gene pair. We can compute a matrix C where $C(i,j)$ denotes the number of pairs with i pieces of Positive evidence and similarity score j, where the scores are divided into discrete bins. Figure D.6 through Figure D.8 show the results as a heatmap, with the number of pieces of evidence shown along the horizontal axis and the value of the similarity scores shown along the vertical axis (maximum similarity is shown at the

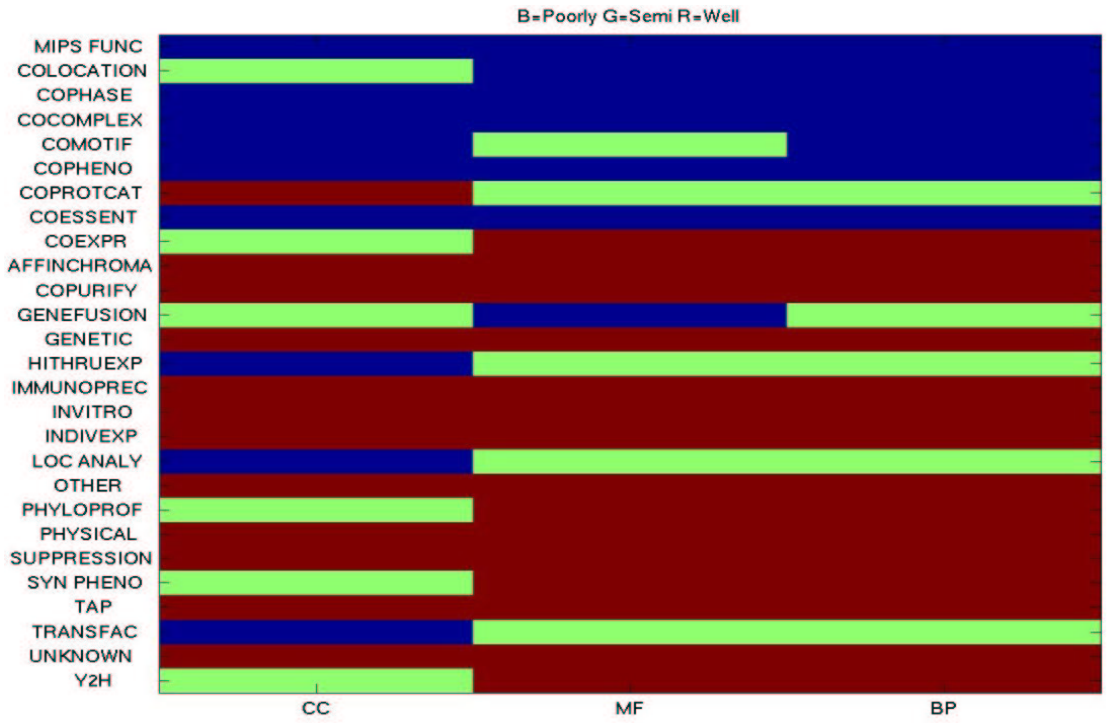
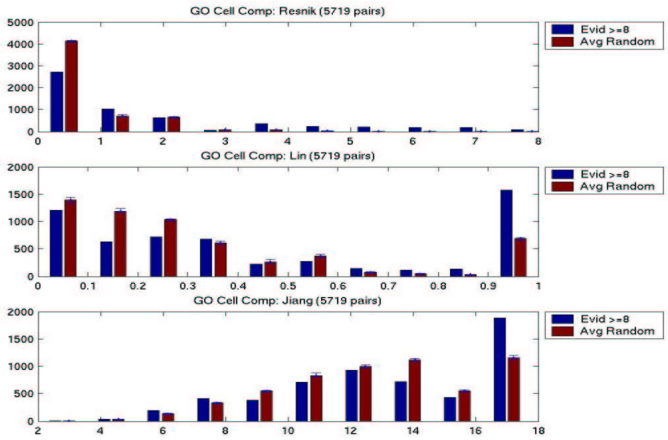


Figure D.4: Summary of Separation of GO Semantic Similarity Distributions per Expert

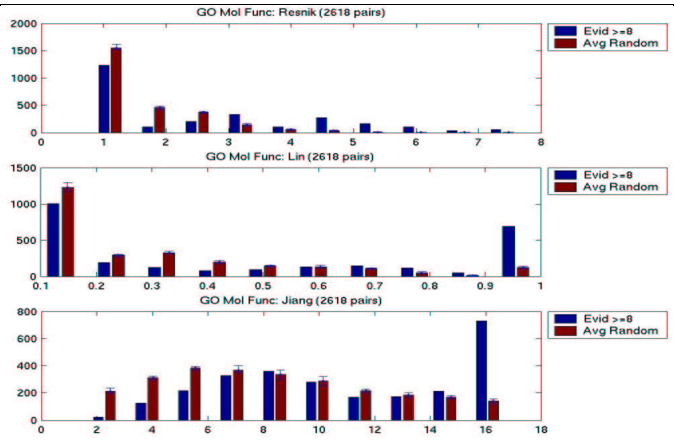
bottom vertically). The color at each position in the matrix indicates the corresponding number of pairs $C(i,j)$ where *hotter* colors indicate a large number of pairs. In the title of each plot, we give the total number of pairs with at least one piece of evidence (pairs total) and the maximum number of pairs found in a cell of the matrix C (given in parentheses). The objective is to have a cluster of hotter colors near the lower right corner of each heatmap, indicating that more pieces of evidence corresponds to higher scores.

Figure D.6 shows the heatmap for GO Cellular Component for RESNIK, LIN, and JIANG, from left to right respectively. Superimposed on the figure is the value of the maximum $C(i,j)$ for each matrix. For example, the RESNIK picture shows a maximum of 293 pairs with 7 pieces of Positive evidence and RESNIK scores ranging around 0.77. For the JIANG measure, we see a maximum of 9 pieces of evidence for 346 pairs with JIANG similarities around 17.7.

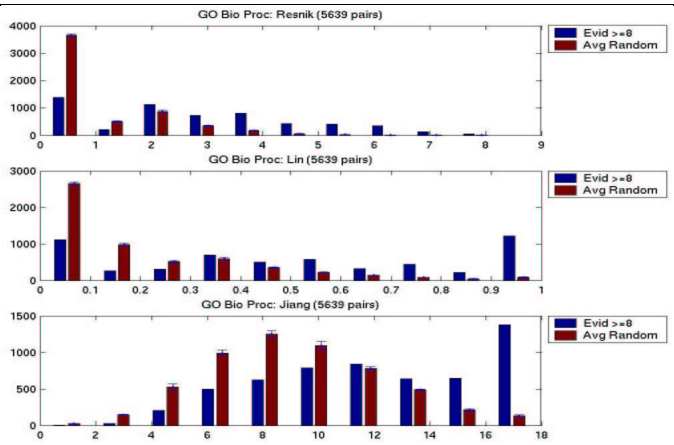
Figure D.7 and Figure D.8 show analogous plots for the GO branches describing Molecular Function and Biological Process, respectively. We can see for the collection of figures, that the displays for JIANG typically have a cloud of color nearer to the lower right than RESNIK or LIN. The LIN results show a band of color along the bottom of each plot yet the remainder of the cloud of color generally is in the lower range of similarity scores. The presence of this nearly bimodal distribution for LIN can also be seen in the graphs of Figure D.3.



(a) Cellular Component



(b) Molecular Function



(c) Biological Process

Figure D.5: Average Similarity from Random Samples of Pairs versus Pairs with Evidence from Any Interaction Source

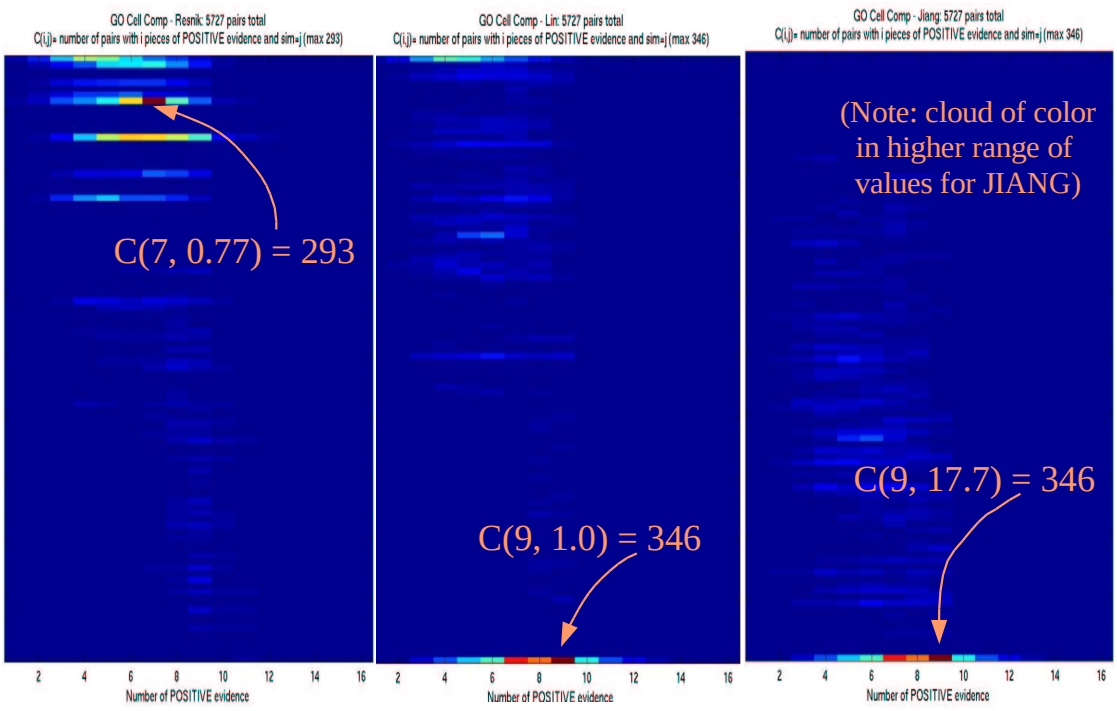


Figure D.6: Number of Pieces of Positive Evidence versus GO Similarity Score: Cellular Component

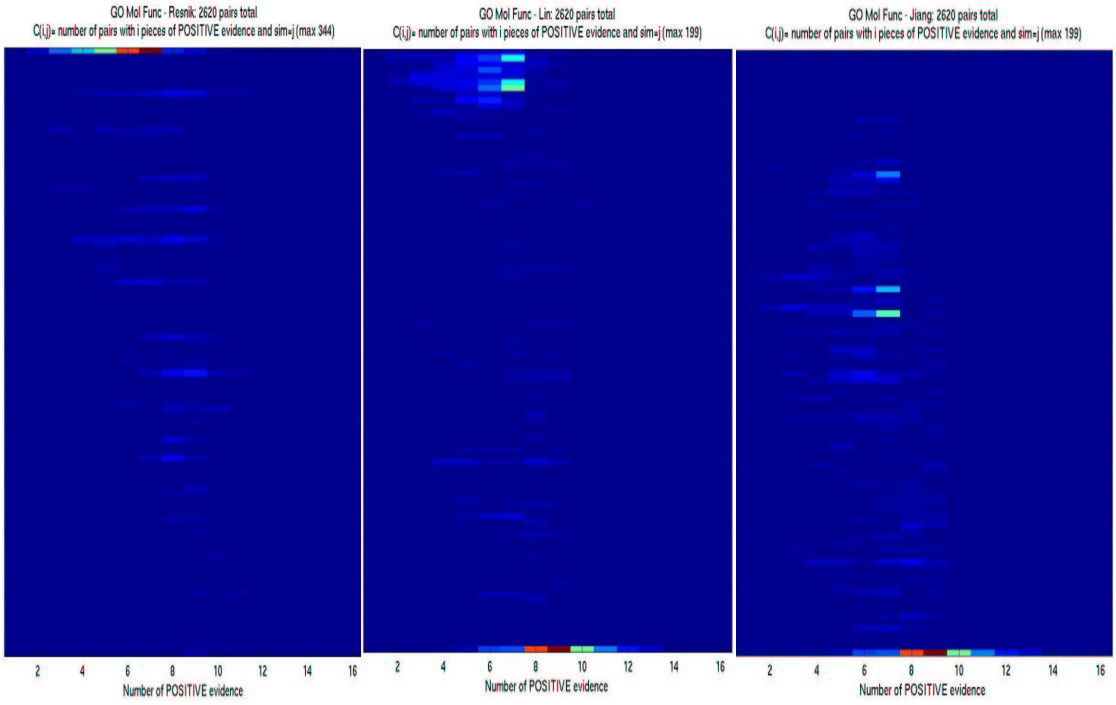


Figure D.7: Number of Pieces of Positive Evidence versus GO Similarity Score: Molecular Function

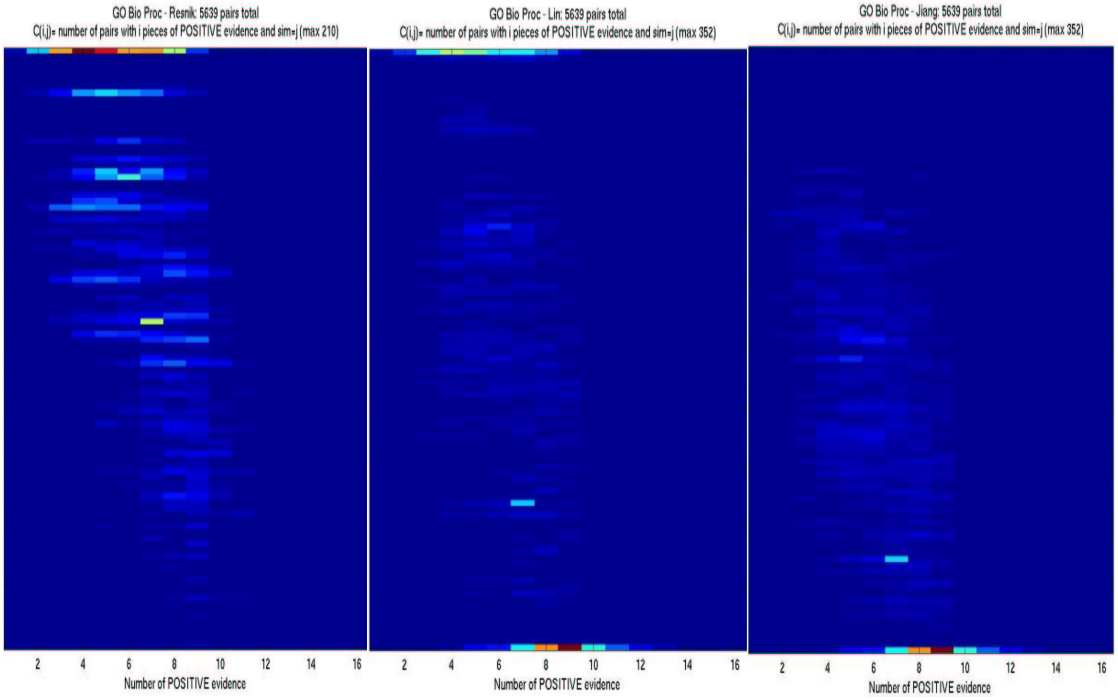


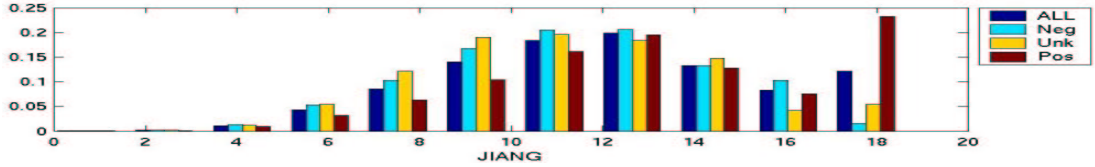
Figure D.8: Number of Pieces of Positive Evidence versus GO Similarity Score: Biological Process

D.4 Conclusions

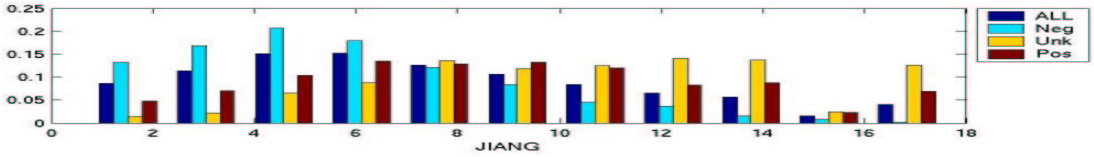
Overall, we conclude that high GO semantic similarity scores are correlated with the presence of interaction information. We find that JIANG is the most discriminative measure. This is in contrast to the similar study of Lord *et al.* [LSBG03] who investigated the correlation with protein sequence similarity where RESNIK was found to be the best, as well as the study by Wang *et al.* [WABD04] where the relationship of similarity scores to gene expression correlation values was indifferent to the metric used.

One additional point of interest is to investigate the correspondence for those experts created from the MIPS catalogues [MFG+02] with information that intuitively overlaps that in GO. We would expect the correlation between indication of interaction and similarity score to be high between the COLOC expert and GO:CC, and high for COFUNC versus GO:MF and GO:BP. Figure D.9 shows the correspondence of the appropriate GO branch with the MIPS catalogue expert using the JIANG scores. We see that GO:CC versus MIPS localization is semi-separable, while GO:MF and GO:BP versus MIPS function is poorly to semi-separable. This indicates that these sources actually have little overlap of information. Further investigation (data not shown) suggests that the issue stems from the granularity of the GO terms. For this reason, we consider the GO catalogues as experts distinct from the others when creating consensus likelihood functions for experts in the yeast and mouse genomes.

$ GO:CC vs MIPS COLOC



$ GO:MF vs MIPS FUNCREL



$ GO:BP vs MIPS FUNCREL

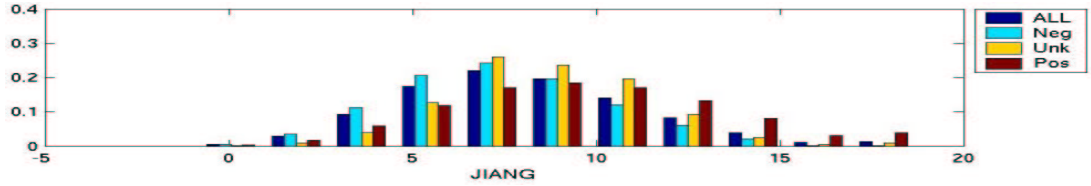


Figure D.9: GO Similarity versus MIPS Catalogues

Appendix E

Individual Results for ALARM Bayesian Network Learning

As described in Section 6.1, we create three types of experts (Positive, Negative, and Indirect) and five different sets of reliability assignments. Recall that each assignment is represented by a three integer string N.I.P where N, I and P represent the reliability levels for the set of Negative, Indirect and Positive experts, respectively. Low, Medium, or High reliability is represented in the integer string by a 0, 1, or 2, respectively. The meaning of each assignment is repeated below, appearing in rough order of decreasing correctness of reliability assigned:

1. **0.0.2:** Positive experts correctly assigned High reliability
2. **0.1.2:** Reliabilities reflect intuitive understanding of correctness
3. **0.2.0:** Indirect experts given most influence
4. **1.1.1:** All experts assigned uniform reliability
5. **2.1.0:** All experts assigned incorrect reliability

For the comparisons, 100 models were sampled from an informed structural prior distribution, formulated using the LinOP or NoisyOR consensus belief function combined with one of the five reliability assignments, to create a total of 10 sets of Informed models. Additionally 100 models were sampled cocurrently from a uniform prior over structure, each set of which we denote as Random. Greedy hill-climbing search was applied to each of the starting models, recording the score components $Pr(S)$ and $Pr(D|S)$ at each iteration of the search.

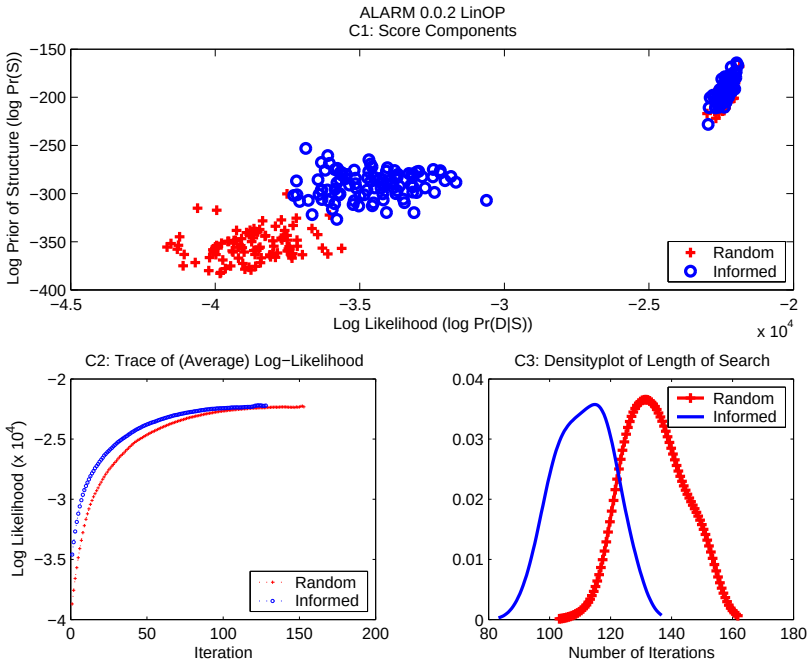


Figure E.1: ALARM 0.0.2 LinOP

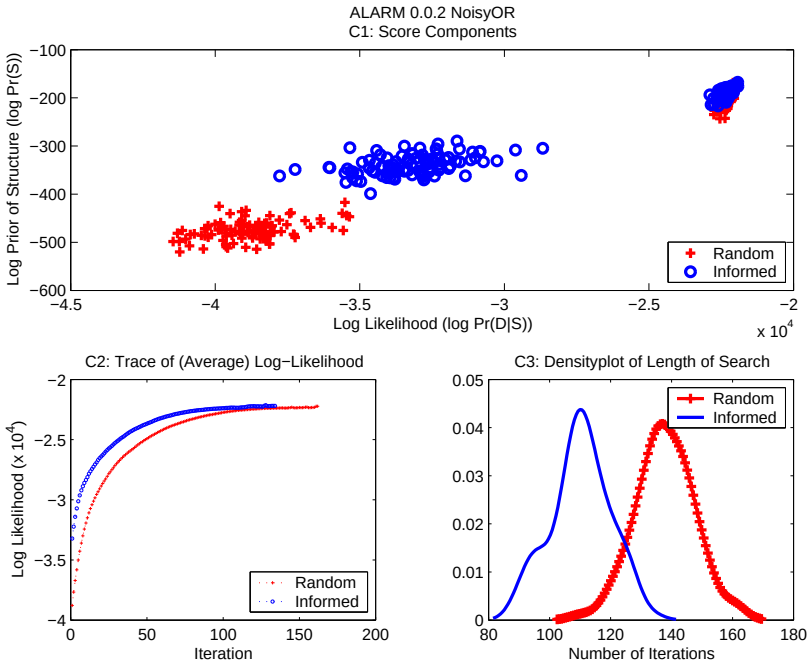


Figure E.2: ALARM 0.0.2 NoisyOR

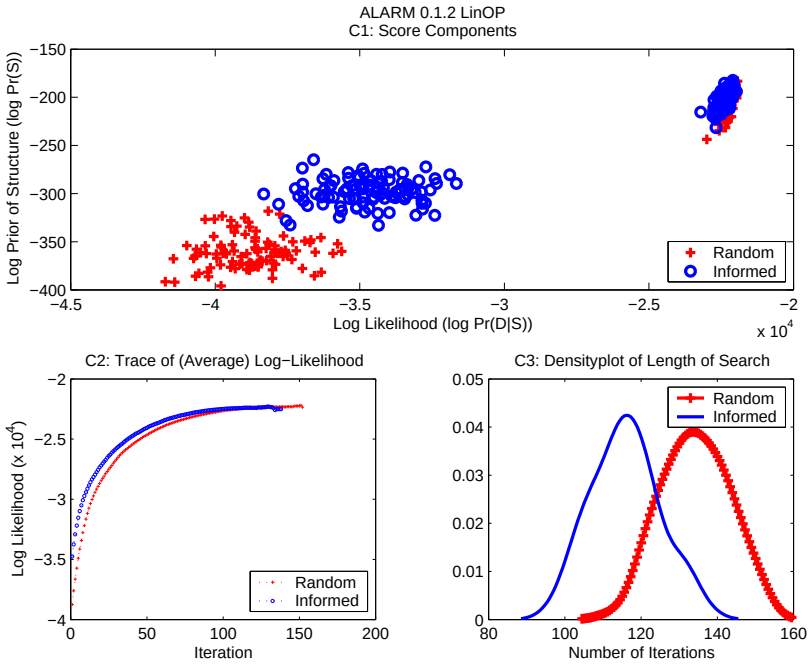


Figure E.3: ALARM 0.1.2 LinOP

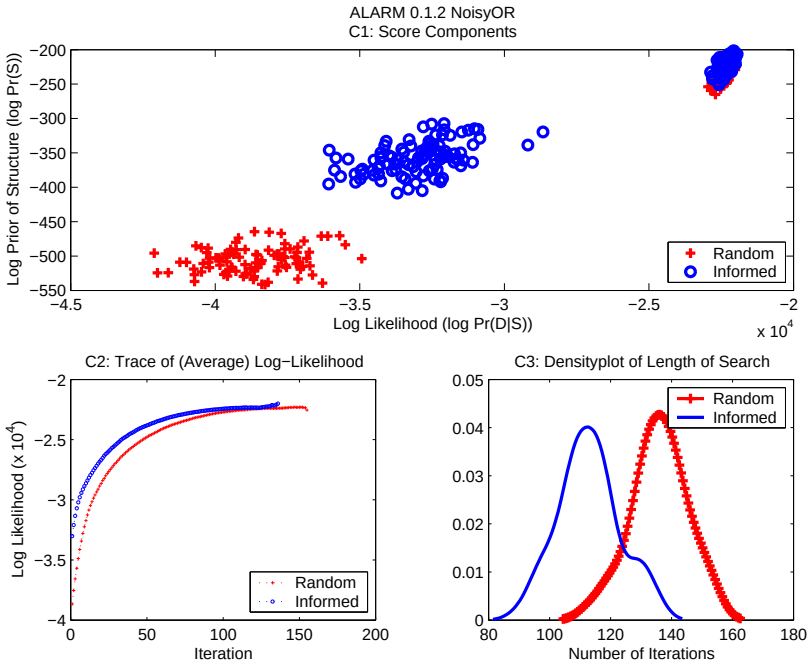


Figure E.4: ALARM 0.1.2 NoisyOR

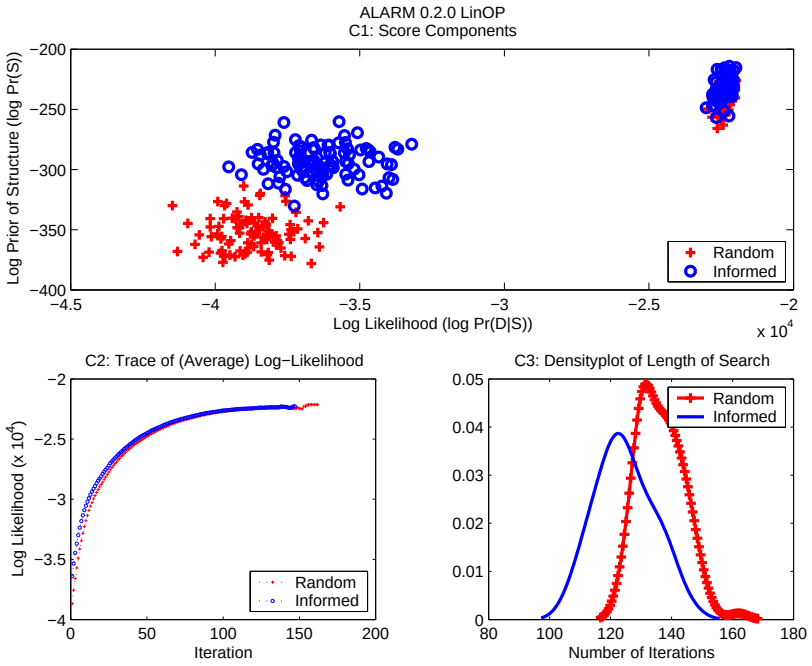


Figure E.5: ALARM 0.2.0 LinOP

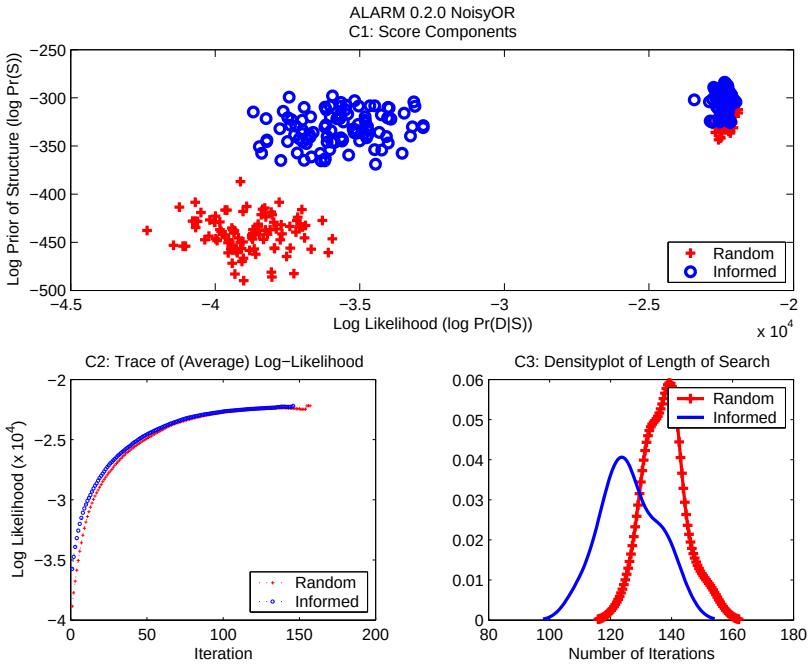


Figure E.6: ALARM 0.2.0 NoisyOR

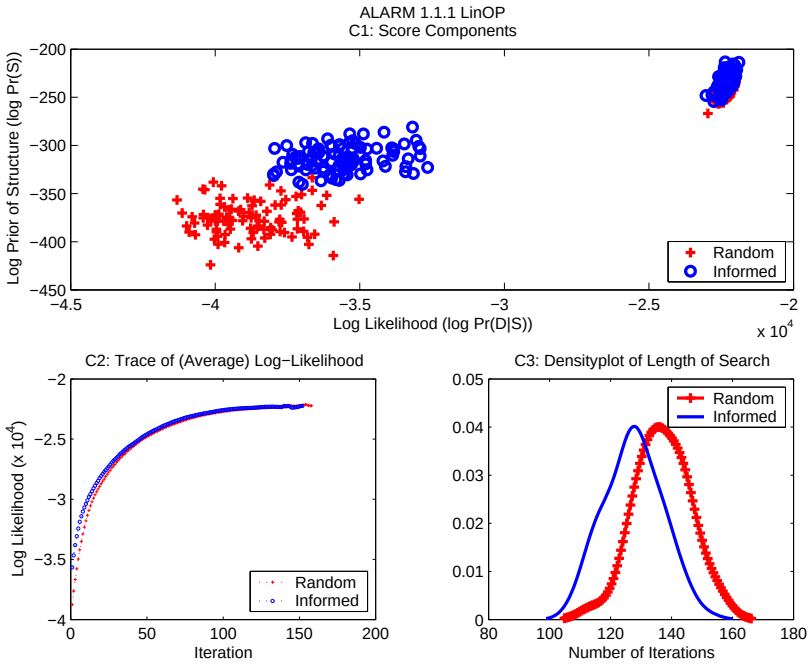


Figure E.7: ALARM 1.1.1 LinOP

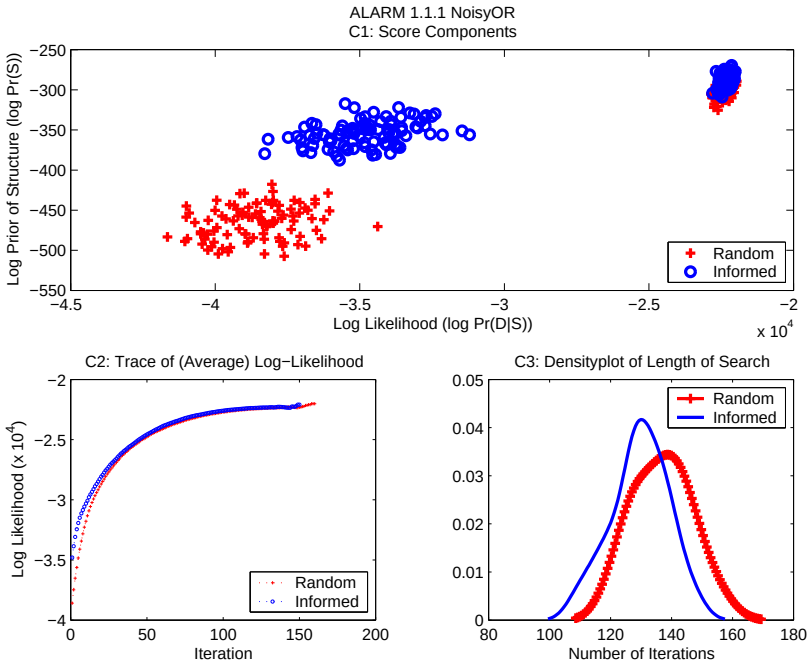


Figure E.8: ALARM 1.1.1 NoisyOR

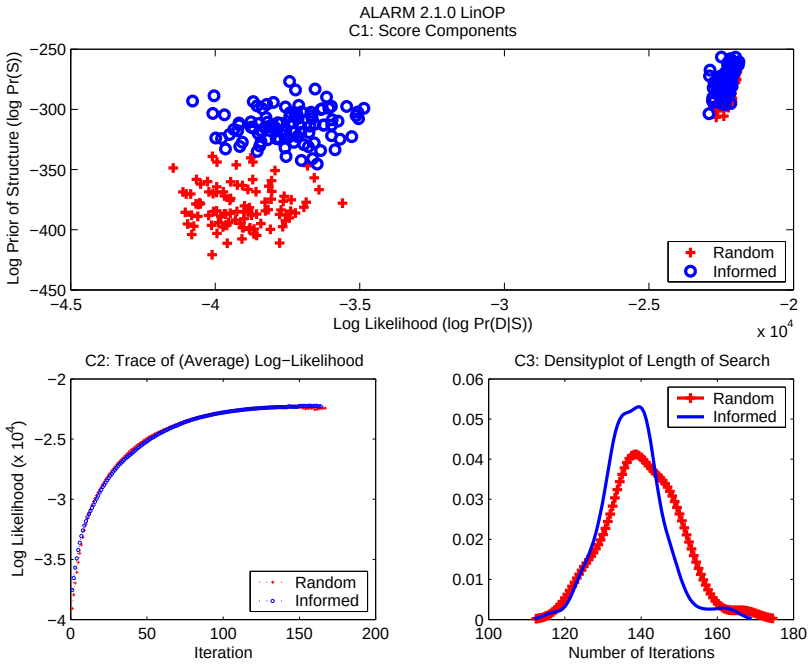


Figure E.9: ALARM 2.1.0 LinOP

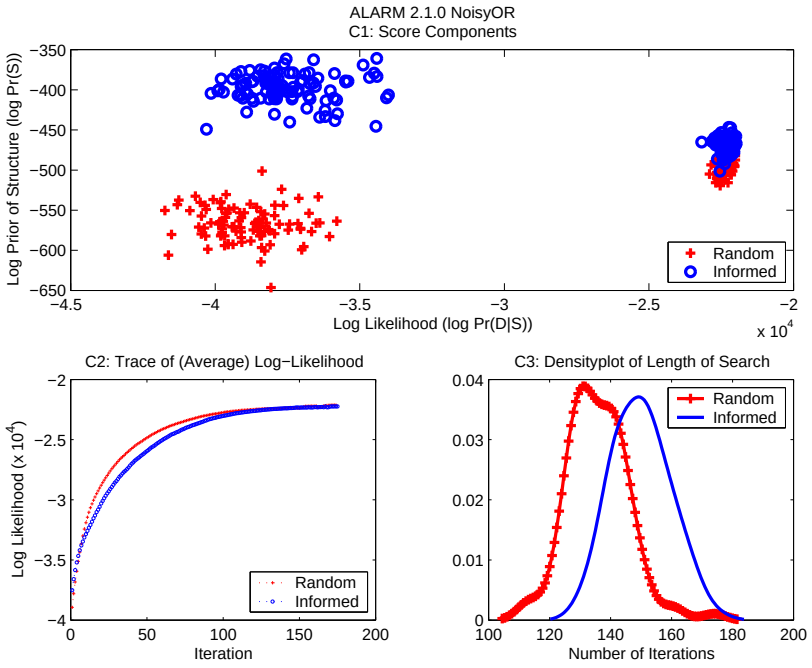


Figure E.10: ALARM 2.1.0 NoisyOR

Appendix F

Individual Results for IDEKER50 YEAST Bayesian Network Learning

The complete results for the individual evaluation criterion are shown in this chapter for all 24 consensus likelihood functions. Two datasets are used for Bayesian network learning, STD and MAG1-IDRR, as described in Section 7.3. For the BDeu metric, two settings are used for the equivalent sample size, namely $N' = 1$ and $N' = 10$ for each dataset, for a total of four different combinations. For STD $N' = 1$, we sample 110 models from each informed structural prior. An additional sample of 110 models is created from a uniform structural prior; the same 110 Random models is used in all comparisons of Informed versus Random. For the remaining learning combinations, we sample 50 models from either Informed or Random, again using the same set of Random models for all comparisons.

For the LinOP and NoisyOR consensus likelihood functions, we use the reliability assignments **MAIR, CONS, UNIF5, RAND1** and **RAND2**, as described in Section 6.2.2. For BAYES, we learn the parameter of the BAYES model from three different datasets, **UNSET**, **SEMRELG0** and **SEMRELG3**, and calculate the consensus likelihood using the **No Ratio** or **Ratio** technique, as given by Eq (5.6). The titles for the BAYES variants denotes which dataset and which method of calculating probabilities was used. For example, the name BAYESUN denotes learning from the **UNSET** dataset using **No ratio** while the name BAYES3R denotes learning from the **SEMRELG3** dataset using the **Ratio** computation. The PRM variants are named similarly based on the dataset used to learn the parameters of the PRM model and the method (*i.e.,* Eq (5.9)) by which the consensus likelihood was calculated.

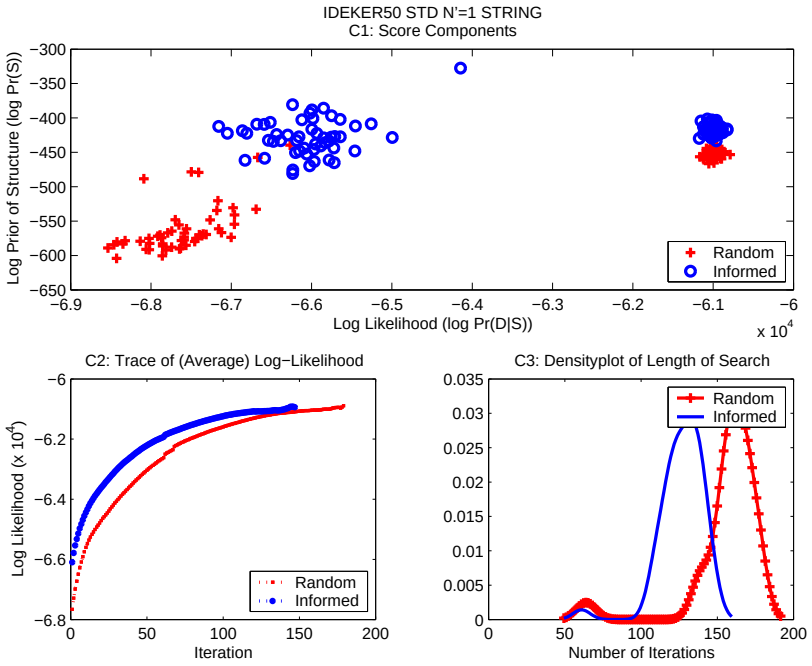


Figure F.1: IDEKER50 STD $N' = 1$ STRING

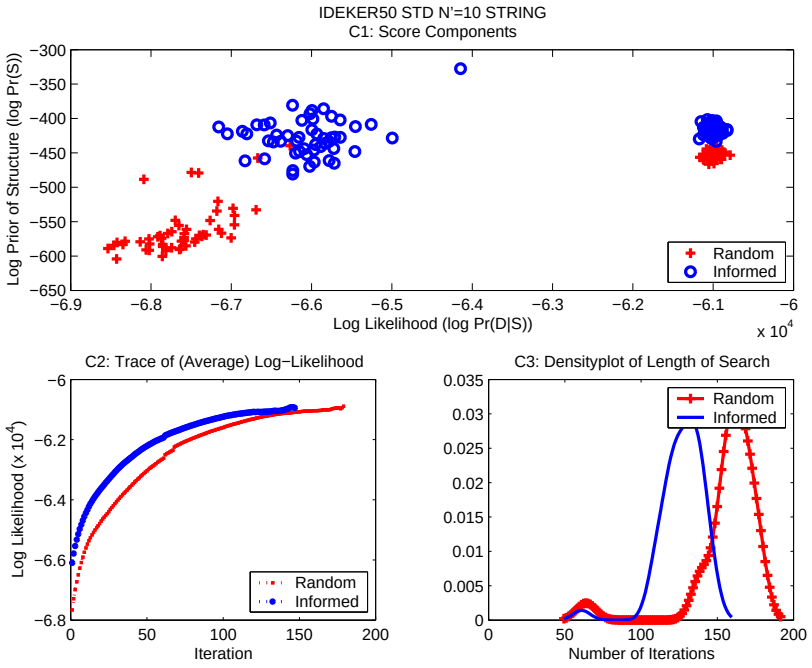


Figure F.2: IDEKER50 STD $N' = 10$ STRING

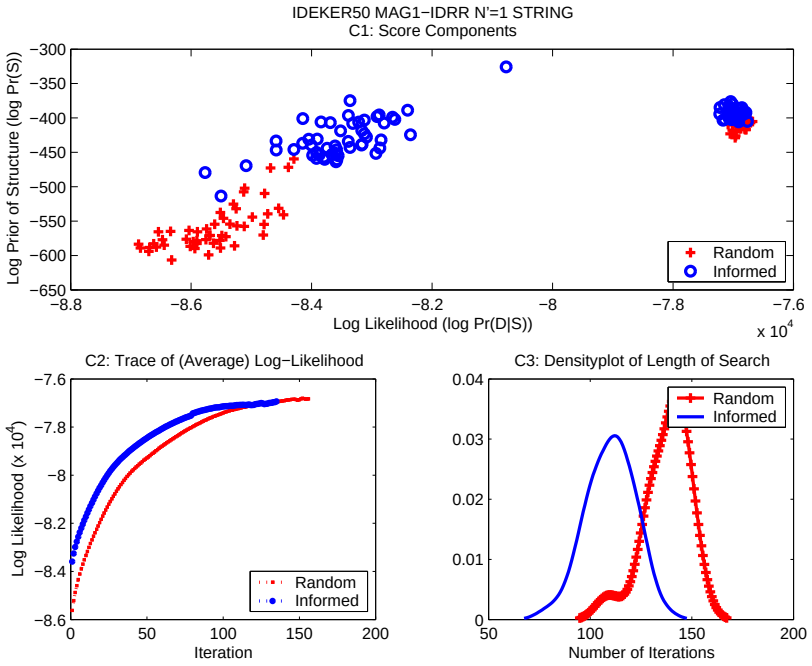


Figure F.3: IDEKER50 MAG1-IDRR $N' = 1$ STRING

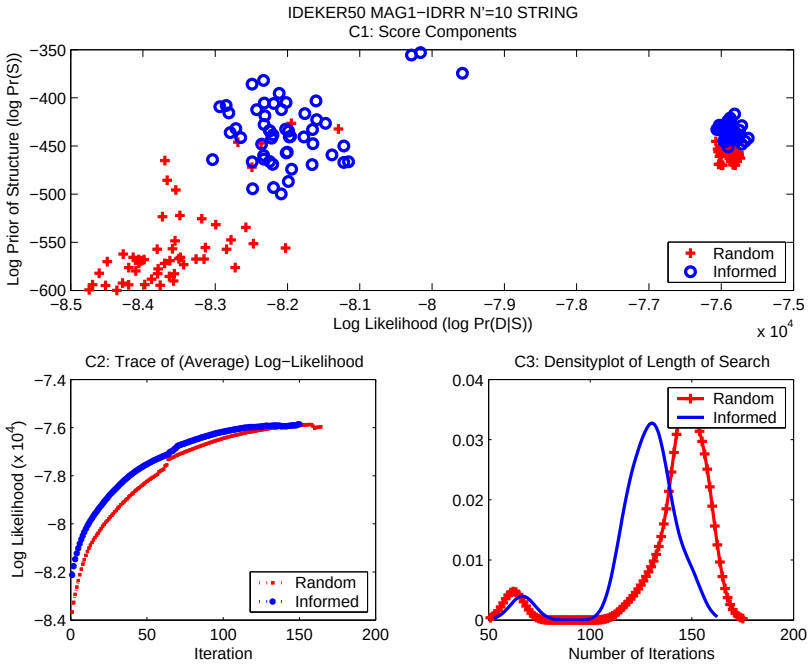


Figure F.4: IDEKER50 MAG1-IDRR $N' = 10$ STRING

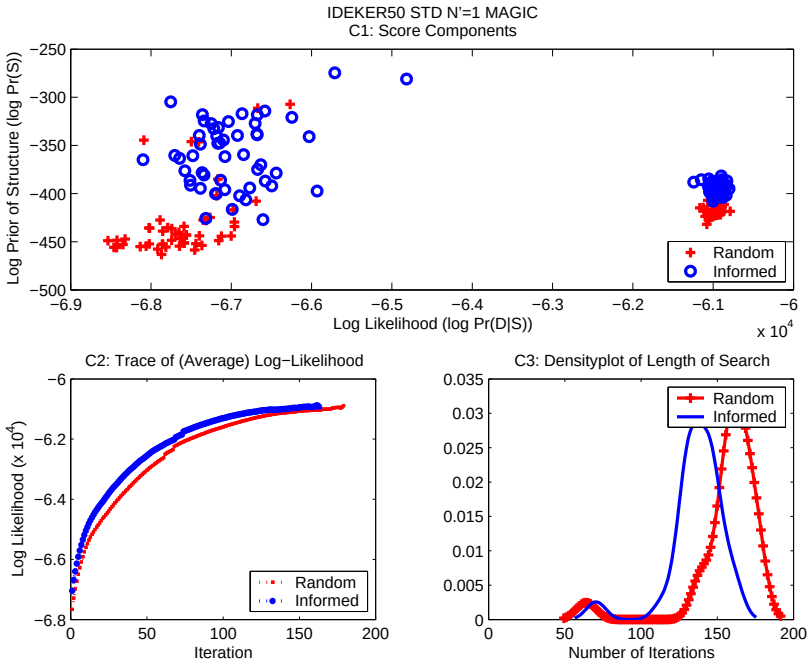


Figure F.5: IDEKER50 STD $N' = 1$ MAGIC

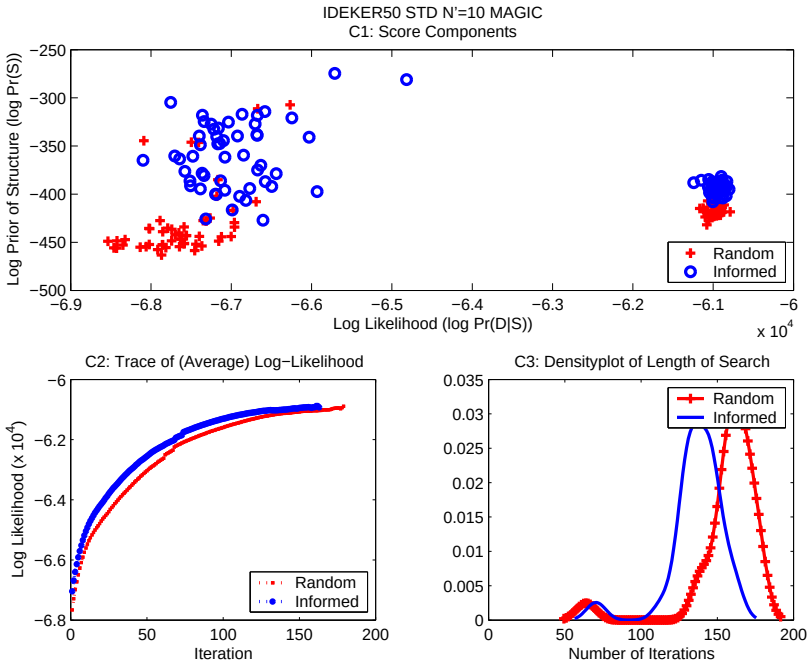


Figure F.6: IDEKER50 STD $N' = 10$ MAGIC

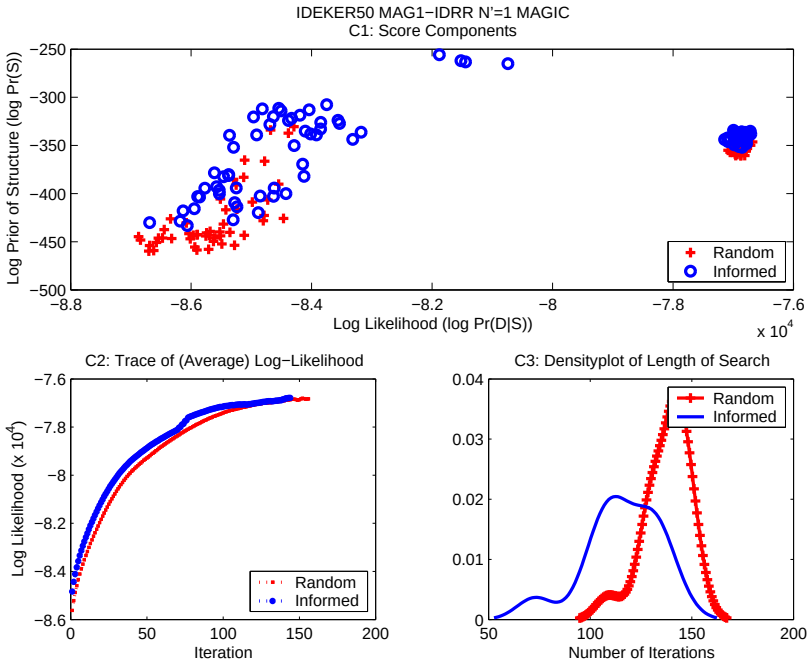


Figure F.7: IDEKER50 MAG1-IDRR $N' = 1$ MAGIC

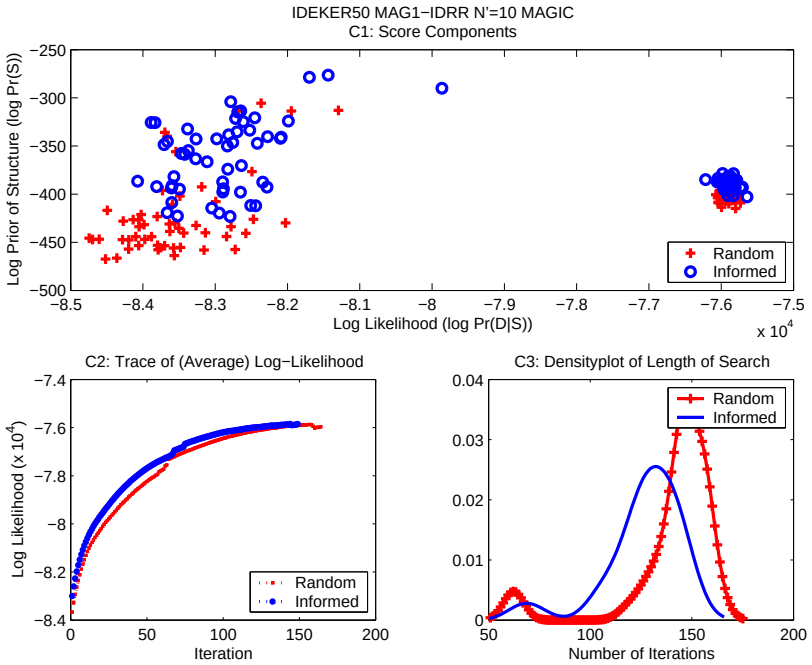


Figure F.8: IDEKER50 MAG1-IDRR $N' = 10$ MAGIC

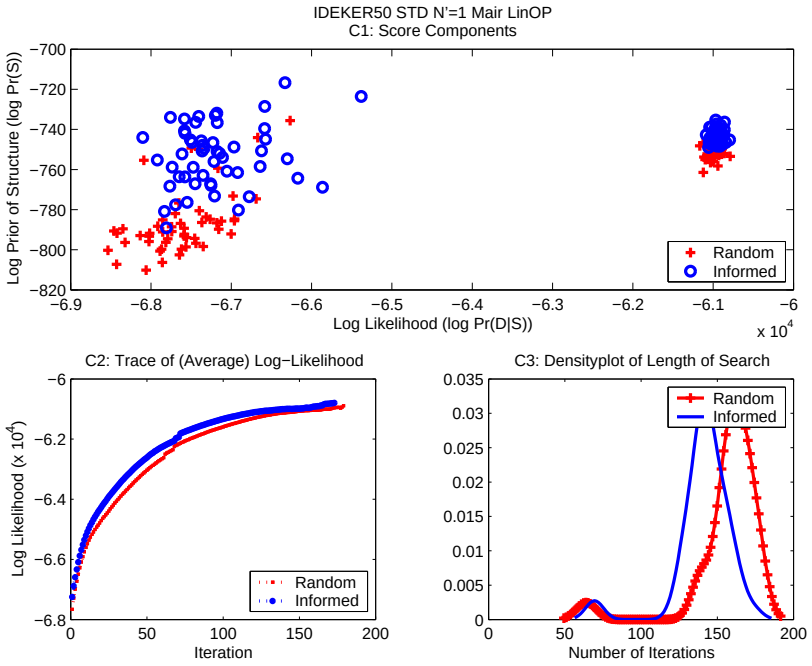


Figure F.9: IDEKER50 STD $N' = 1$ Mair LinOP

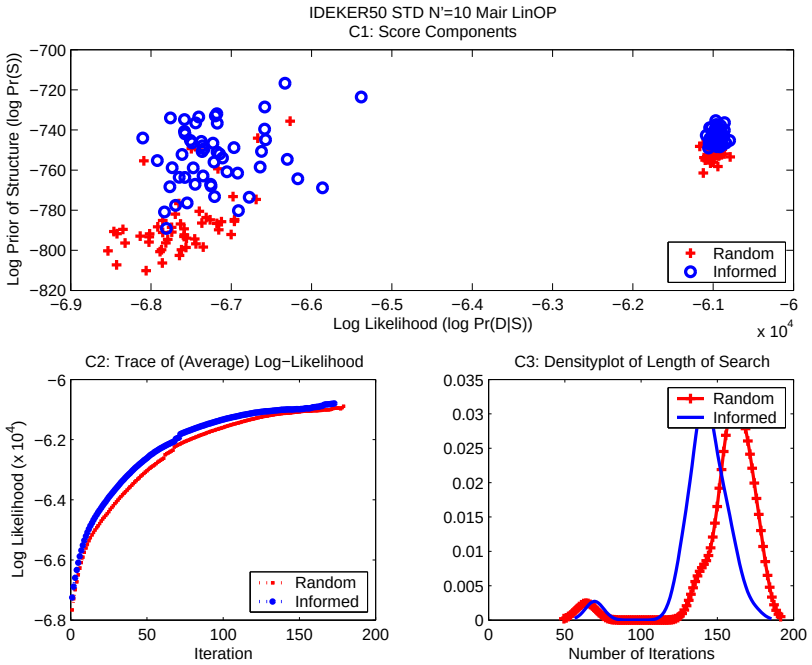


Figure F.10: IDEKER50 STD $N' = 10$ Mair LinOP

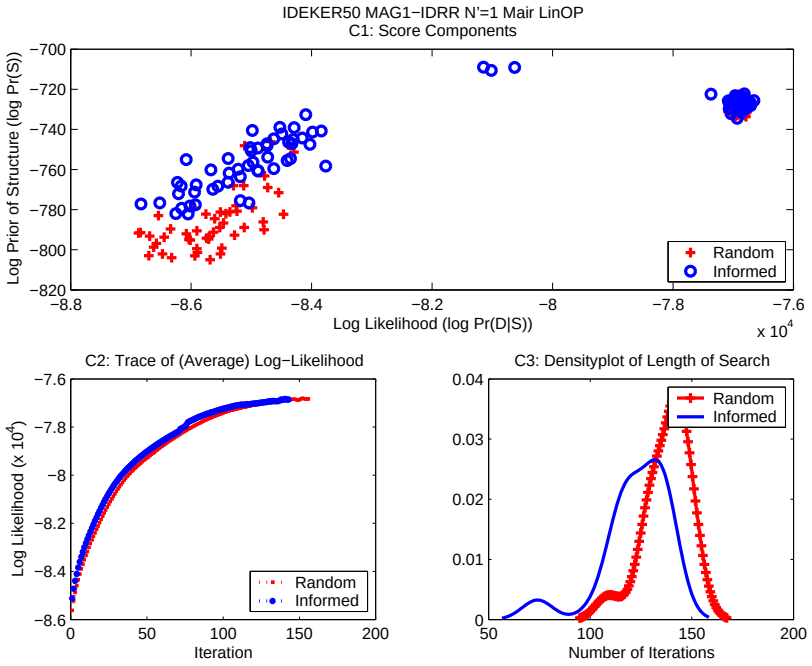


Figure F.11: IDEKER50 MAG1-IDRR $N' = 1$ Mair LinOP

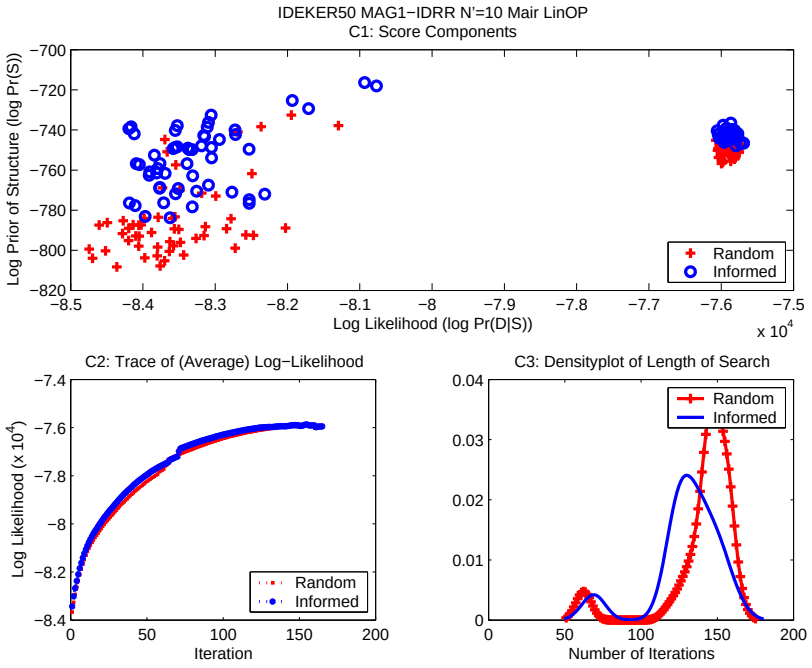


Figure F.12: IDEKER50 MAG1-IDRR $N' = 10$ Mair LinOP

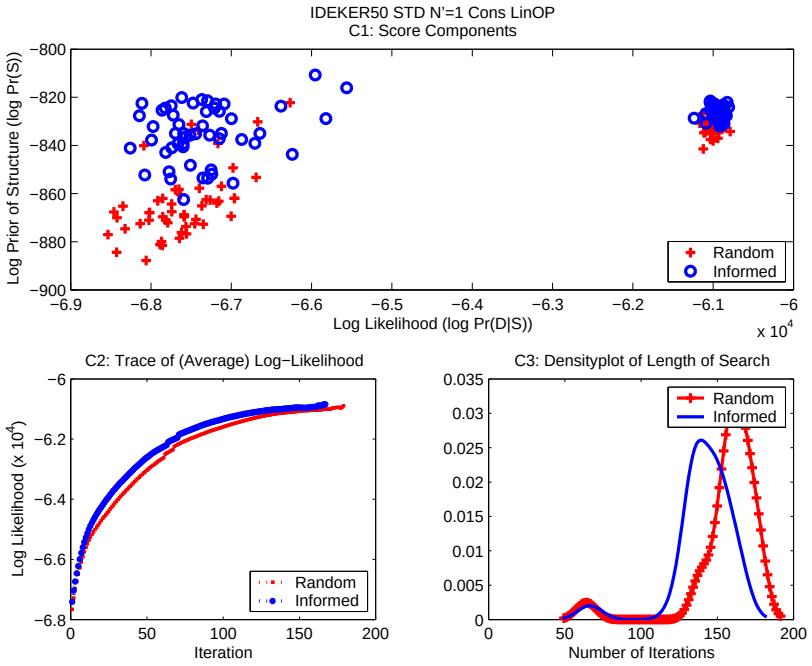


Figure F.13: IDEKER50 STD $N' = 1$ Cons LinOP

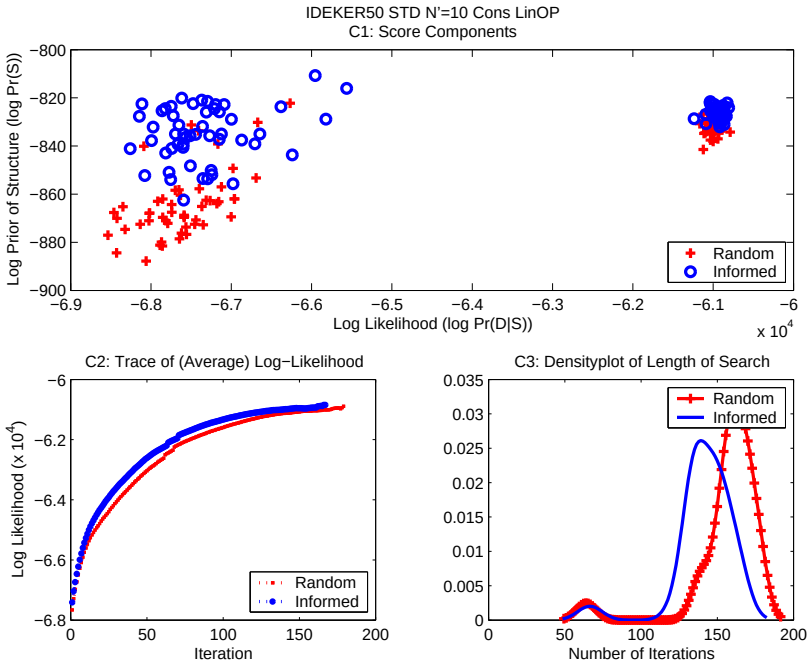


Figure F.14: IDEKER50 STD $N' = 10$ Cons LinOP

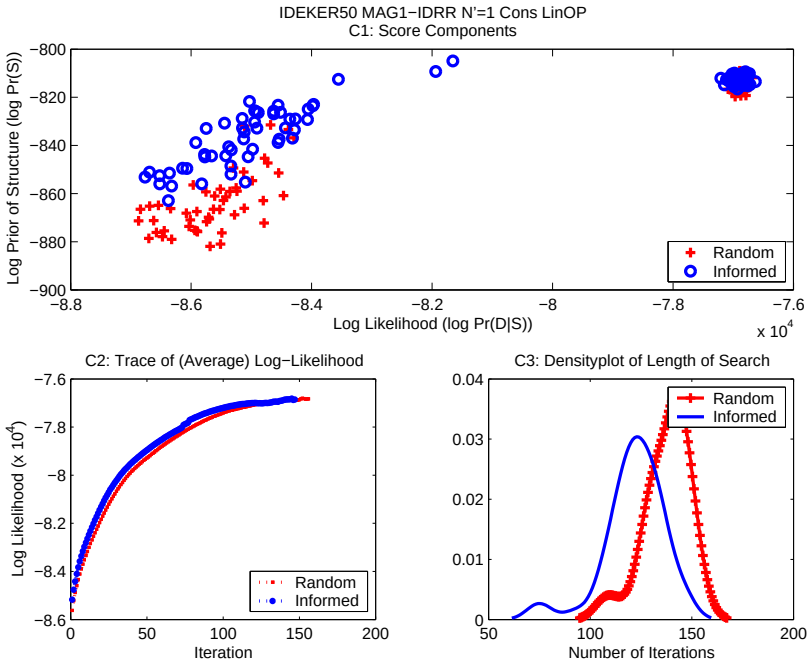


Figure F.15: IDEKER50 MAG1-IDRR $N' = 1$ Cons LinOP

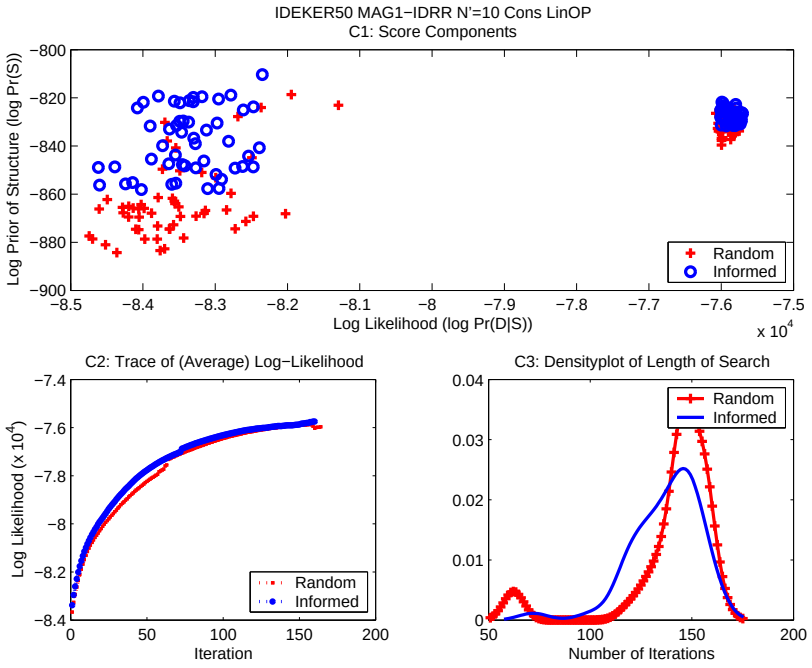


Figure F.16: IDEKER50 MAG1-IDRR $N' = 10$ Cons LinOP

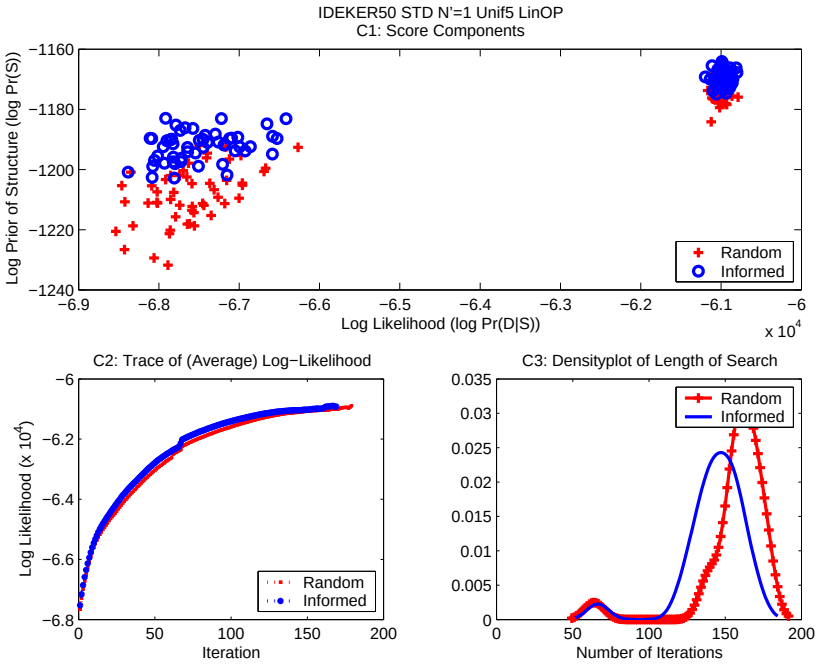


Figure F.17: IDEKER50 STD $N' = 1$ Unif5 LinOP

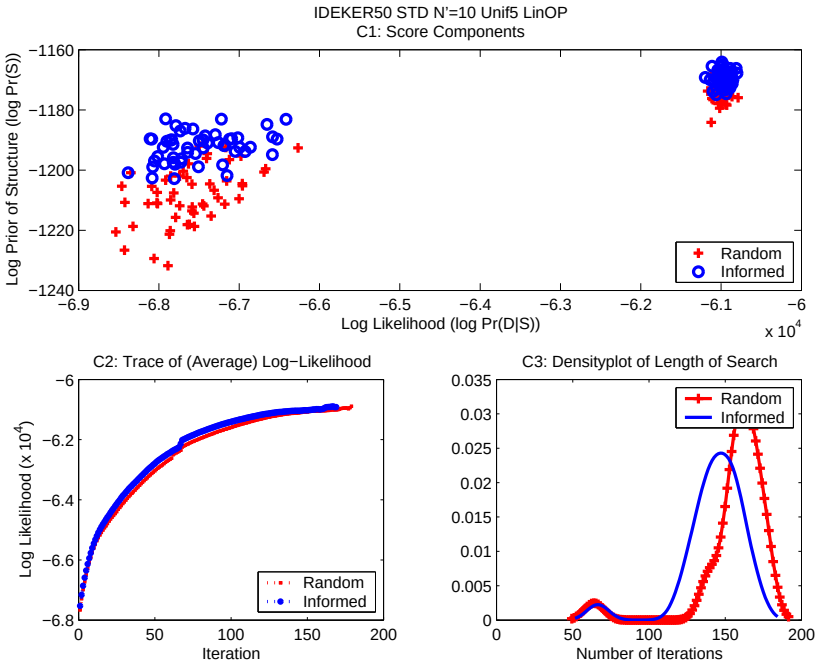


Figure F.18: IDEKER50 STD $N' = 10$ Unif5 LinOP

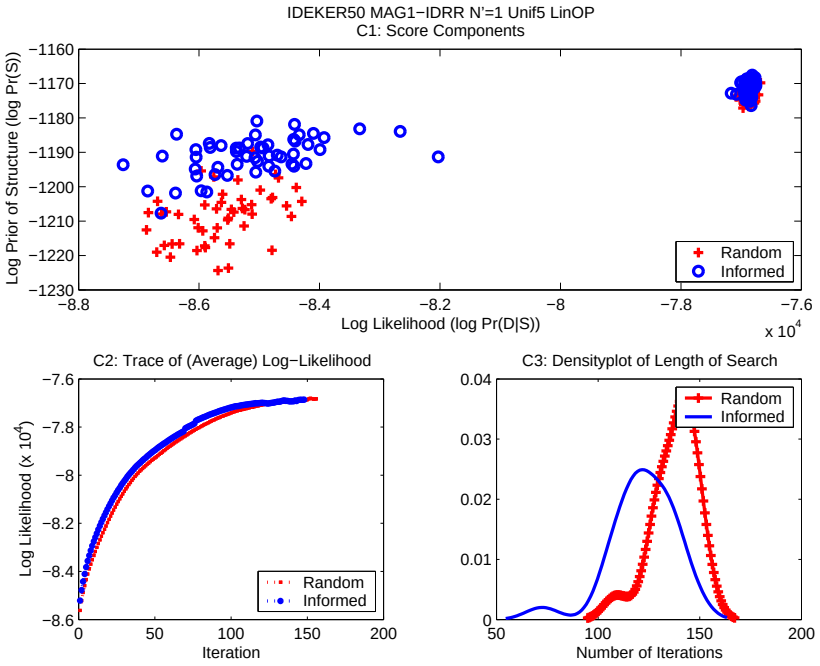


Figure F.19: IDEKER50 MAG1-IDRR $N' = 1$ Unif5 LinOP

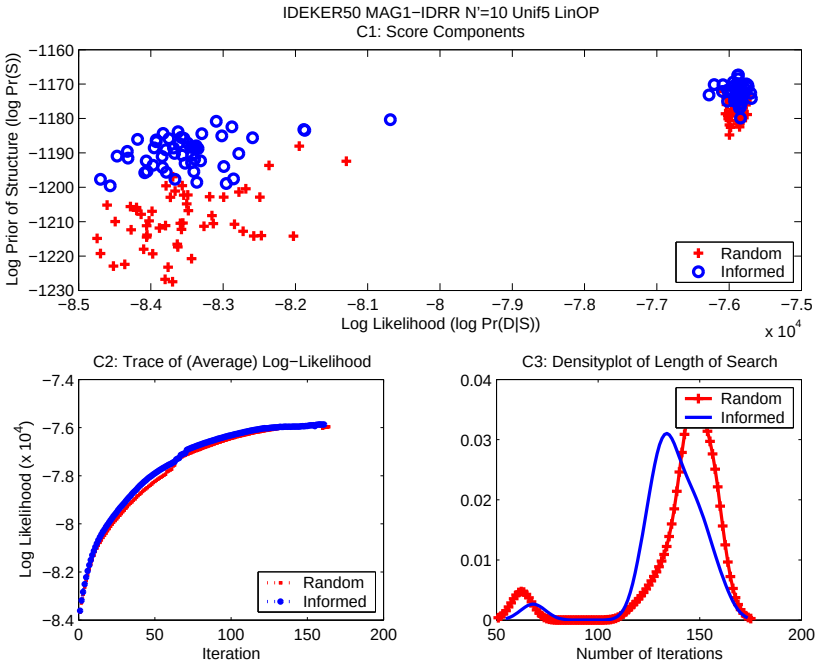


Figure F.20: IDEKER50 MAG1-IDRR $N' = 10$ Unif5 LinOP

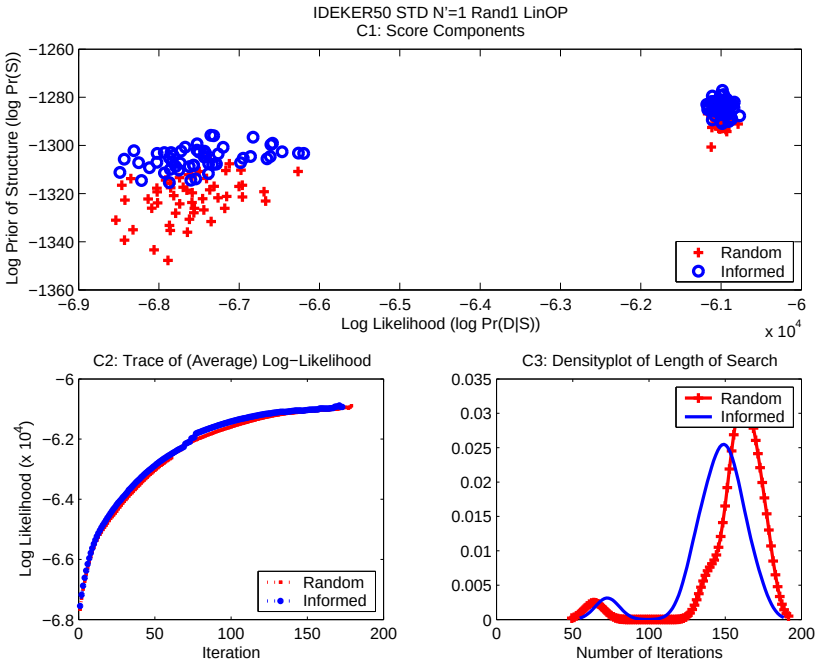


Figure F.21: IDEKER50 STD $N' = 1$ Rand1 LinOP

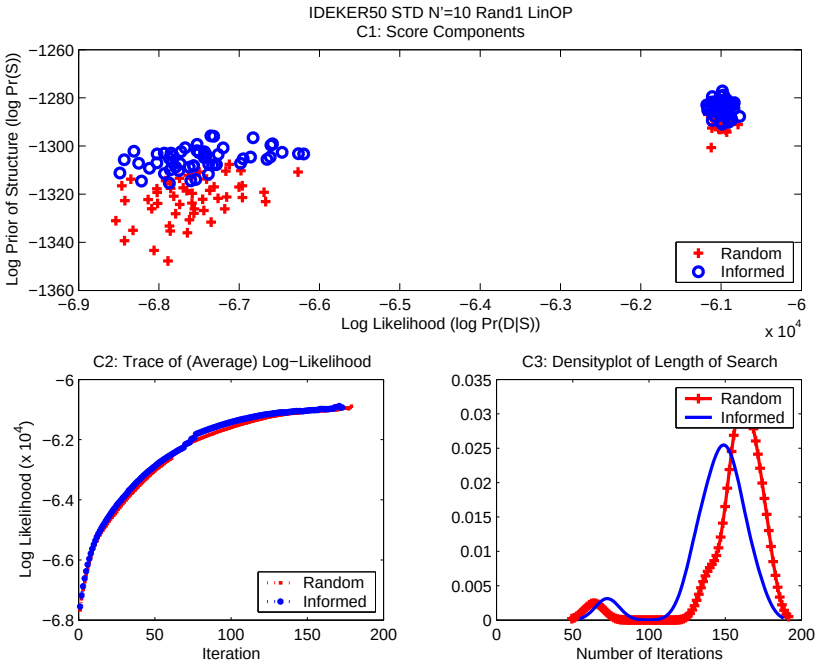


Figure F.22: IDEKER50 STD $N' = 10$ Rand1 LinOP

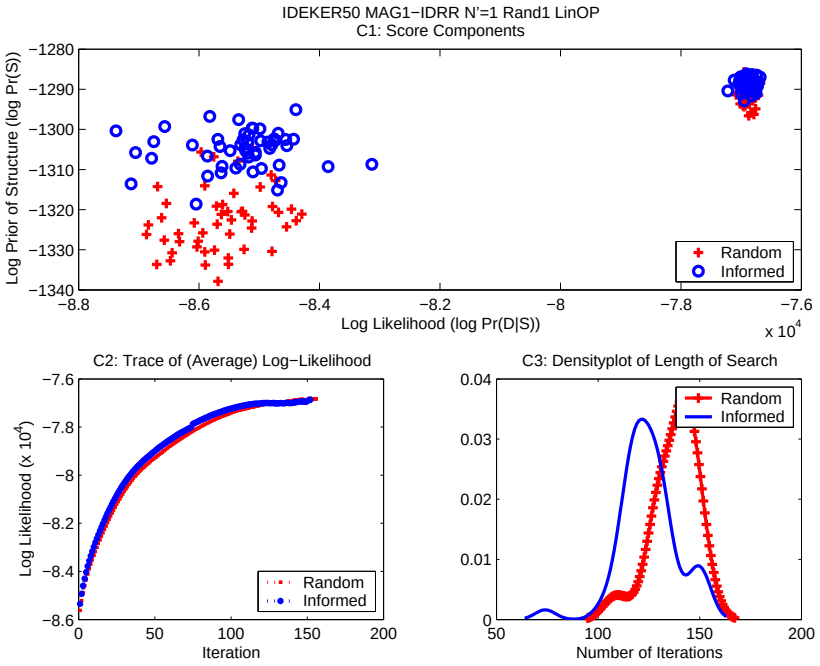


Figure F.23: IDEKER50 MAG1-IDRR $N' = 1$ Rand1 LinOP

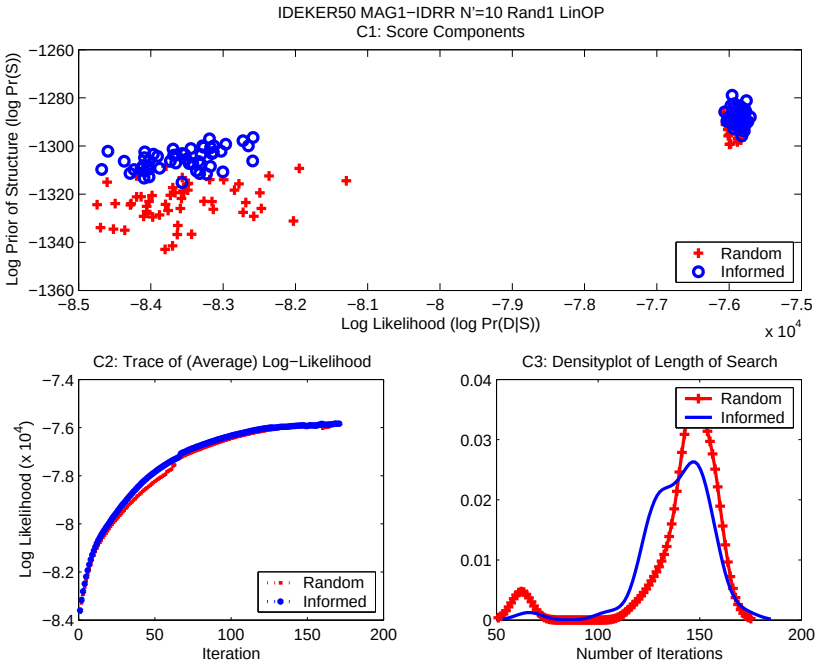


Figure F.24: IDEKER50 MAG1-IDRR $N' = 10$ Rand1 LinOP

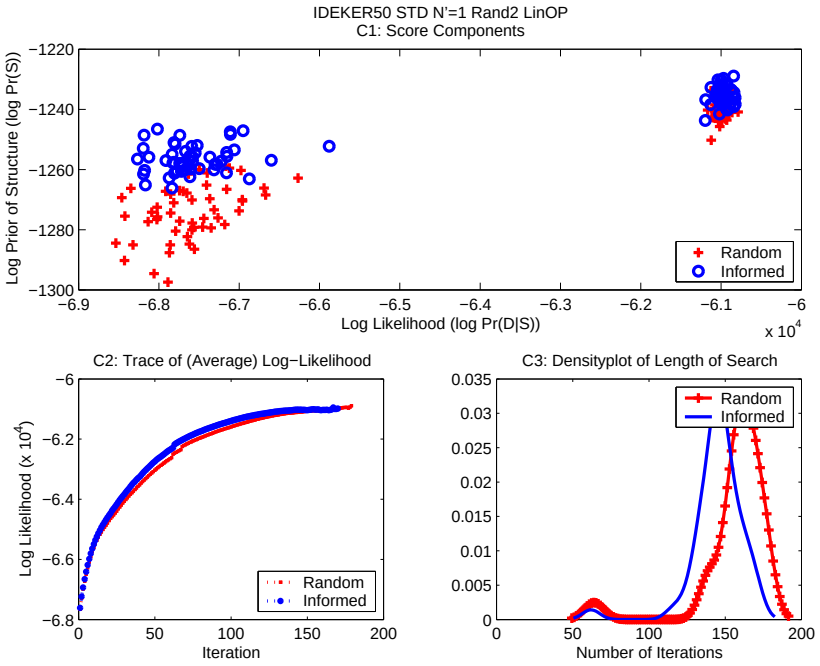


Figure F.25: IDEKER50 STD $N' = 1$ Rand2 LinOP

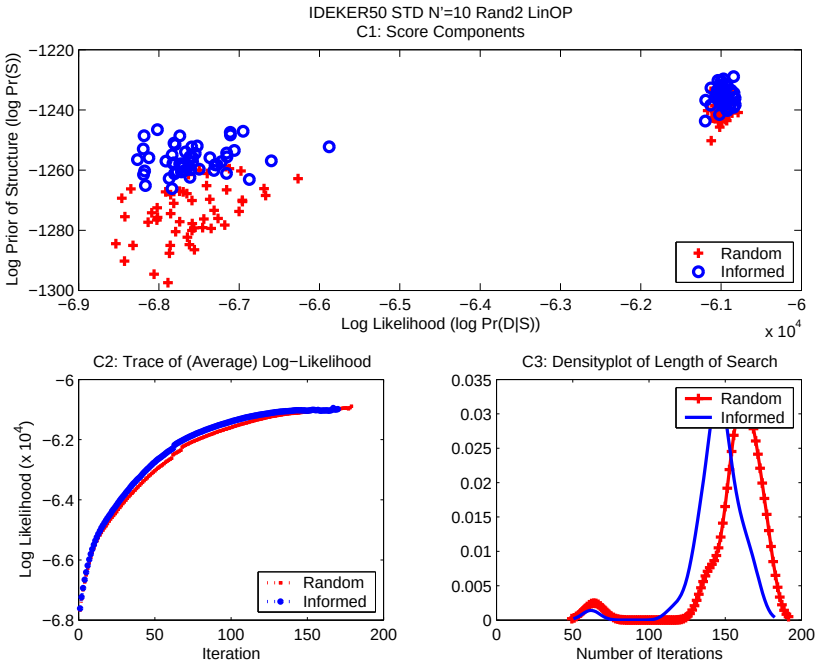


Figure F.26: IDEKER50 STD $N' = 10$ Rand2 LinOP

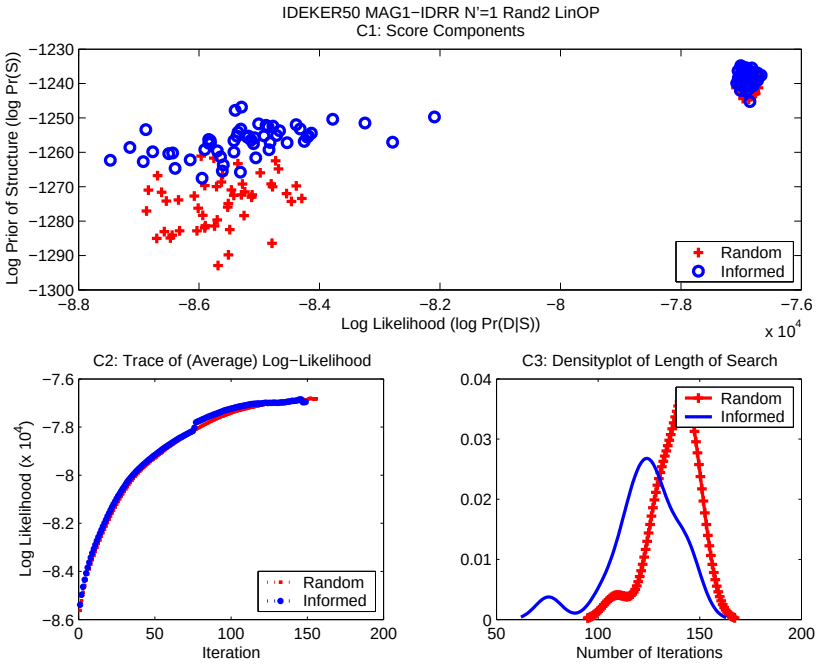


Figure F.27: IDEKER50 MAG1-IDRR $N' = 1$ Rand2 LinOP

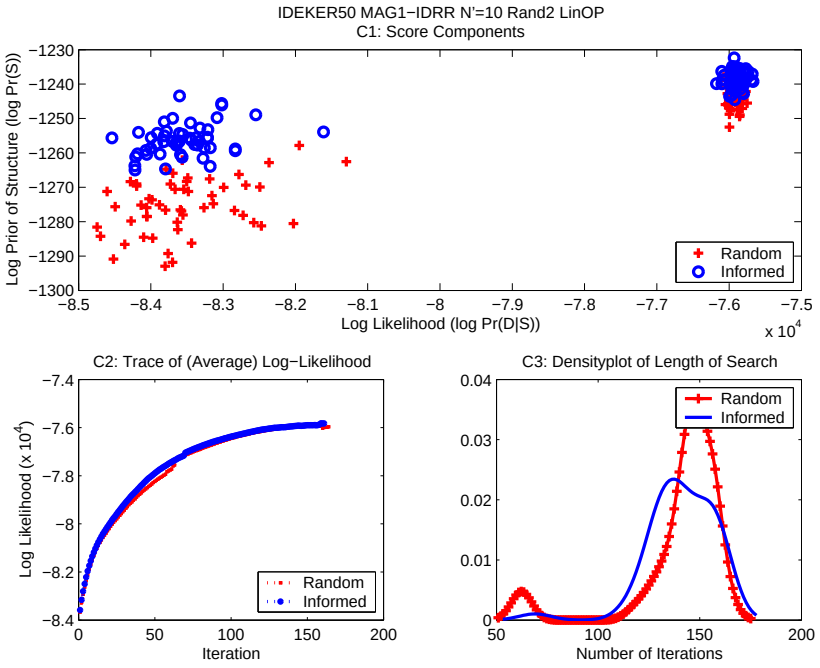


Figure F.28: IDEKER50 MAG1-IDRR $N' = 10$ Rand2 LinOP

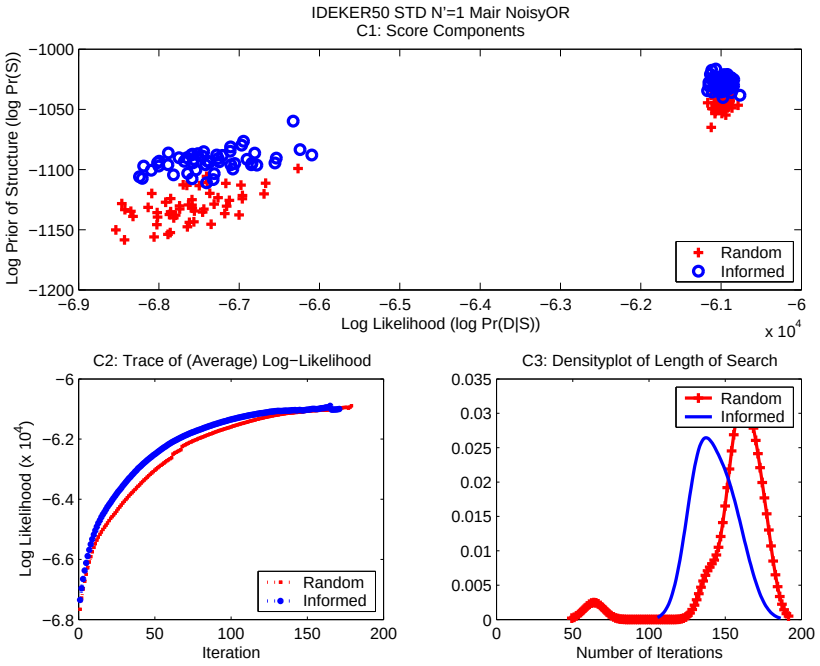


Figure F.29: IDEKER50 STD $N' = 1$ Mair NoisyOR

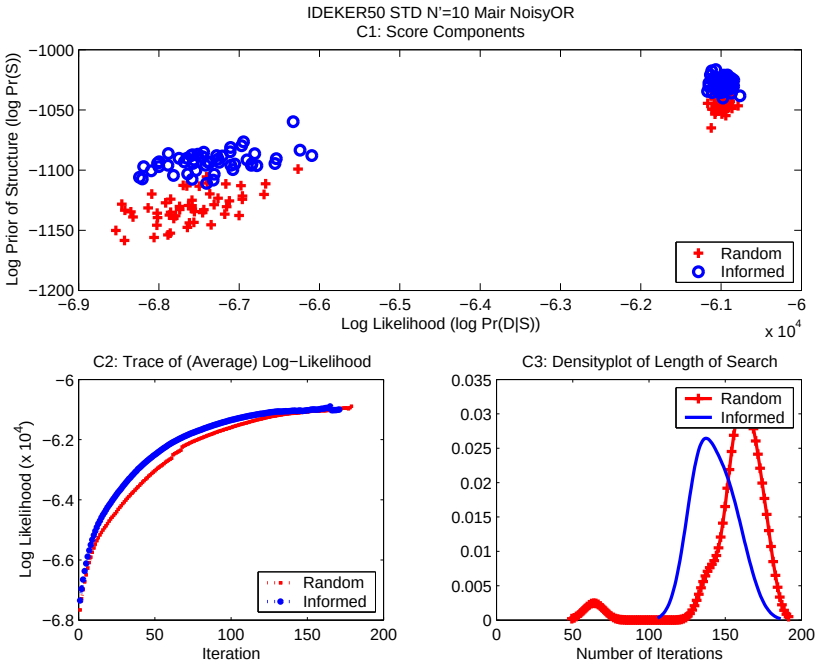


Figure F.30: IDEKER50 STD $N' = 10$ Mair NoisyOR

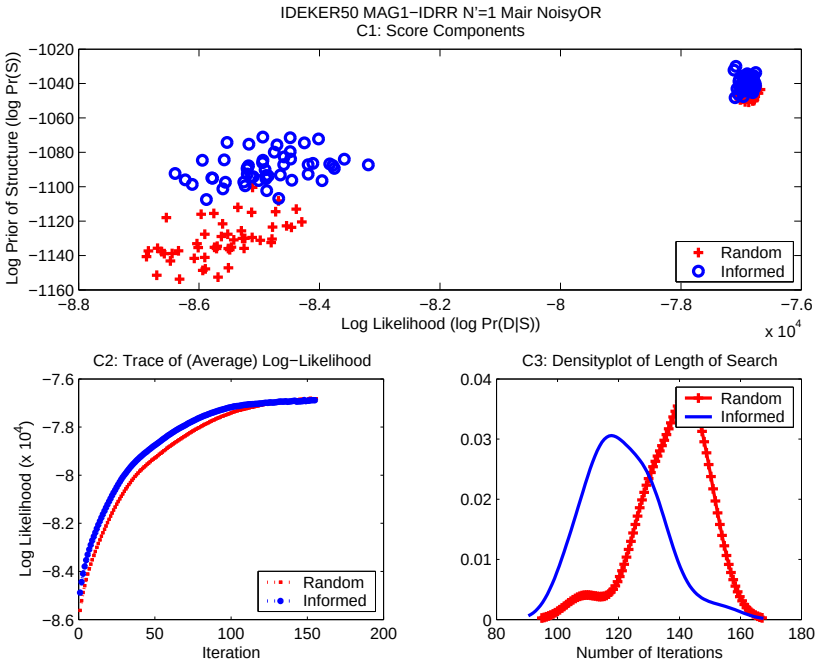


Figure F.31: IDEKER50 MAG1-IDRR $N' = 1$ Mair NoisyOR

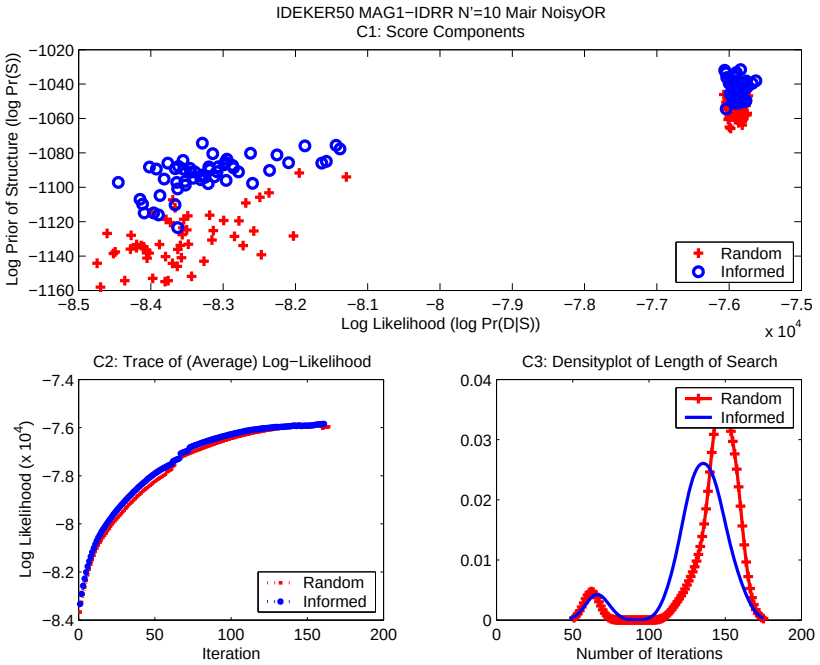


Figure F.32: IDEKER50 MAG1-IDRR $N' = 10$ Mair NoisyOR

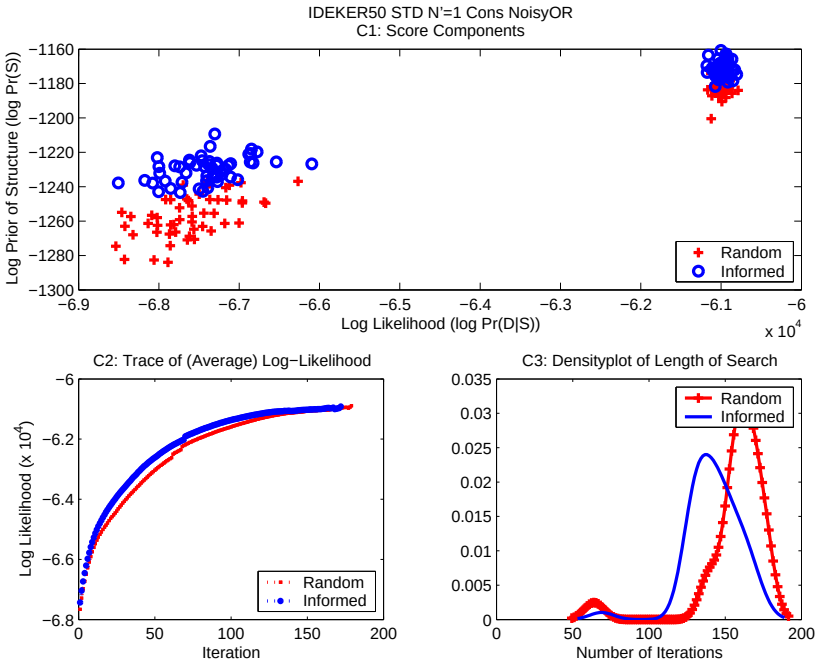


Figure F.33: IDEKER50 STD $N' = 1$ Cons NoisyOR

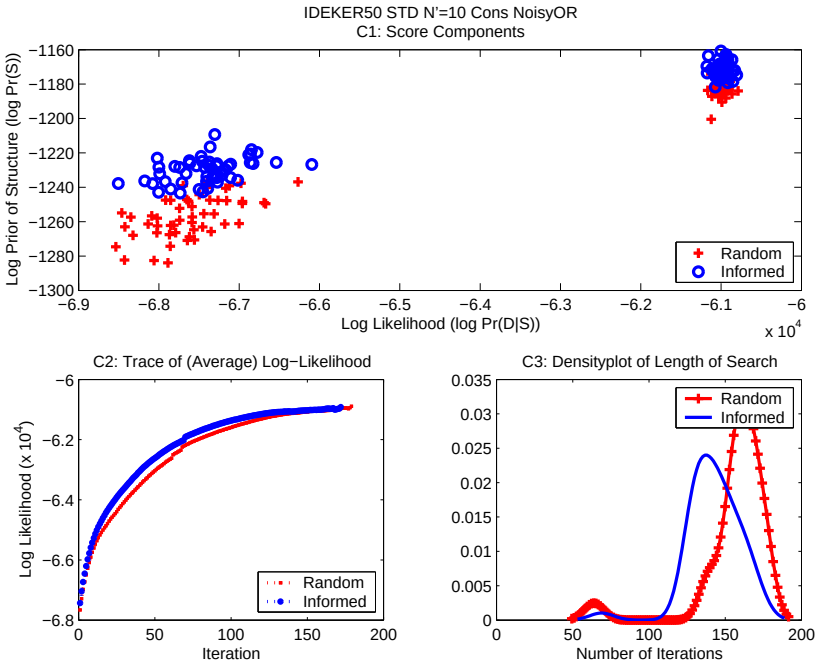


Figure F.34: IDEKER50 STD $N' = 10$ Cons NoisyOR

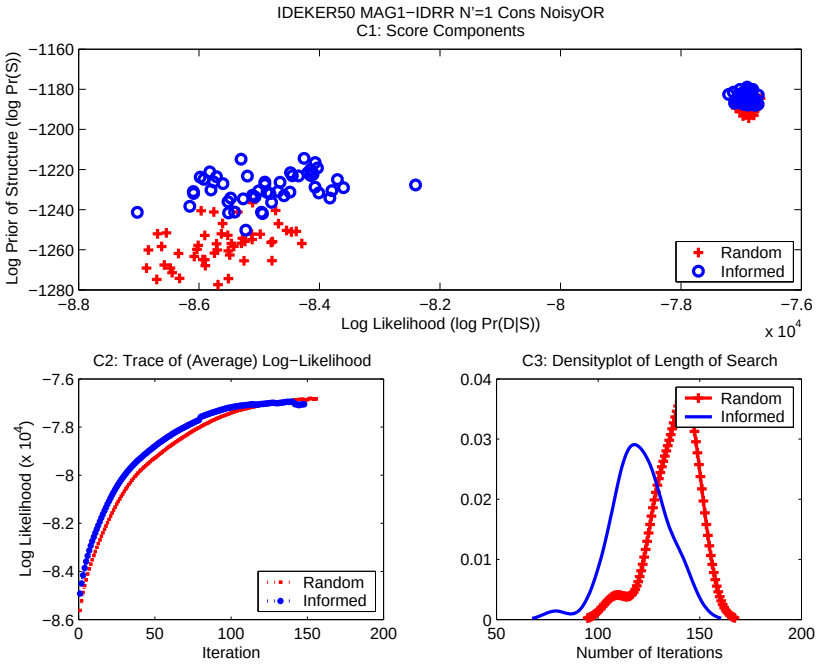


Figure F.35: IDEKER50 MAG1-IDRR $N' = 1$ Cons NoisyOR

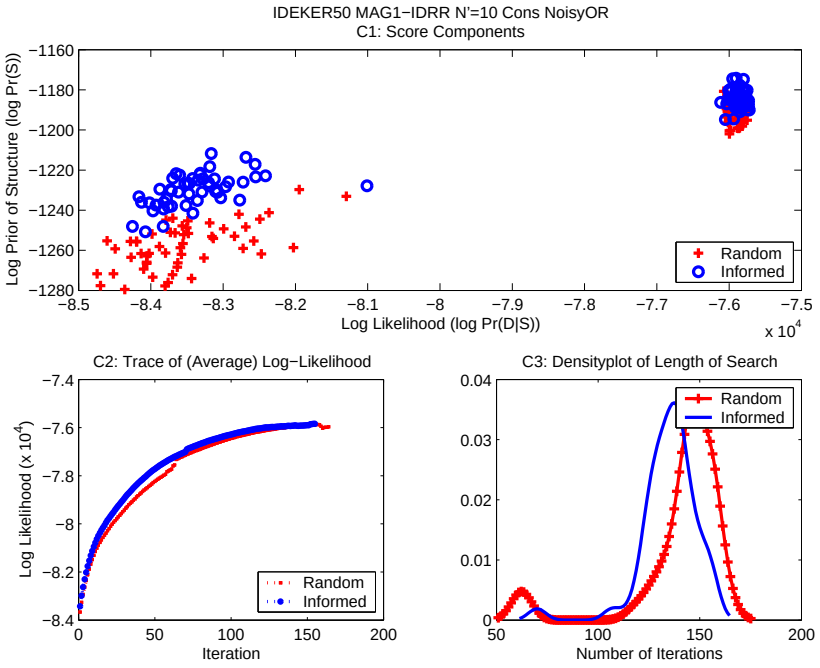


Figure F.36: IDEKER50 MAG1-IDRR $N' = 10$ Cons NoisyOR

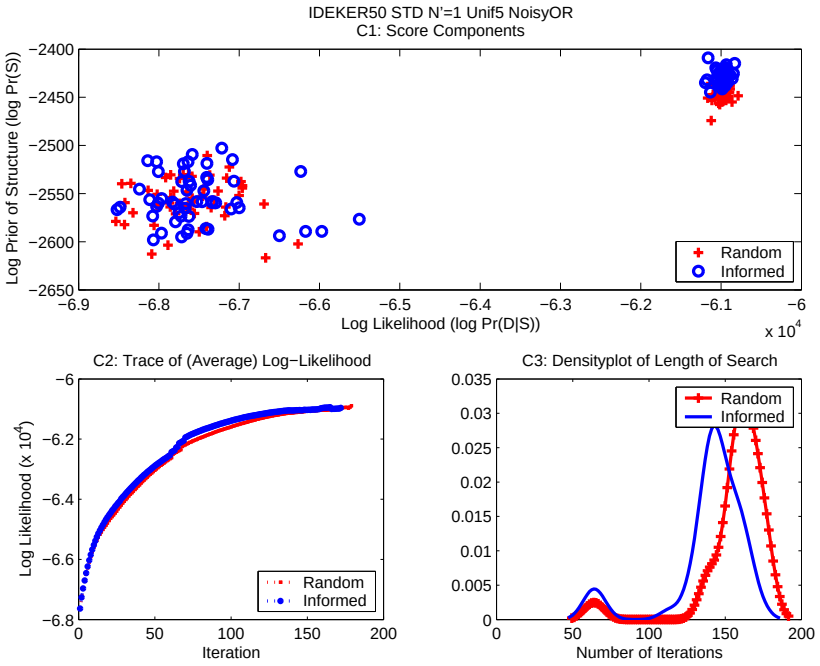


Figure F.37: IDEKER50 STD $N' = 1$ Unif5 NoisyOR

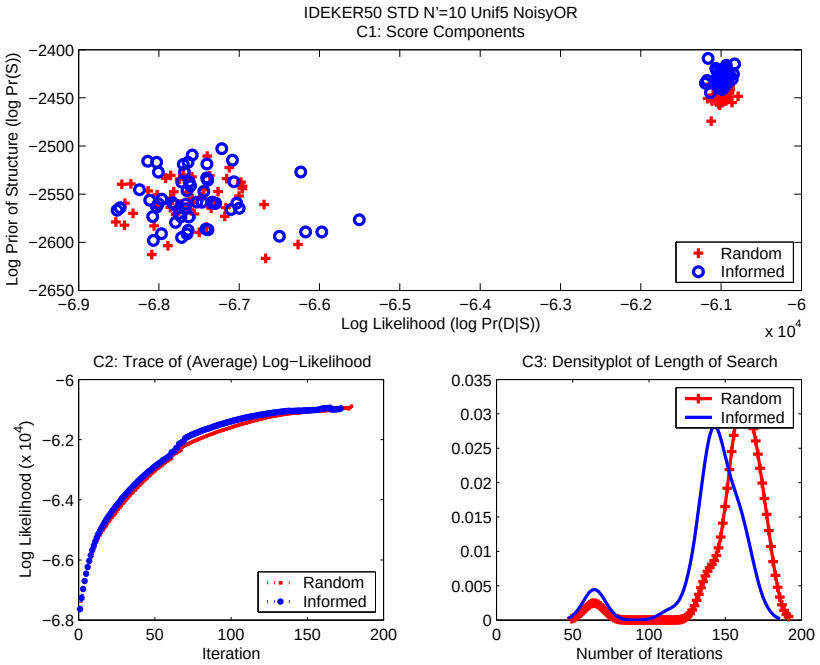


Figure F.38: IDEKER50 STD $N' = 10$ Unif5 NoisyOR

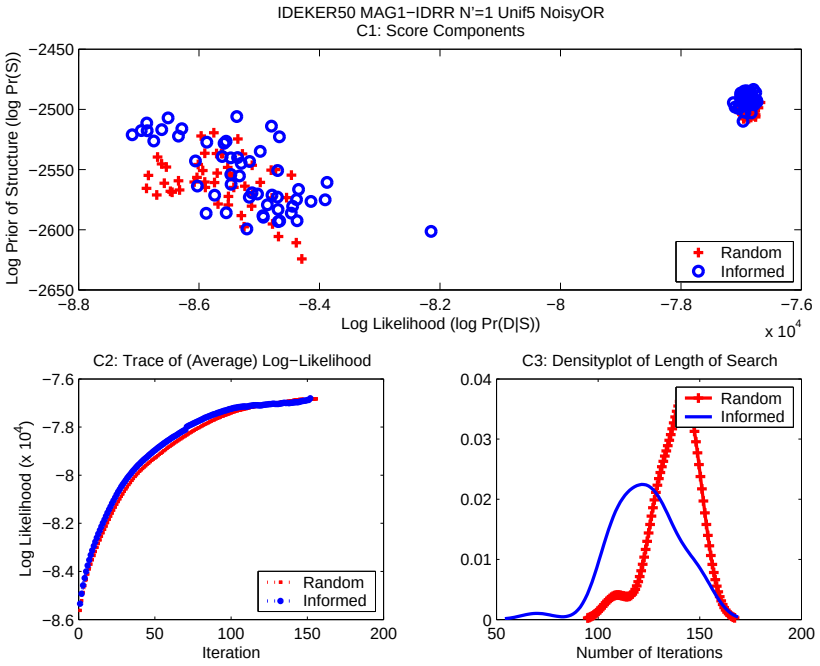


Figure F.39: IDEKER50 MAG1-IDRR $N' = 1$ Unif5 NoisyOR

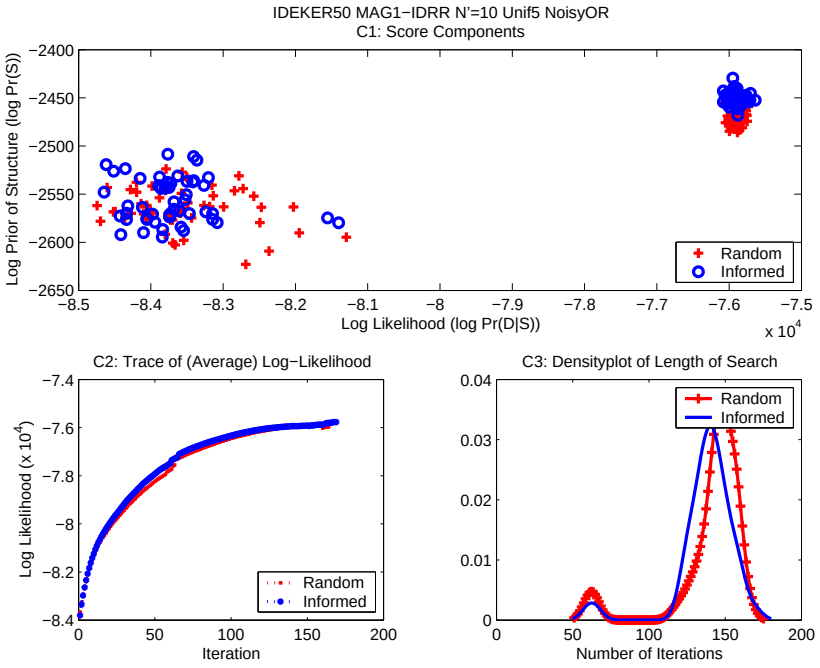


Figure F.40: IDEKER50 MAG1-IDRR $N' = 10$ Unif5 NoisyOR

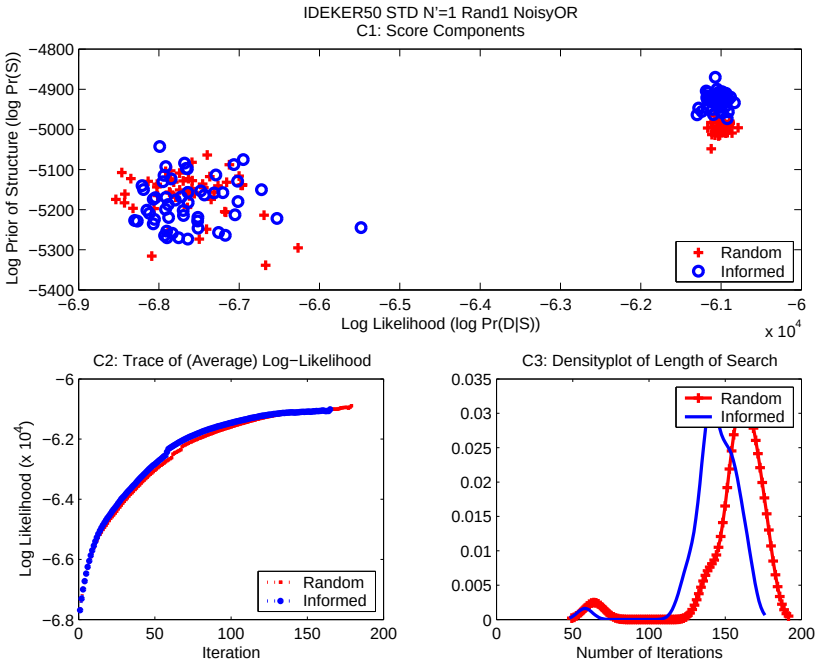


Figure F.41: IDEKER50 STD $N' = 1$ Rand1 NoisyOR

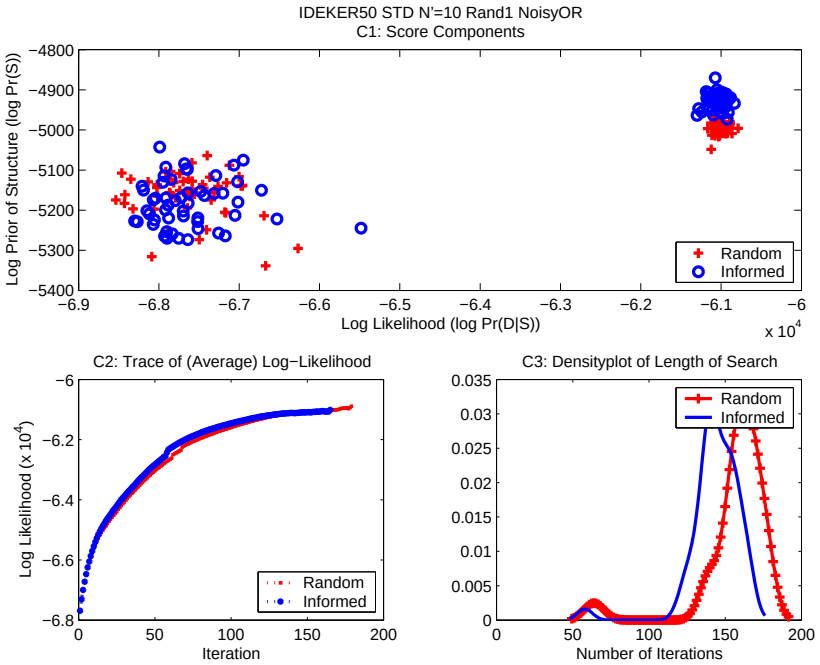


Figure F.42: IDEKER50 STD $N' = 10$ Rand1 NoisyOR

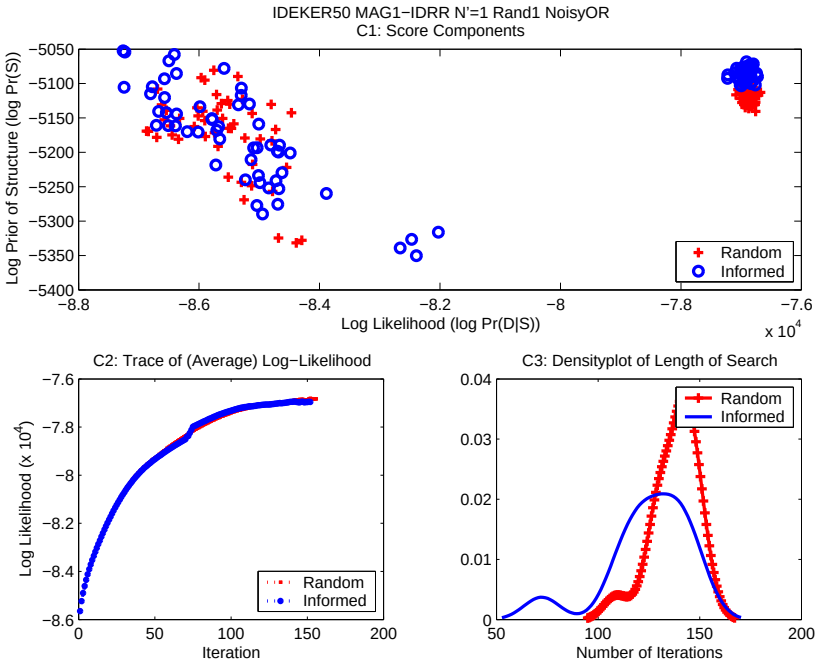


Figure F.43: IDEKER50 MAG1-IDRR $N' = 1$ Rand1 NoisyOR

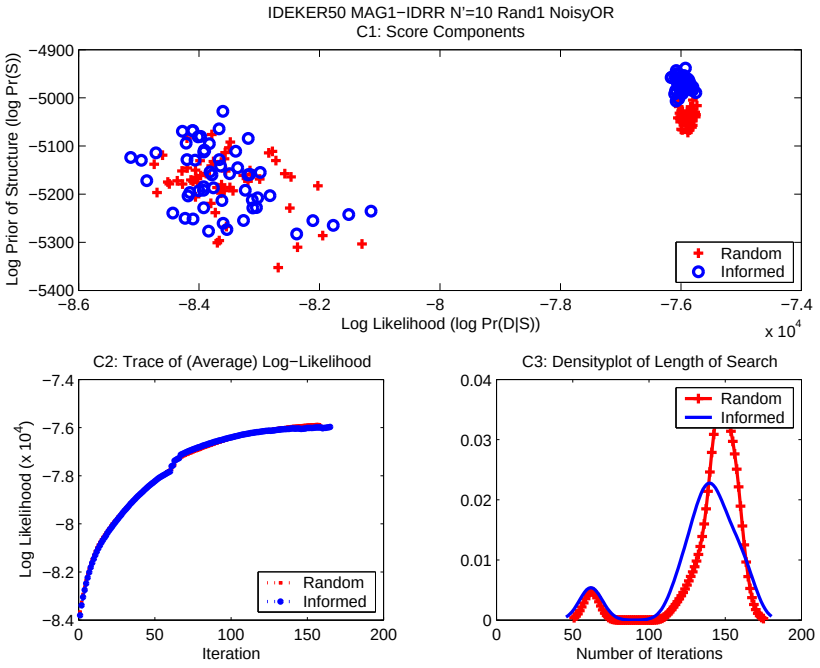


Figure F.44: IDEKER50 MAG1-IDRR $N' = 10$ Rand1 NoisyOR

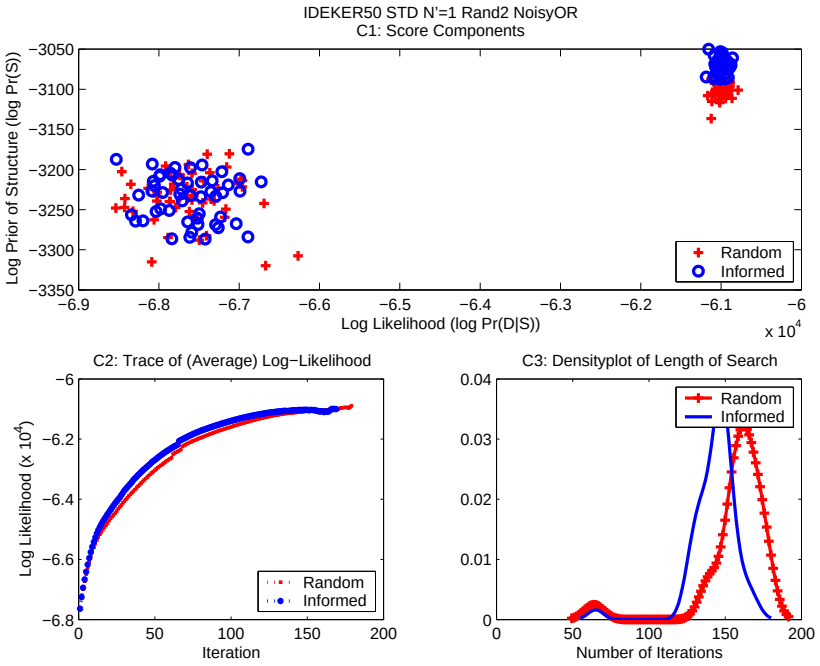


Figure F.45: IDEKER50 STD $N' = 1$ Rand2 NoisyOR

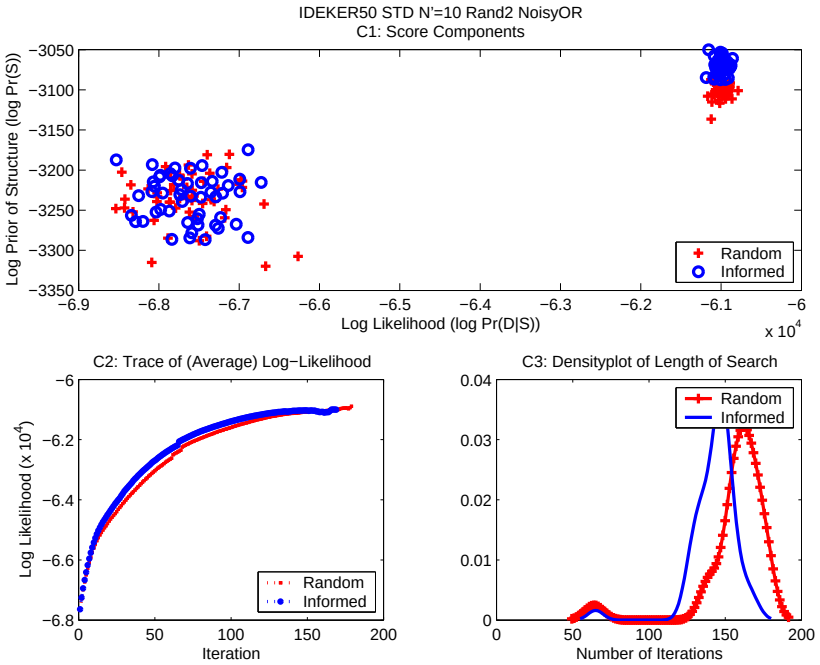


Figure F.46: IDEKER50 STD $N' = 10$ Rand2 NoisyOR

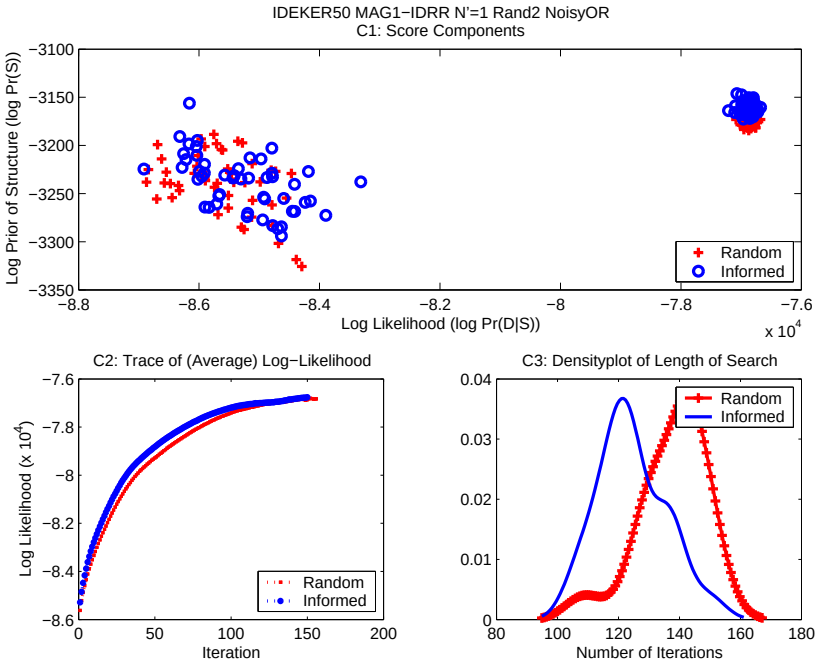


Figure F.47: IDEKER50 MAG1-IDRR $N' = 1$ Rand2 NoisyOR

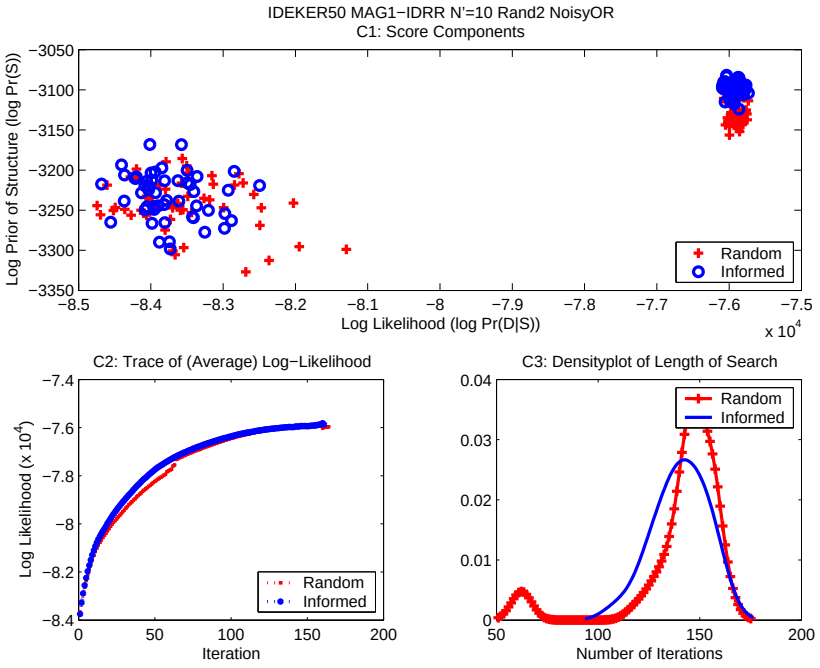


Figure F.48: IDEKER50 MAG1-IDRR $N' = 10$ Rand2 NoisyOR

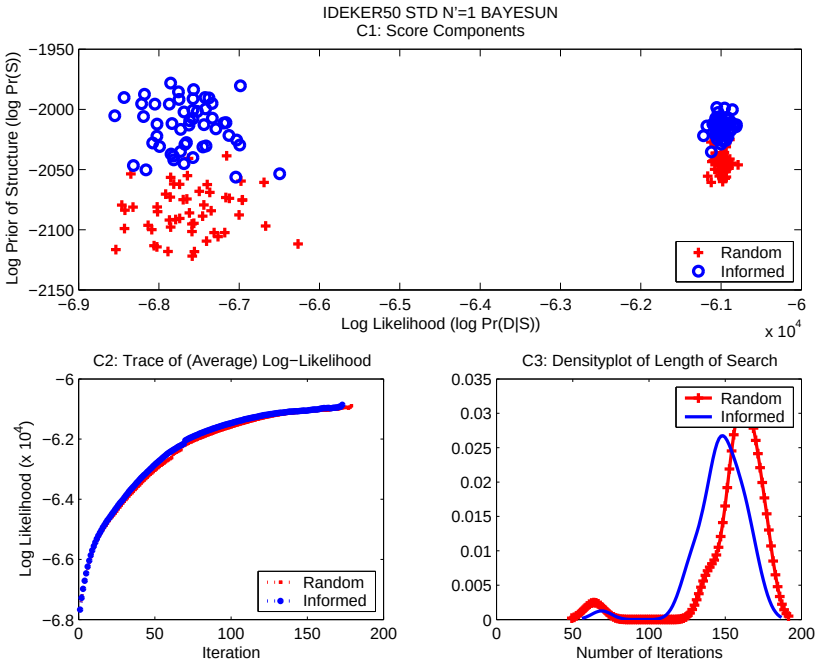


Figure F.49: IDEKER50 STD $N' = 1$ BAYESUN

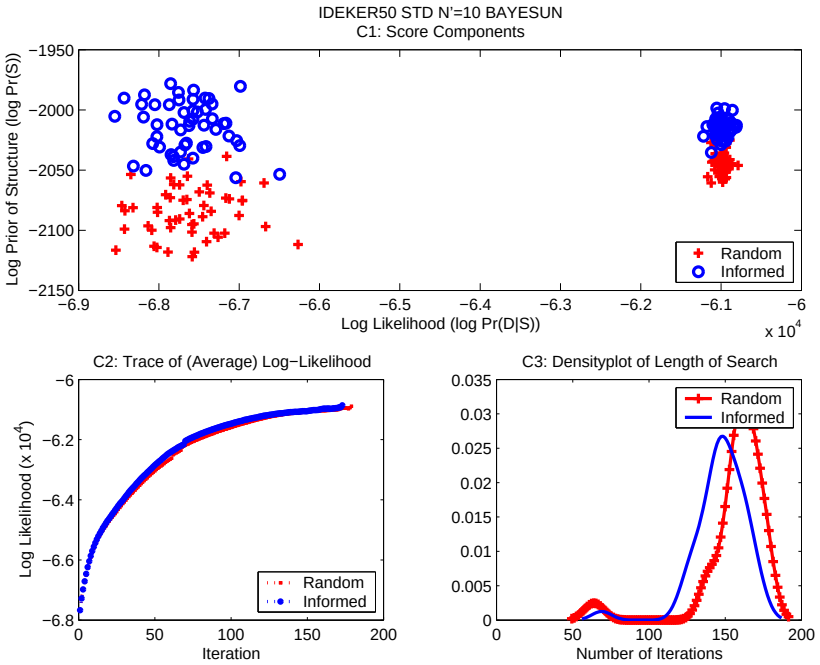


Figure F.50: IDEKER50 STD $N' = 10$ BAYESUN

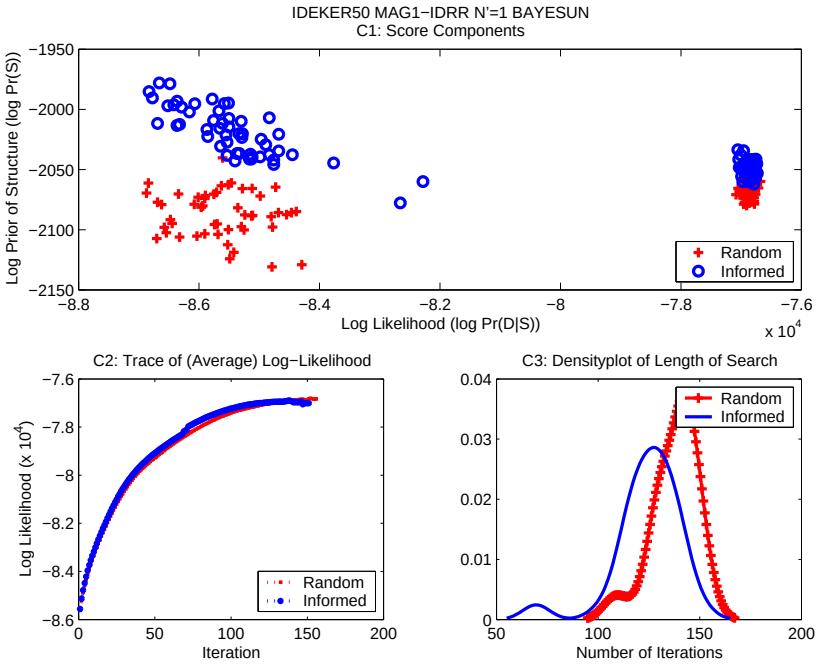


Figure F.51: IDEKER50 MAG1-IDRR $N' = 1$ BAYESUN

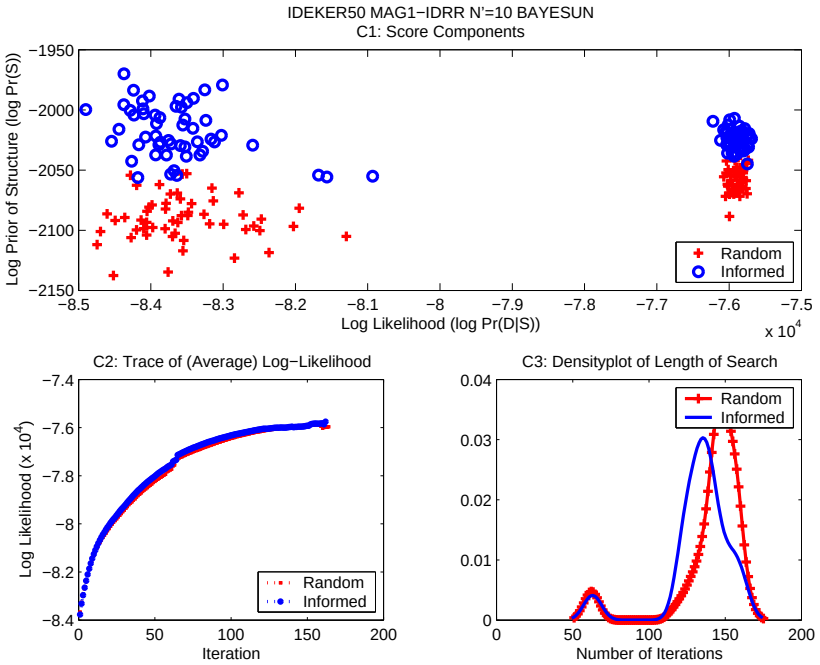


Figure F.52: IDEKER50 MAG1-IDRR $N' = 10$ BAYESUN

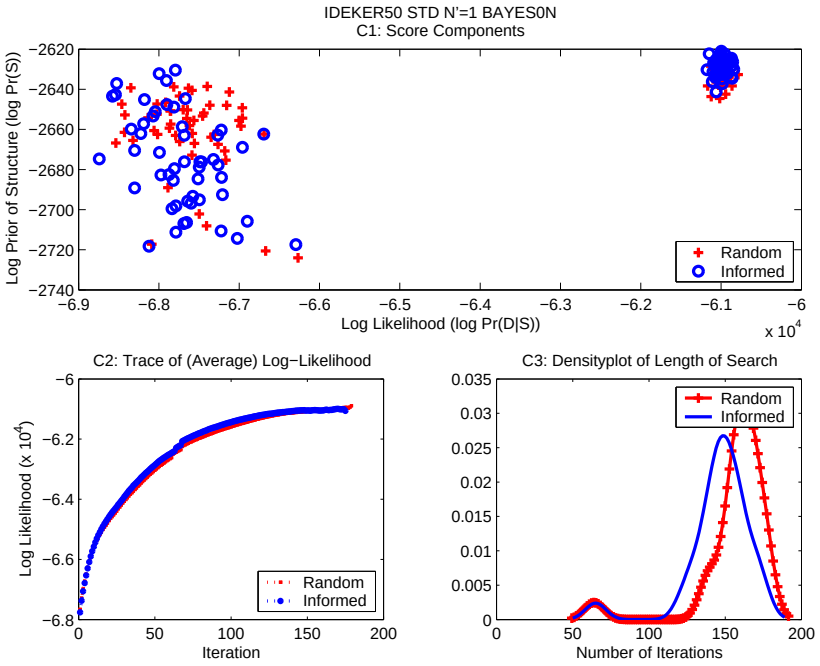


Figure F.53: IDEKER50 STD $N' = 1$ BAYES0N

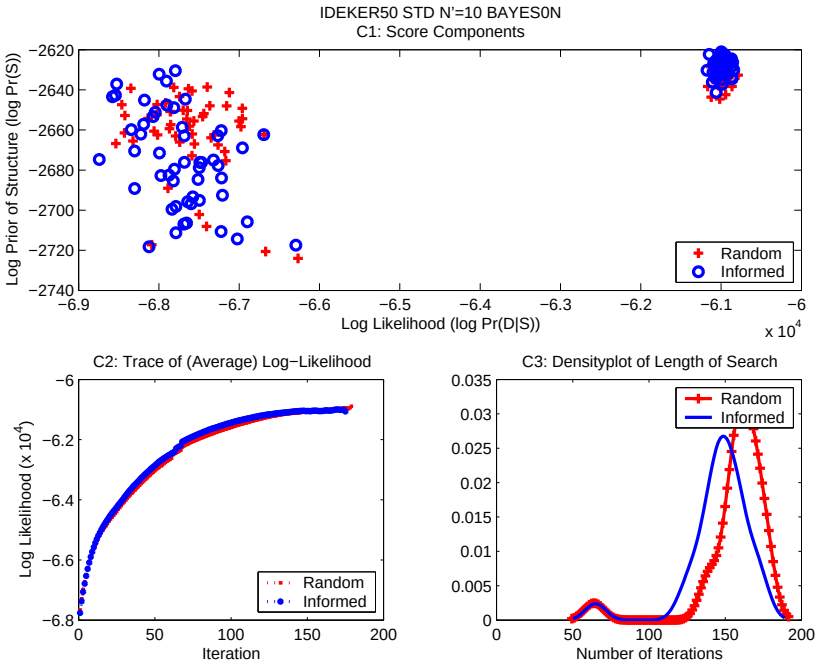


Figure F.54: IDEKER50 STD $N' = 10$ BAYES0N

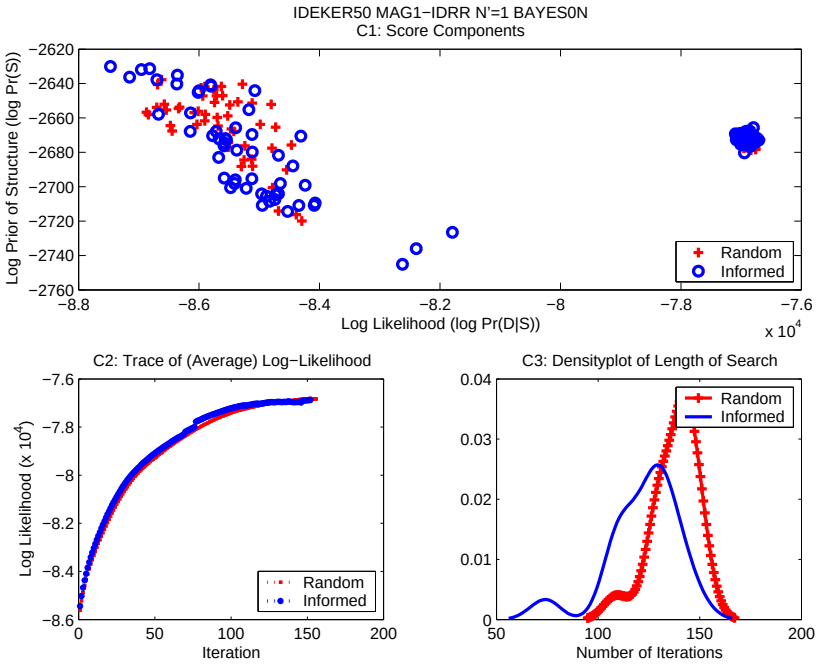


Figure F.55: IDEKER50 MAG1-IDRR $N' = 1$ BAYES0N

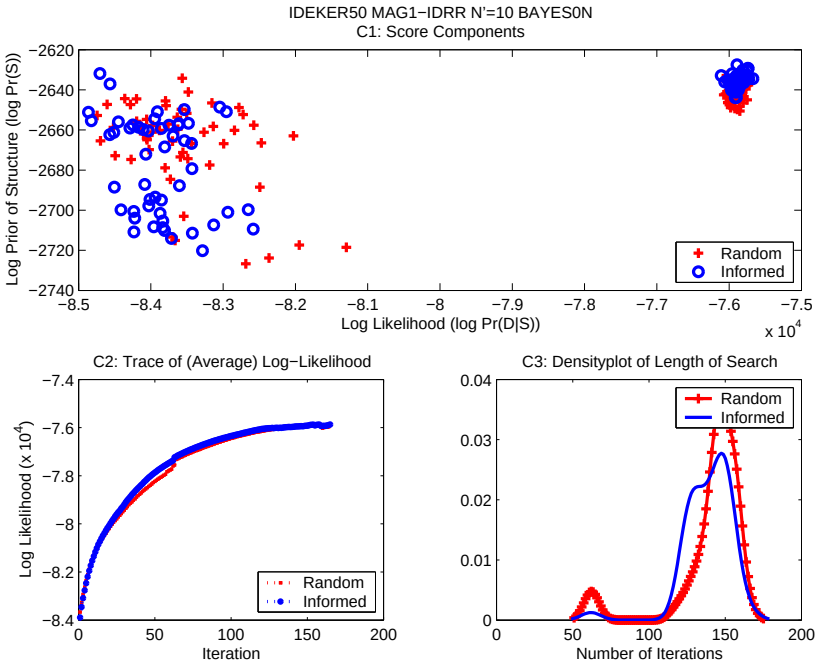


Figure F.56: IDEKER50 MAG1-IDRR $N' = 10$ BAYES0N

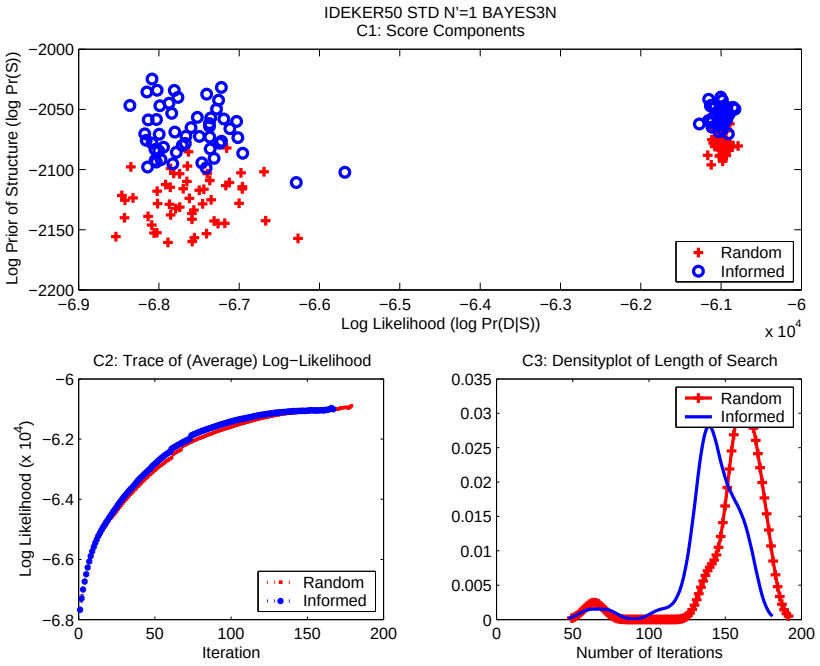


Figure F.57: IDEKER50 STD $N' = 1$ BAYES3N

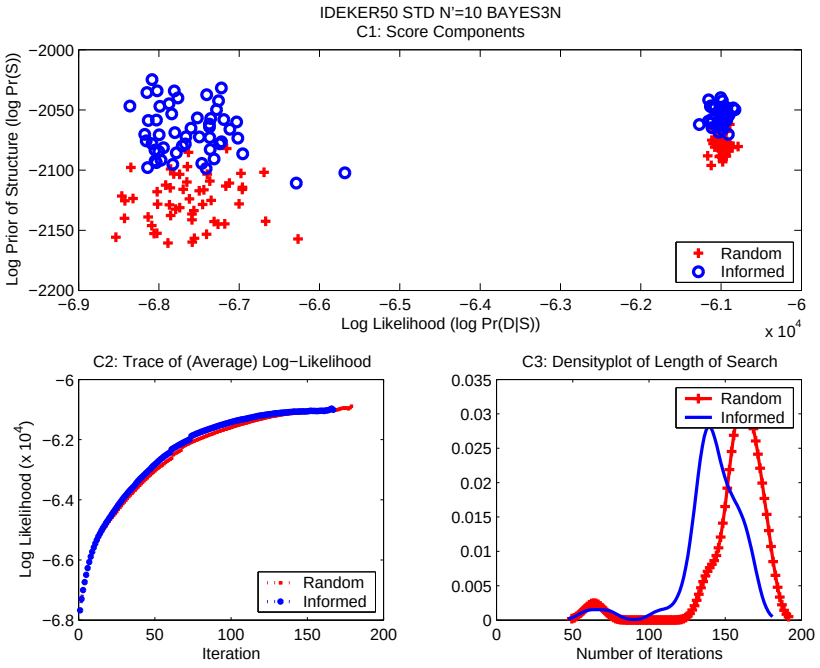


Figure F.58: IDEKER50 STD $N' = 10$ BAYES3N

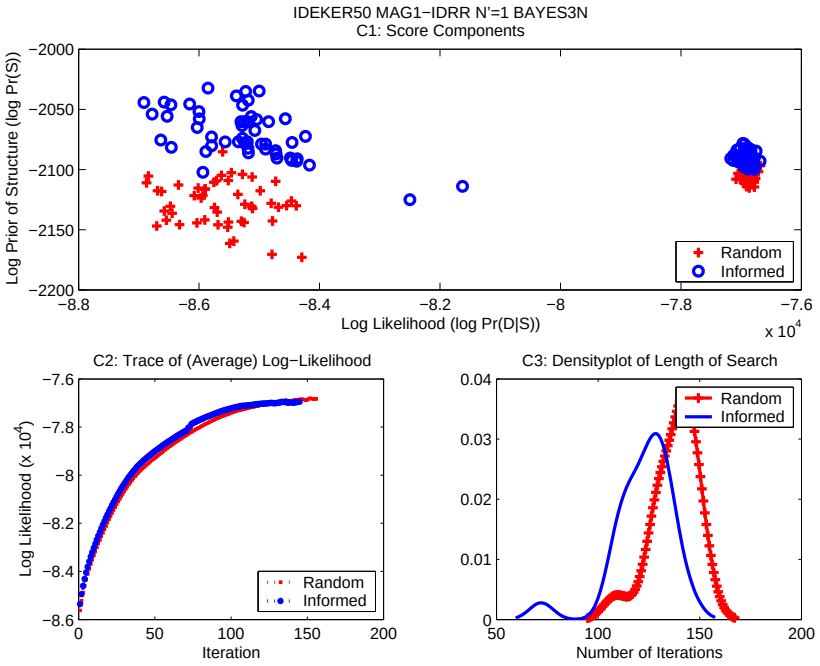


Figure F.59: IDEKER50 MAG1-IDRR $N' = 1$ BAYES3N

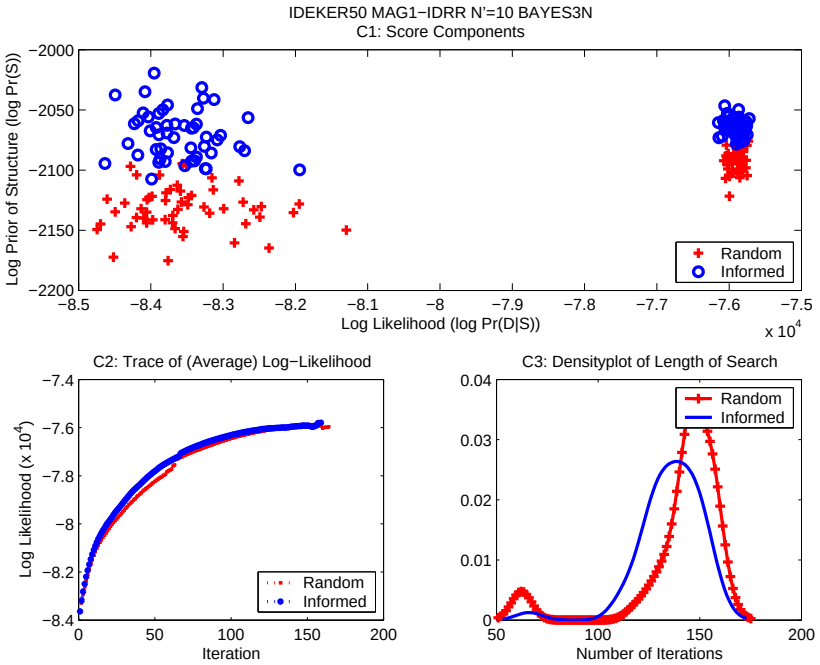


Figure F.60: IDEKER50 MAG1-IDRR $N' = 10$ BAYES3N

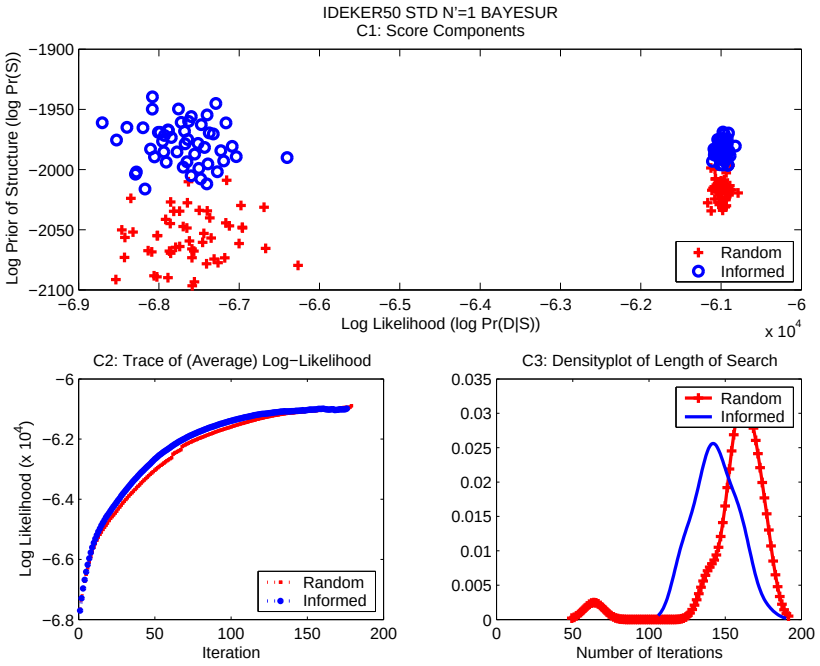


Figure F.61: IDEKER50 STD $N' = 1$ BAYESUR

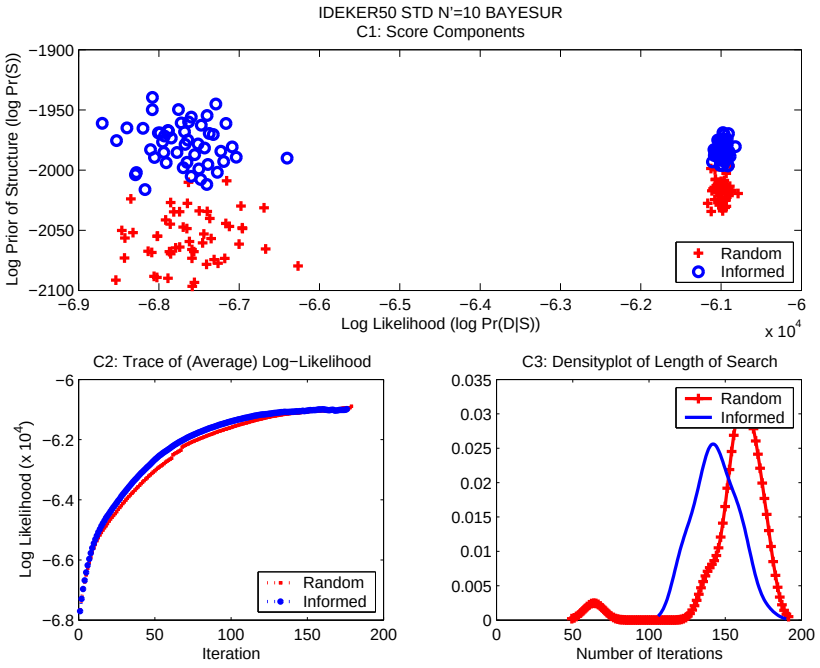


Figure F.62: IDEKER50 STD $N' = 10$ BAYESUR

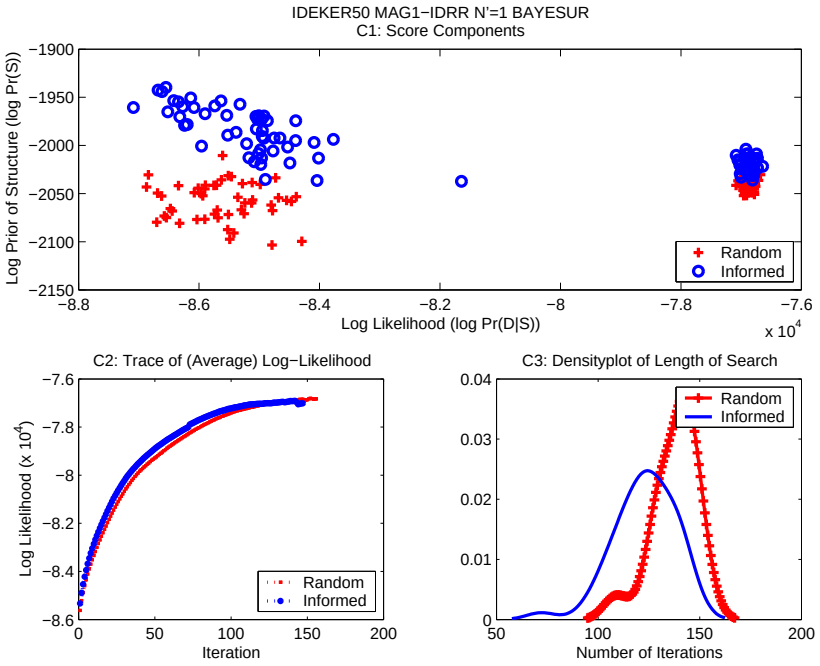


Figure F.63: IDEKER50 MAG1-IDRR $N' = 1$ BAYESUR

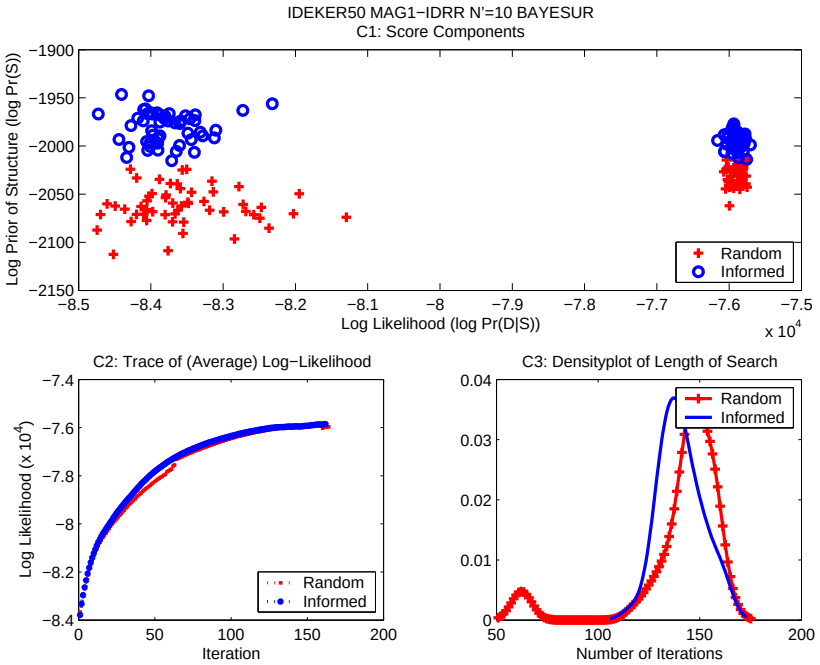


Figure F.64: IDEKER50 MAG1-IDRR $N' = 10$ BAYESUR

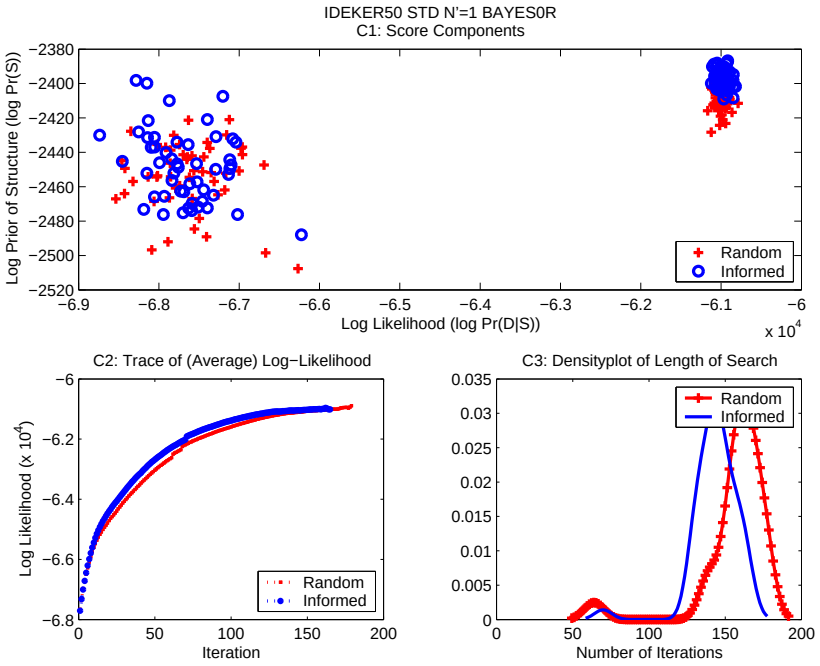


Figure F.65: IDEKER50 STD $N' = 1$ BAYES0R

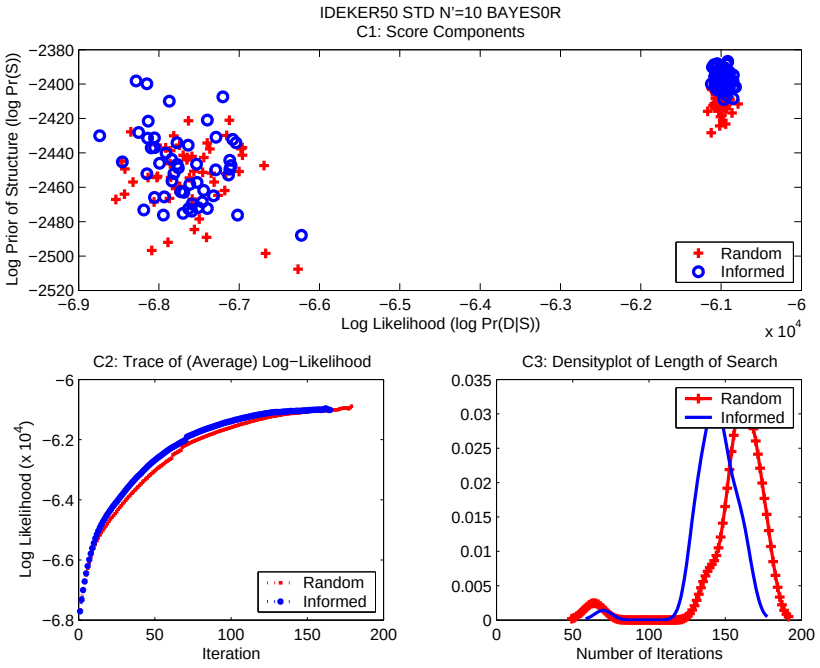


Figure F.66: IDEKER50 STD $N' = 10$ BAYES0R

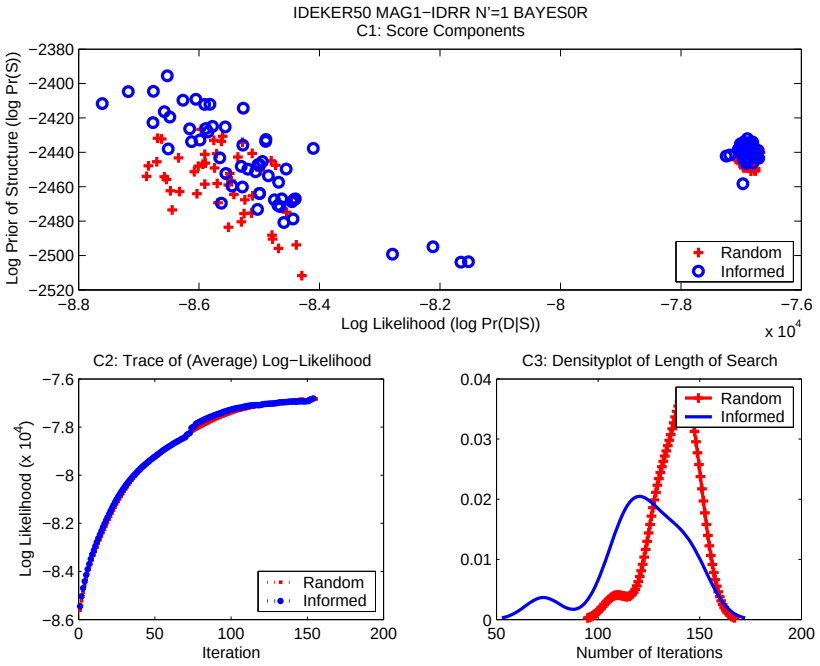


Figure F.67: IDEKER50 MAG1-IDRR $N' = 1$ BAYES0R

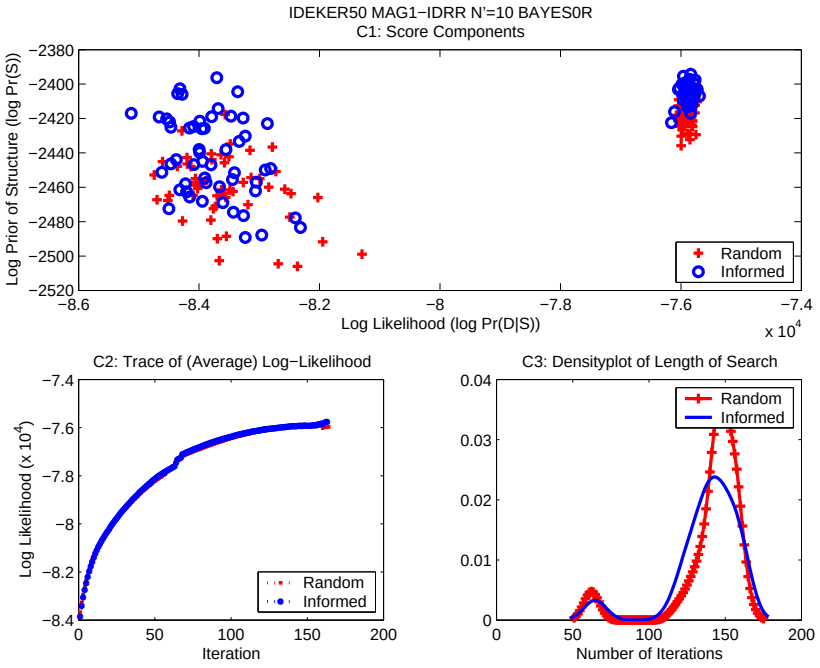


Figure F.68: IDEKER50 MAG1-IDRR $N' = 10$ BAYES0R

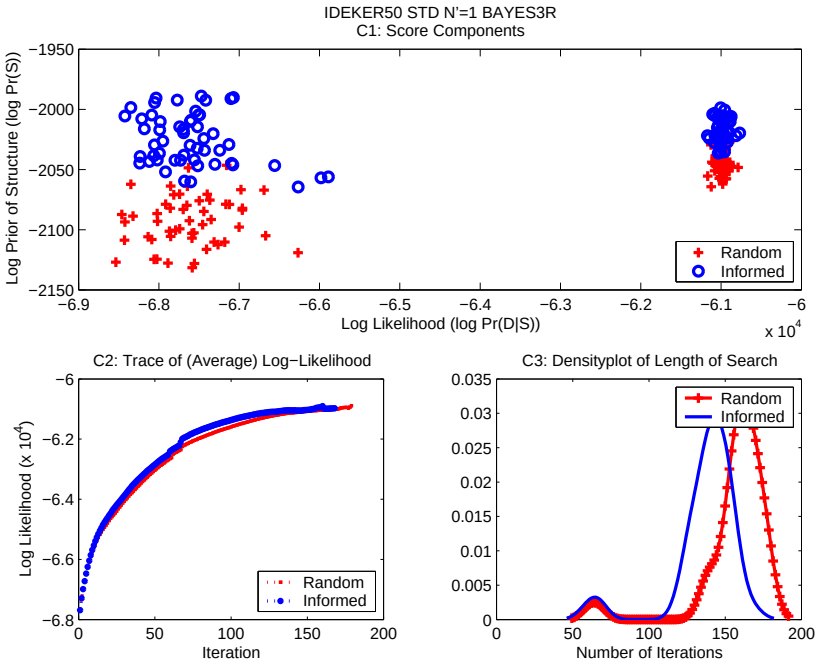


Figure F.69: IDEKER50 STD $N' = 1$ BAYES3R

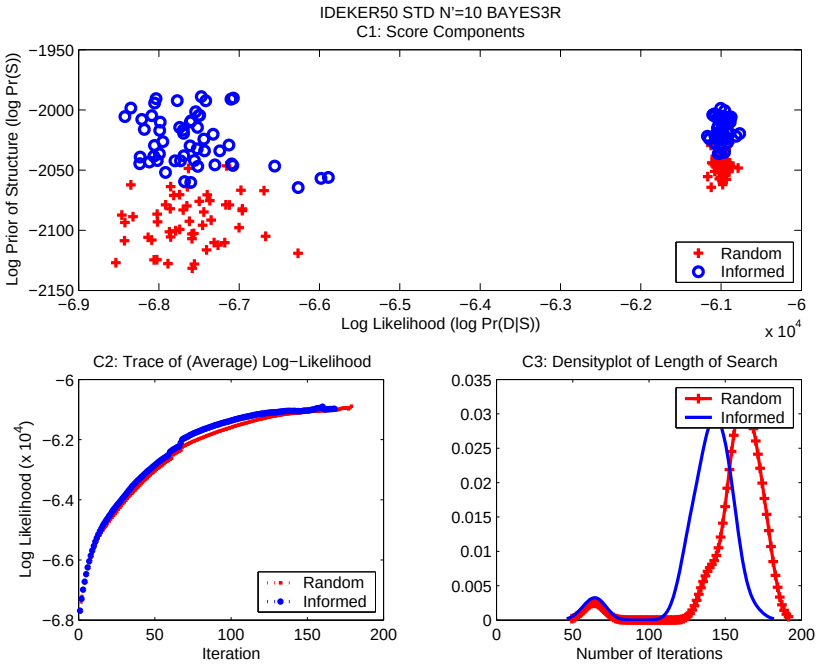


Figure F.70: IDEKER50 STD $N' = 10$ BAYES3R

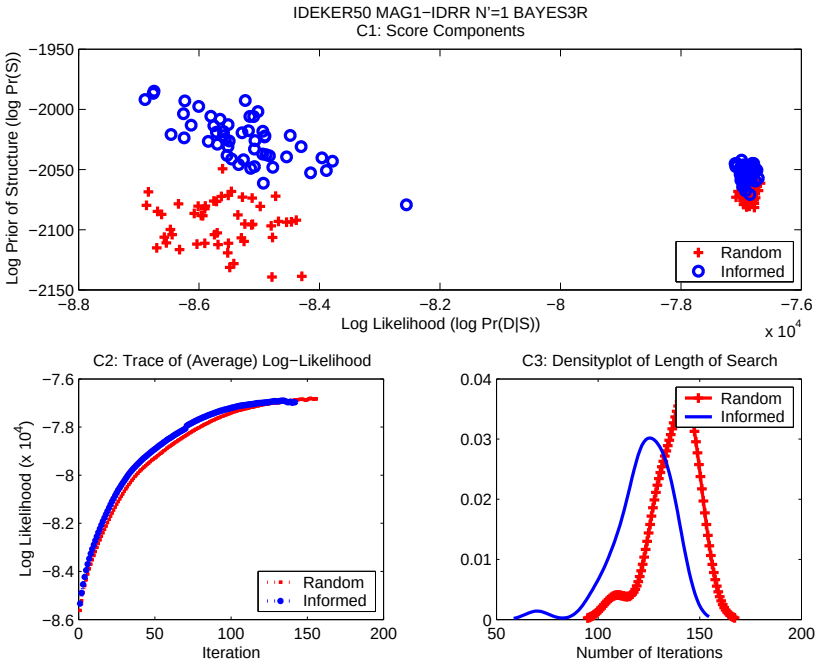


Figure F.71: IDEKER50 MAG1-IDRR $N' = 1$ BAYES3R

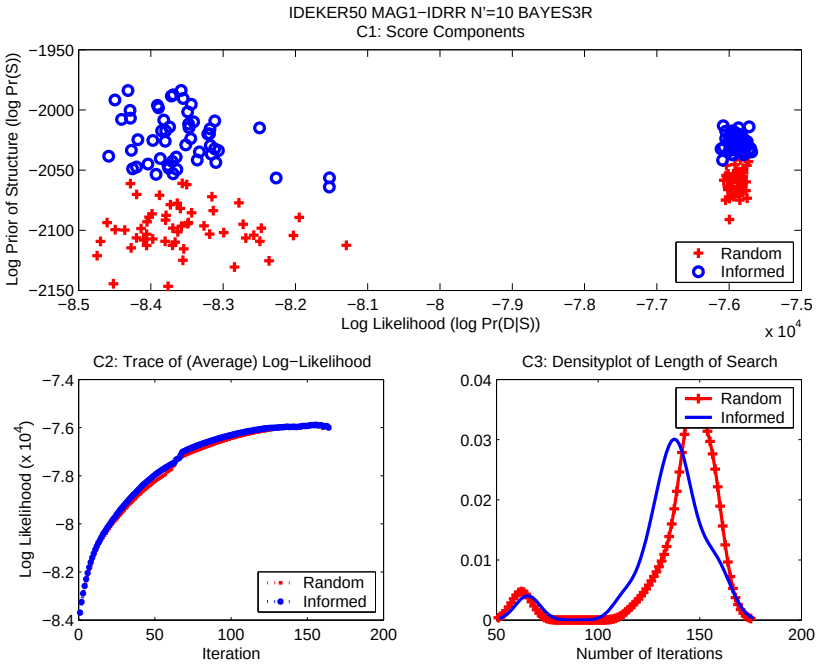


Figure F.72: IDEKER50 MAG1-IDRR $N' = 10$ BAYES3R

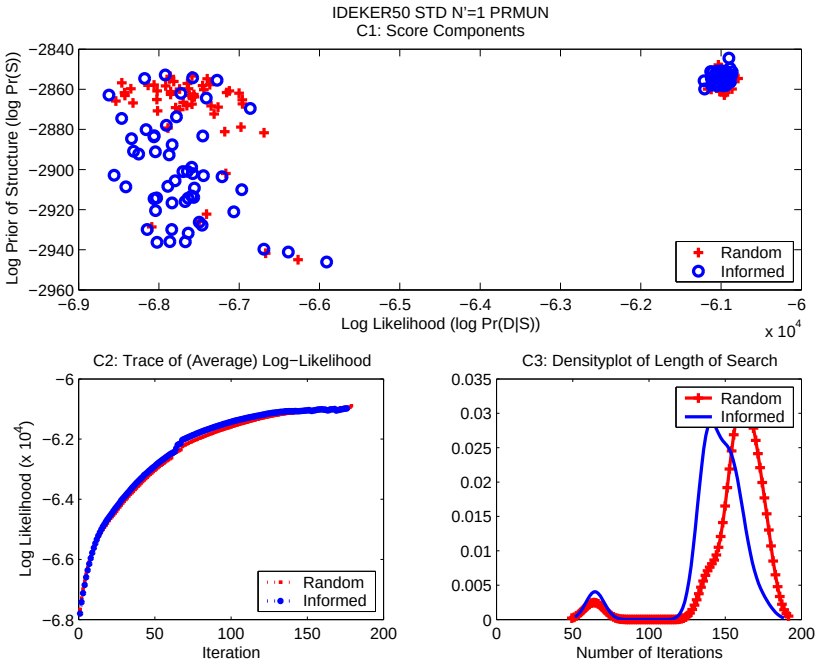


Figure F.73: IDEKER50 STD $N' = 1$ PRMUN

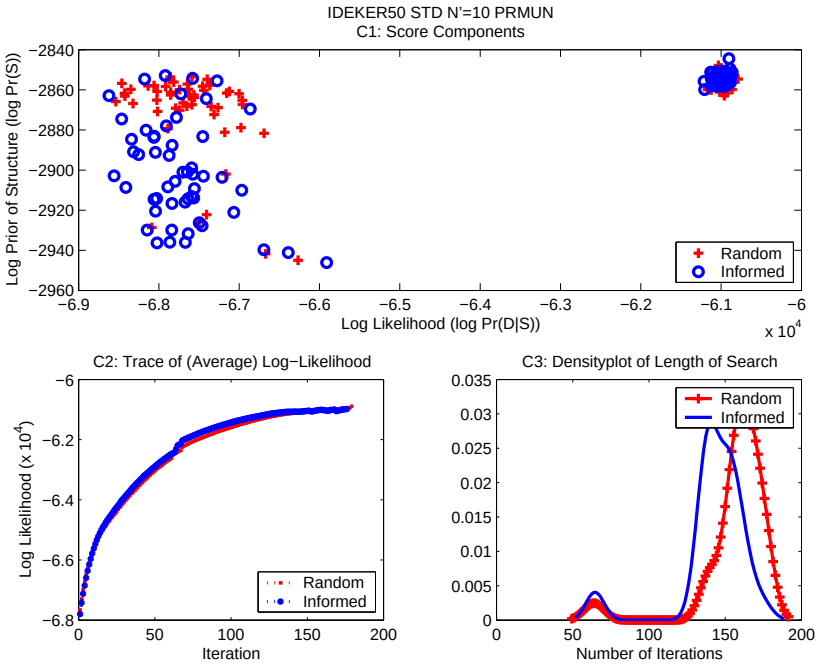


Figure F.74: IDEKER50 STD $N' = 10$ PRMUN

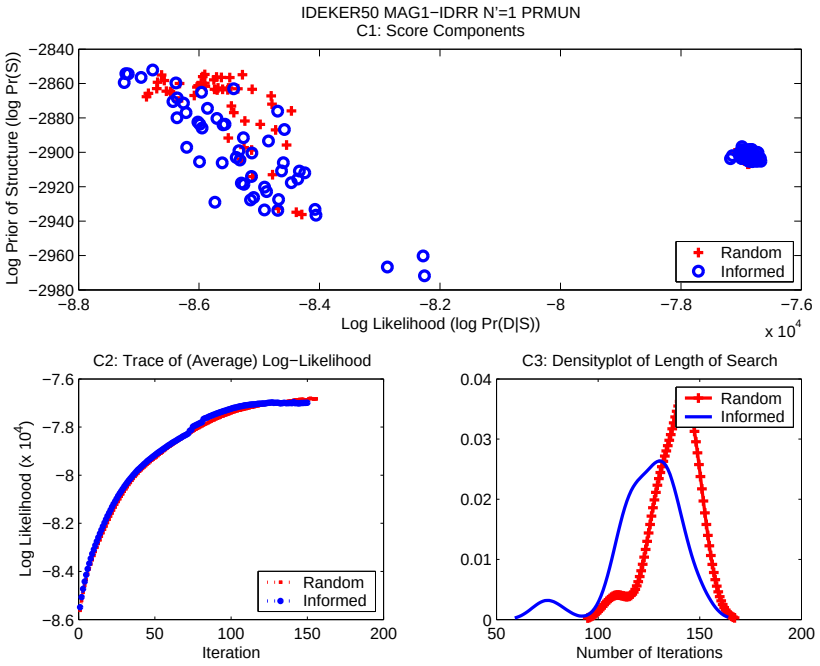


Figure F.75: IDEKER50 MAG1-IDRR $N' = 1$ PRMUN

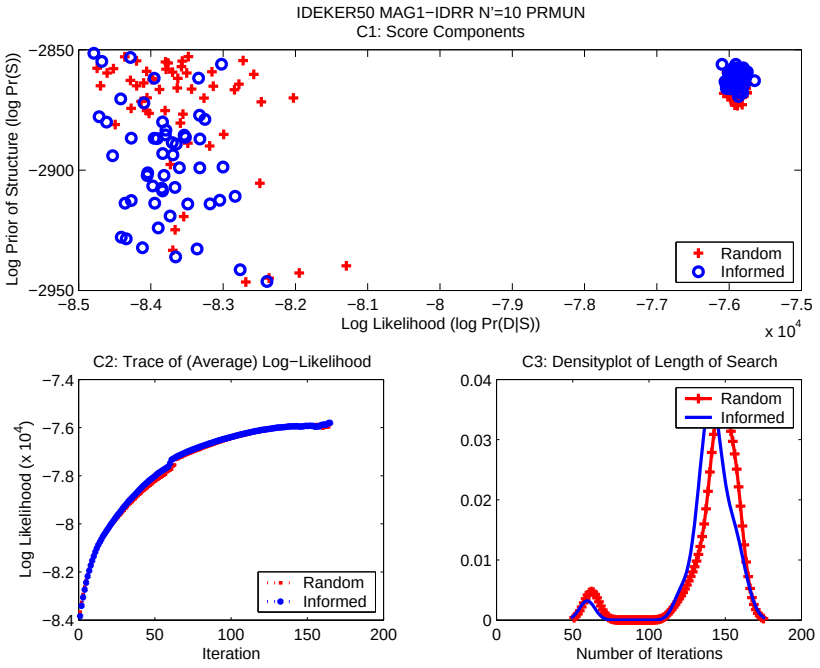


Figure F.76: IDEKER50 MAG1-IDRR $N' = 10$ PRMUN

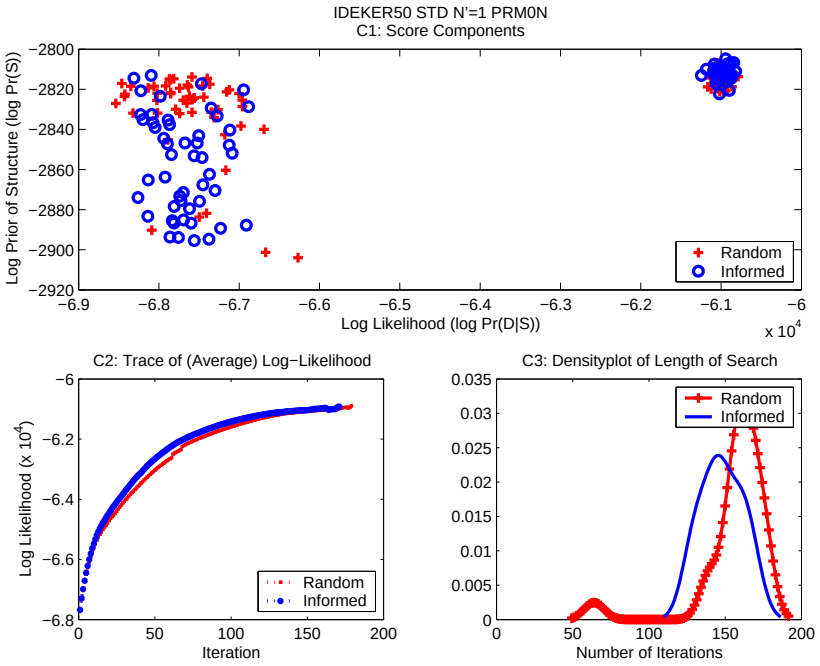


Figure F.77: IDEKER50 STD $N' = 1$ PRM0N

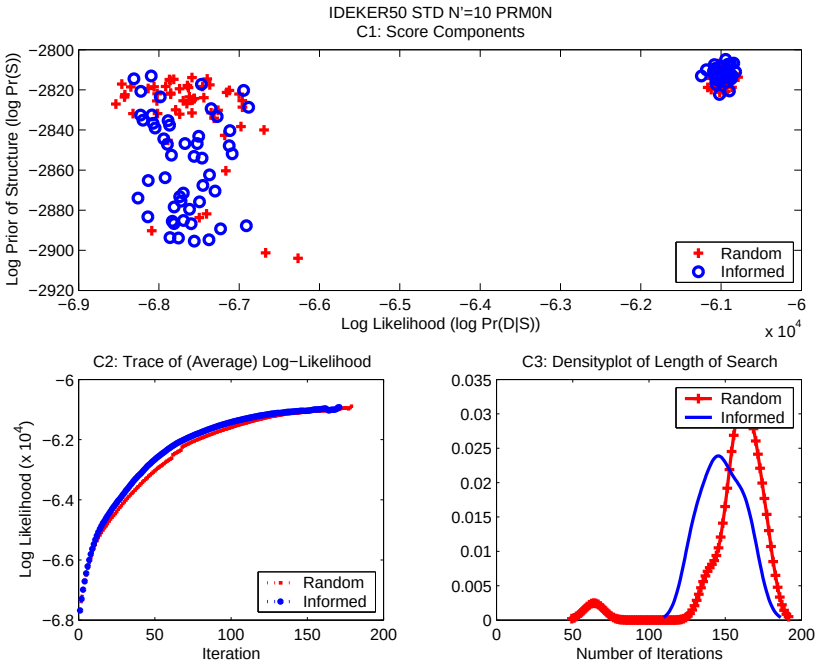


Figure F.78: IDEKER50 STD $N' = 10$ PRM0N

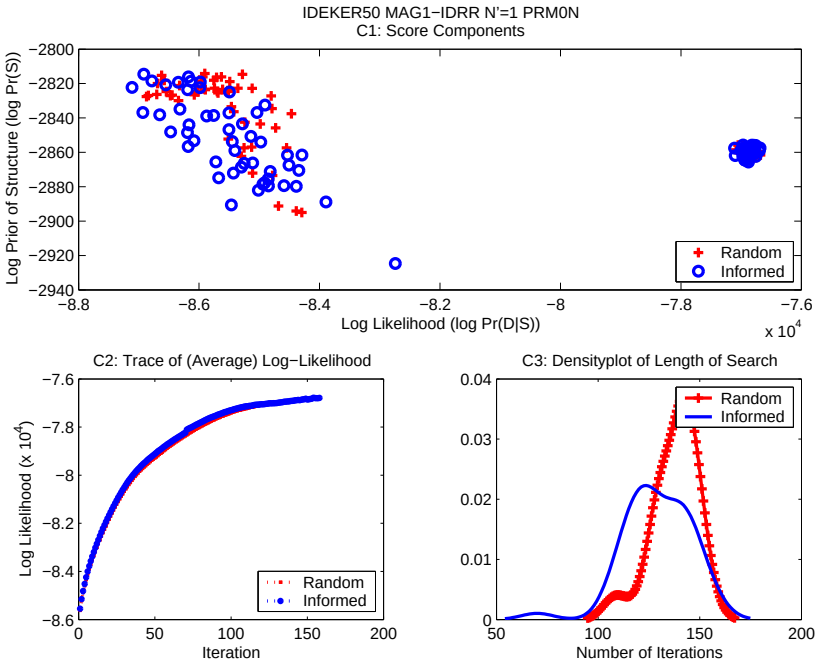


Figure F.79: IDEKER50 MAG1-IDRR $N' = 1$ PRM0N

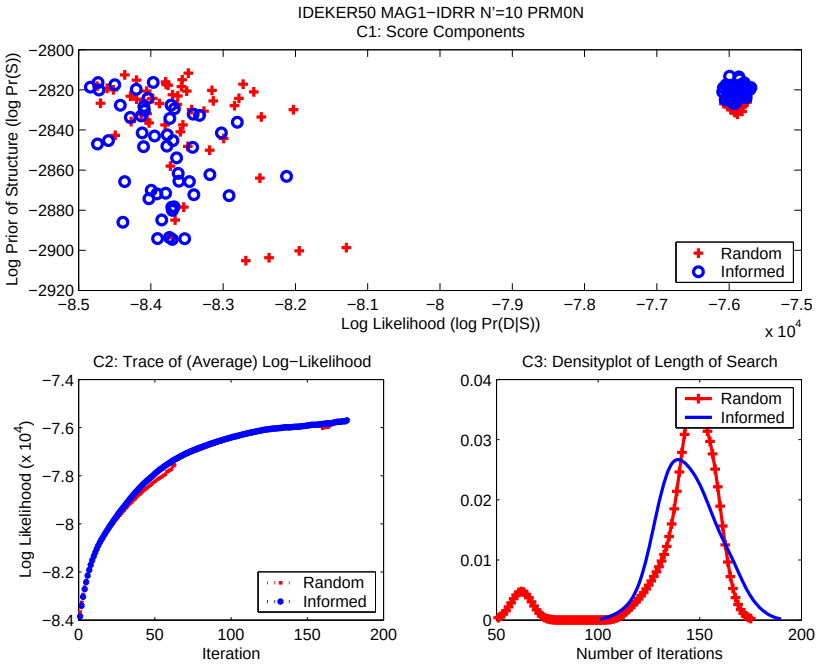


Figure F.80: IDEKER50 MAG1-IDRR $N' = 10$ PRM0N

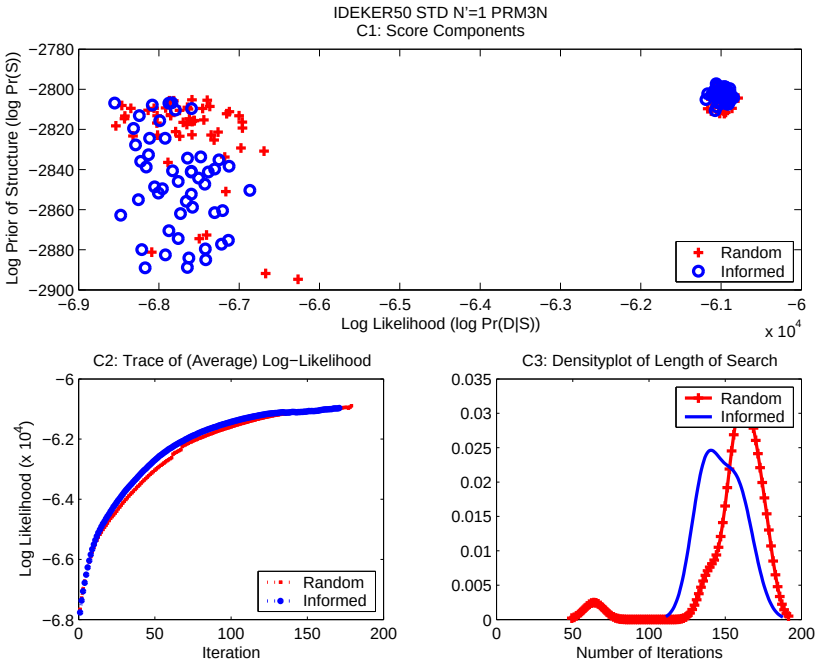


Figure F.81: IDEKER50 STD $N' = 1$ PRM3N

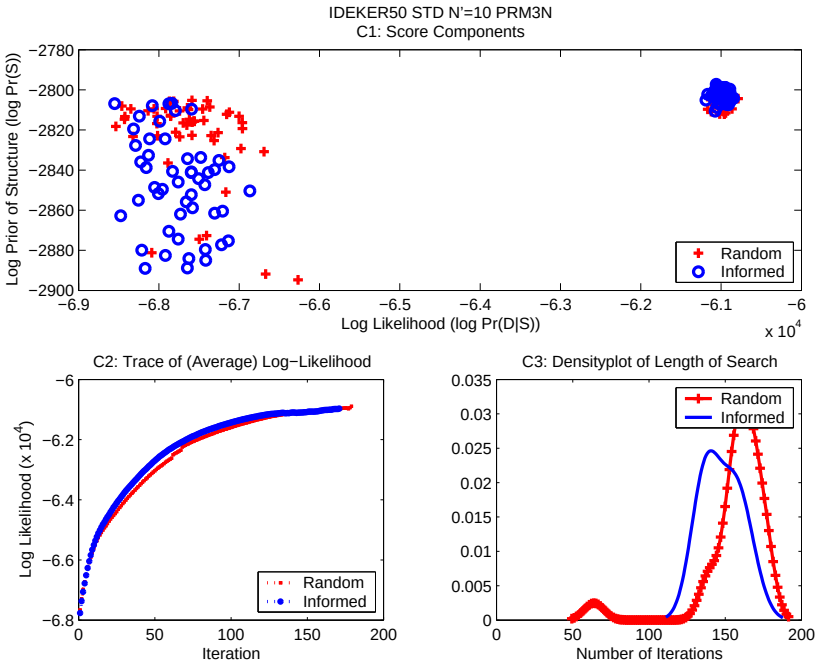


Figure F.82: IDEKER50 STD $N' = 10$ PRM3N

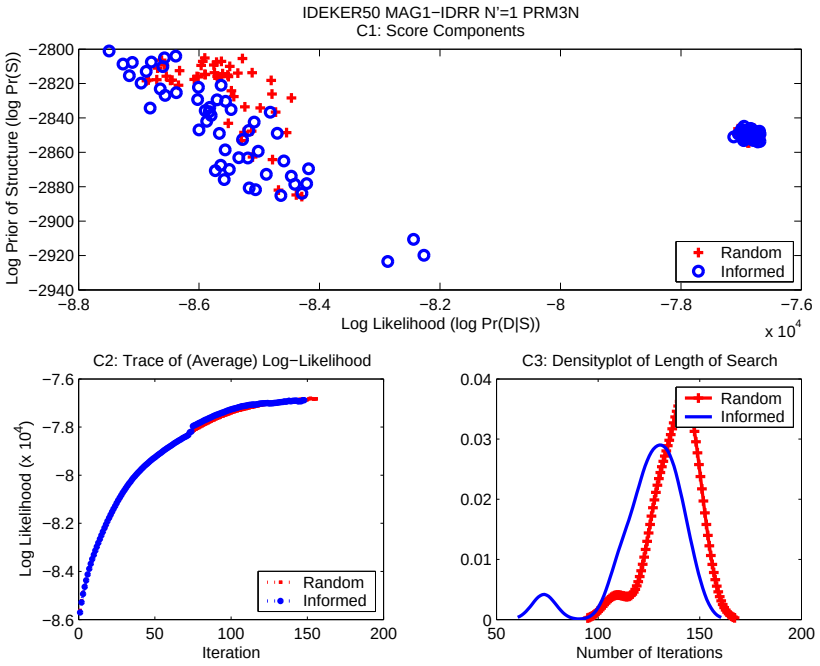


Figure F.83: IDEKER50 MAG1-IDRR $N' = 1$ PRM3N

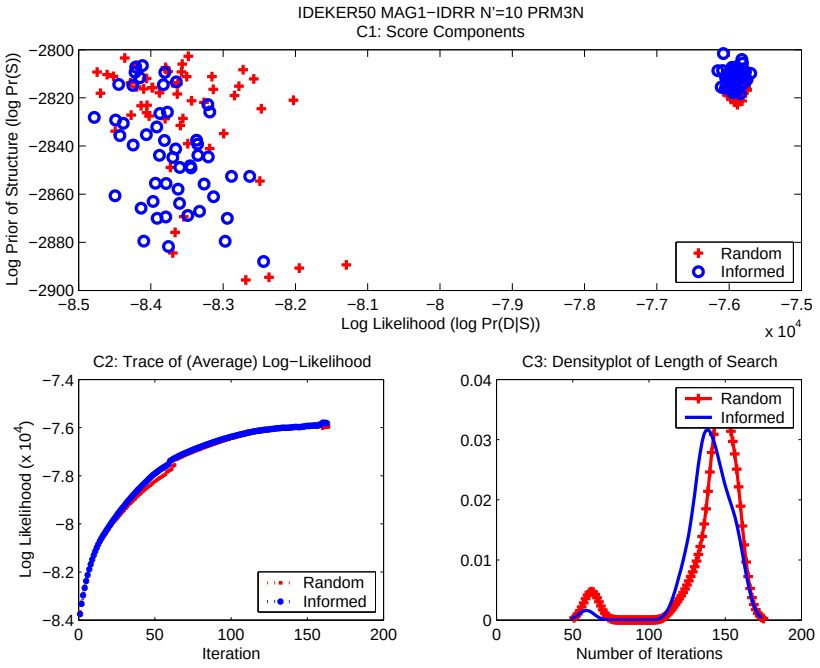


Figure F.84: IDEKER50 MAG1-IDRR $N' = 10$ PRM3N

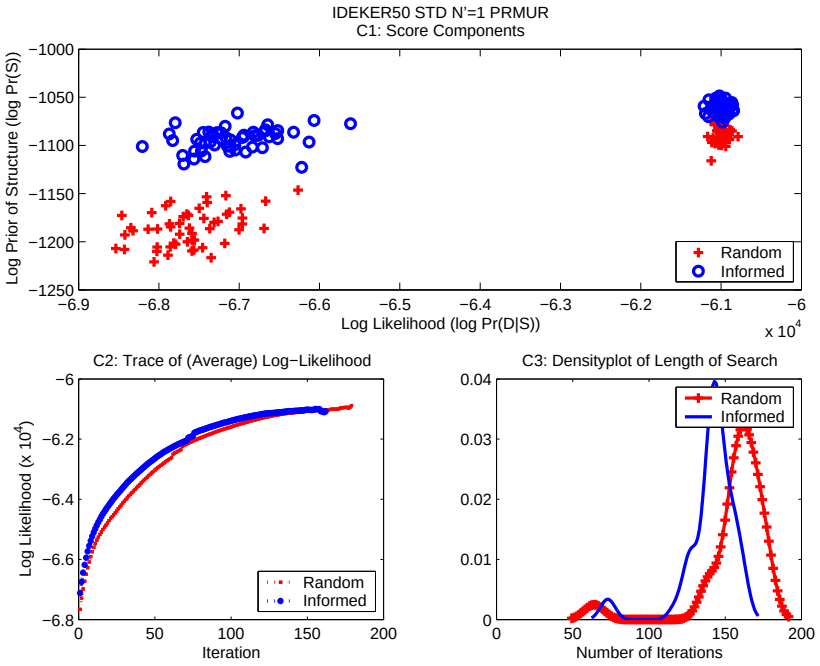


Figure F.85: IDEKER50 STD $N' = 1$ PRMUR

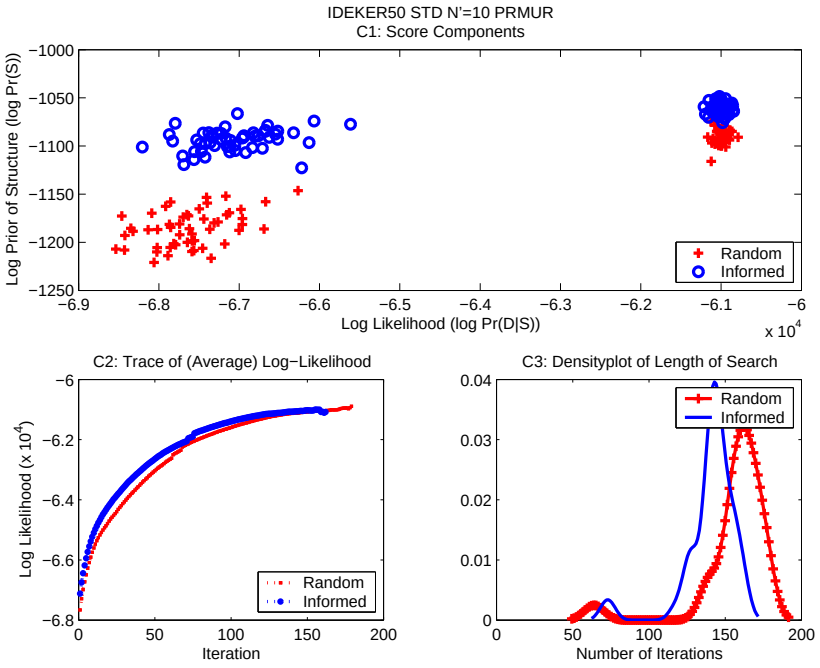


Figure F.86: IDEKER50 STD $N' = 10$ PRMUR

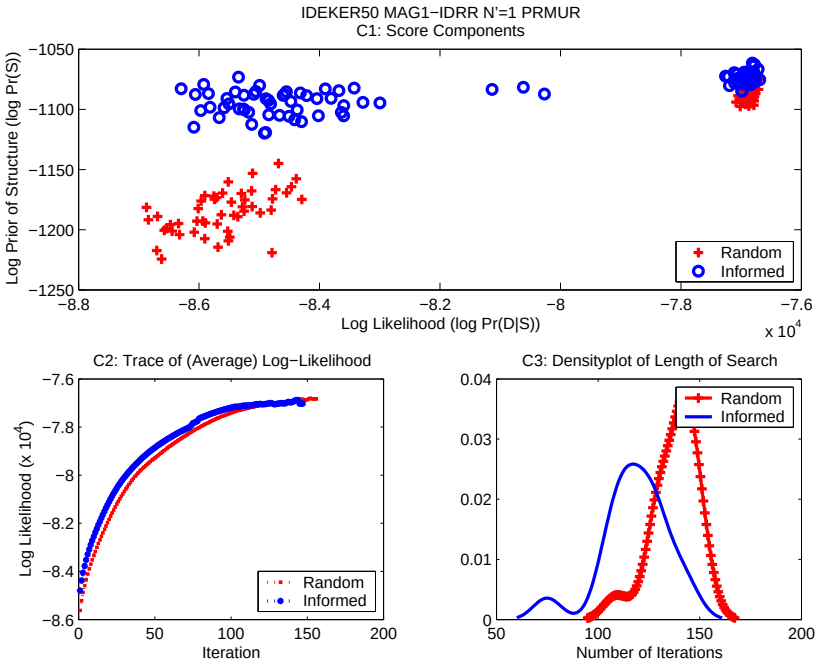


Figure F.87: IDEKER50 MAG1-IDRR $N' = 1$ PRMUR

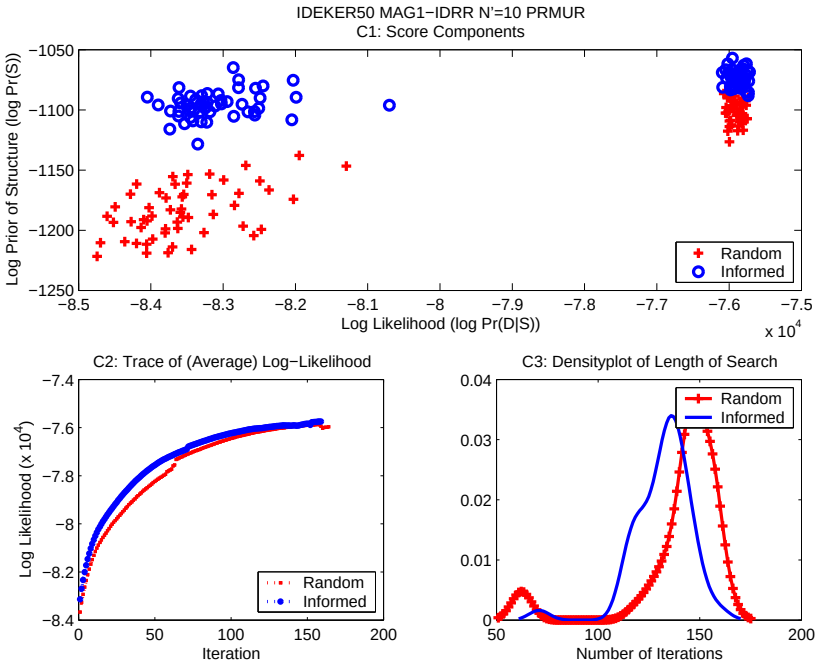


Figure F.88: IDEKER50 MAG1-IDRR $N' = 10$ PRMUR

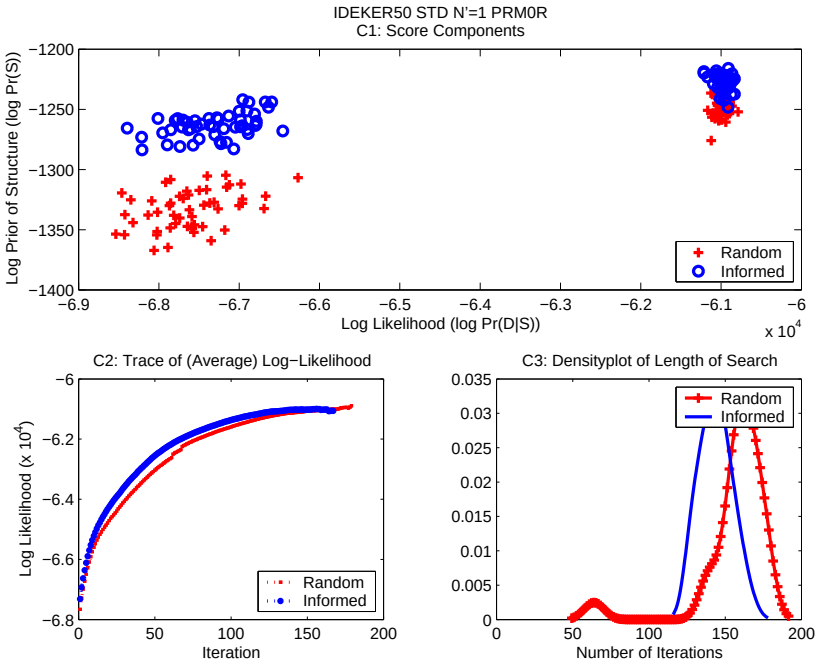


Figure F.89: IDEKER50 STD $N' = 1$ PRM0R

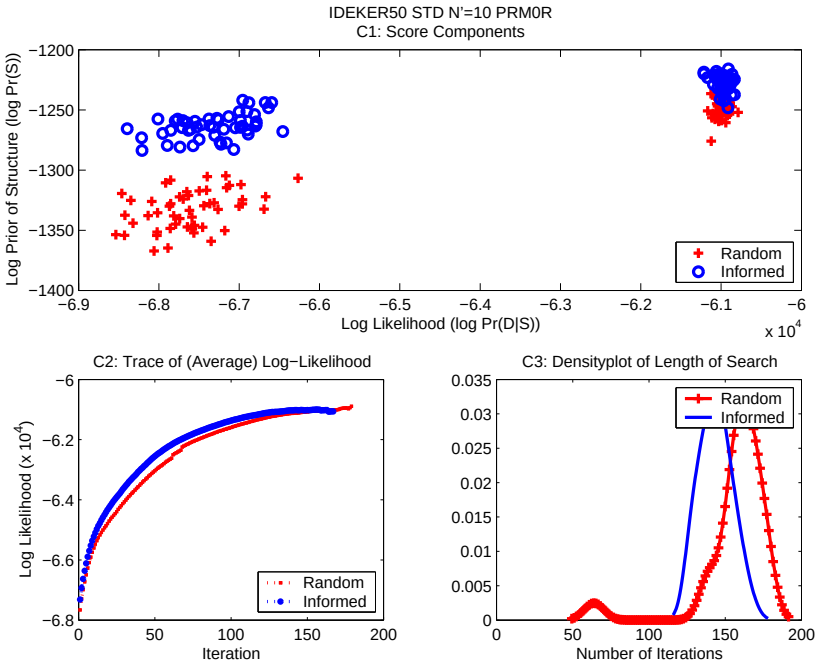


Figure F.90: IDEKER50 STD $N' = 10$ PRM0R

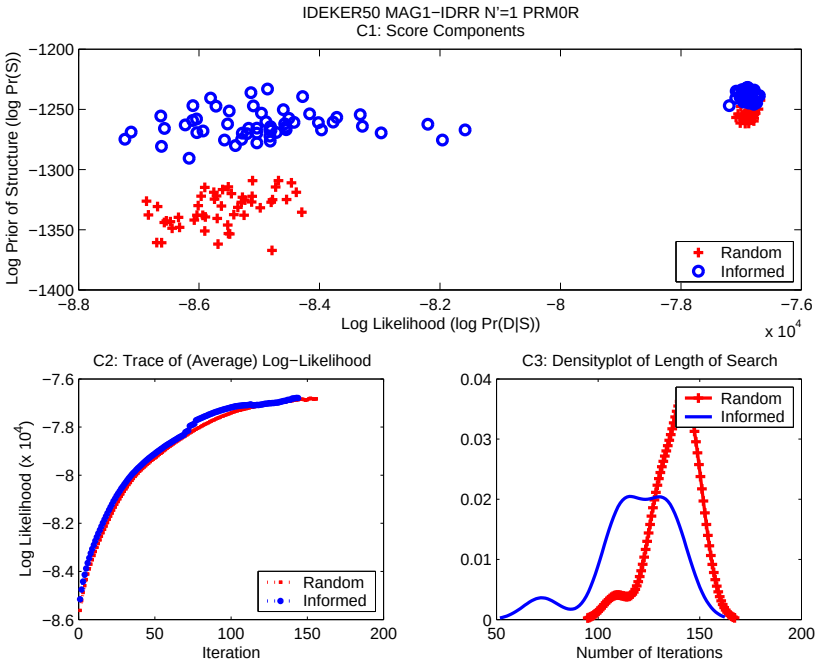


Figure F.91: IDEKER50 MAG1-IDRR $N' = 1$ PRM0R

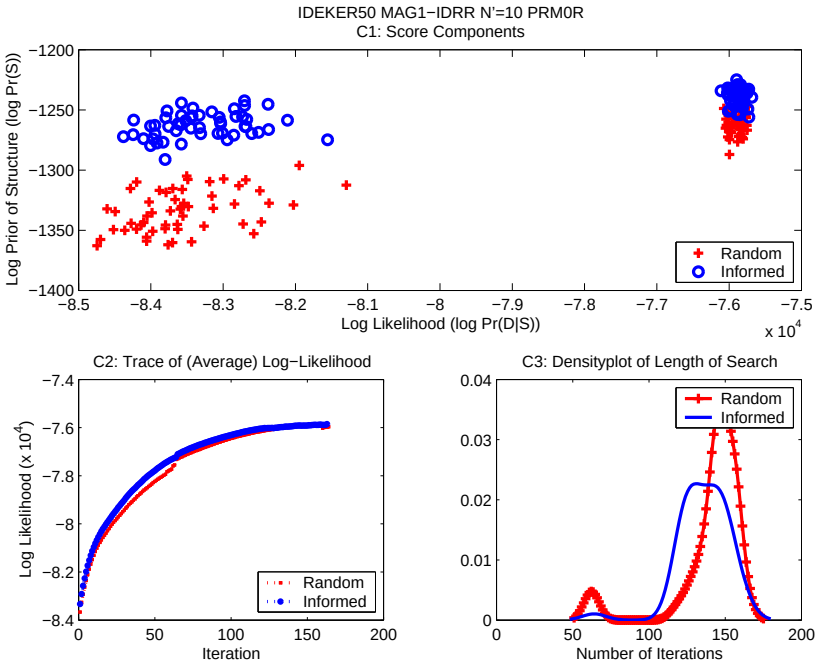


Figure F.92: IDEKER50 MAG1-IDRR $N' = 10$ PRM0R

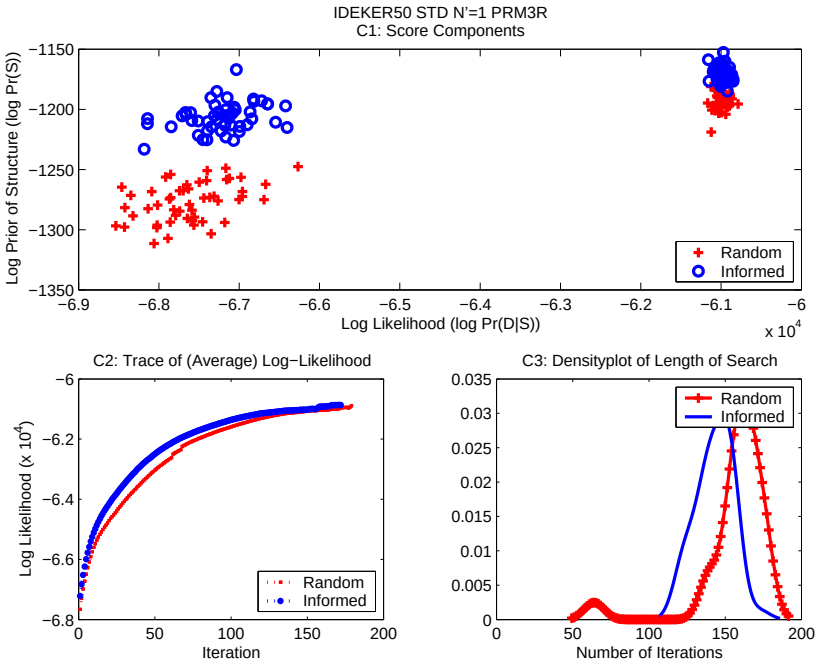


Figure F.93: IDEKER50 STD $N' = 1$ PRM3R

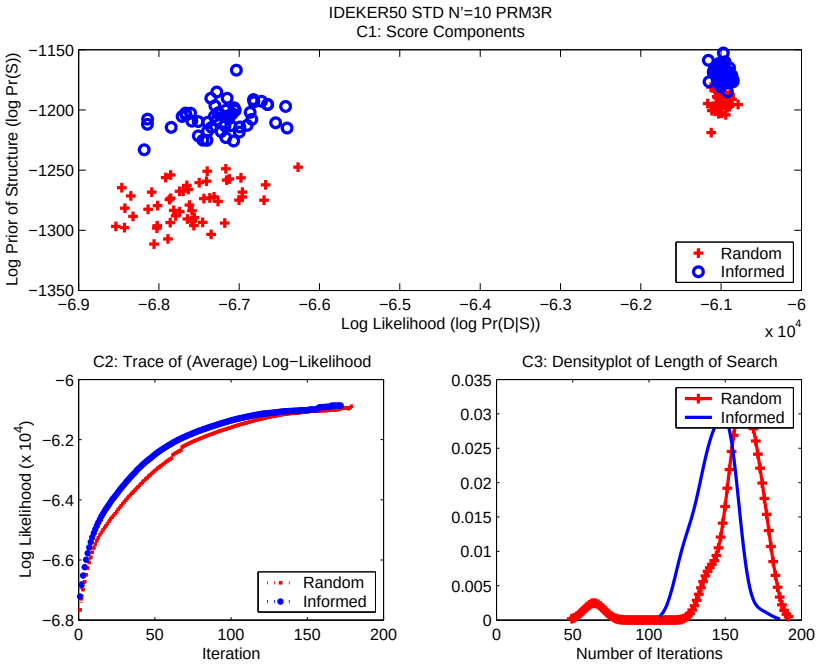


Figure F.94: IDEKER50 STD $N' = 10$ PRM3R

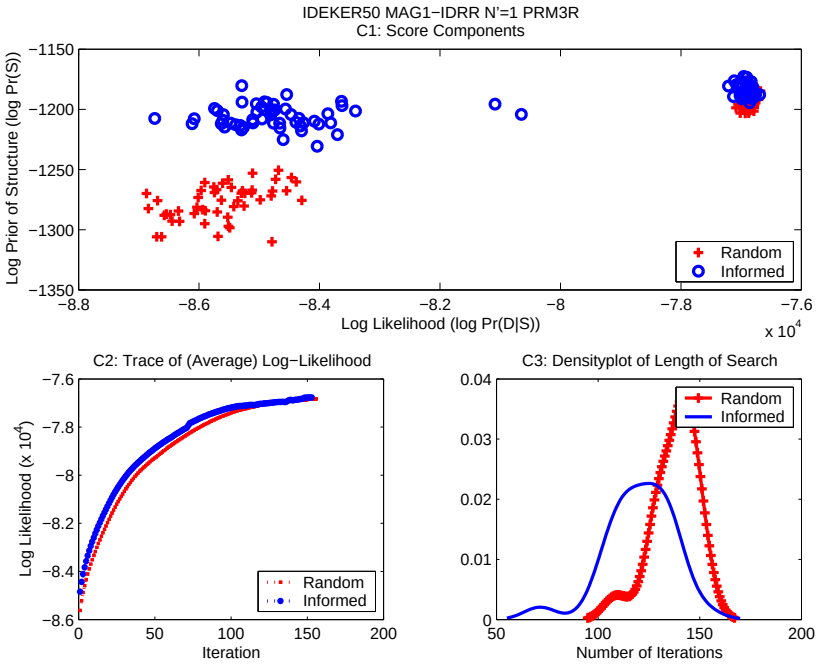


Figure F.95: IDEKER50 MAG1-IDRR $N' = 1$ PRM3R

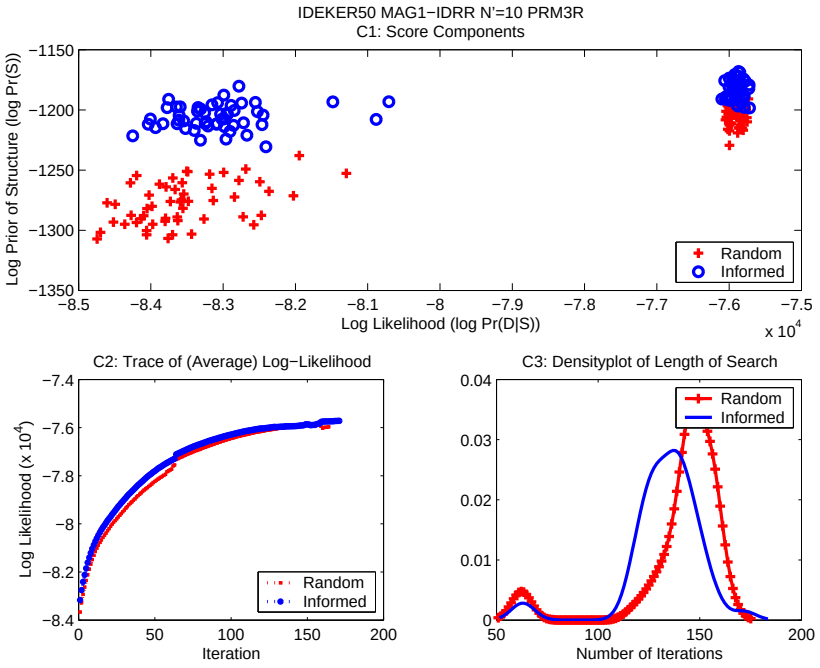


Figure F.96: IDEKER50 MAG1-IDRR $N' = 10$ PRM3R

Appendix G

Visualization of MG122 MOUSE Differential Expression Timeseries

In this chapter, we present full-scale displays of expression data superimposed on networks created from the high probability edges suggested by the **CONS** NoisyOR consensus likelihood function for the MG122 gene set. Graphs include all edges with a probability $Pr(e_{ij}) > 0.52$ according to the **CONS** NoisyOR consensus function (ignoring edge orientation). We connect each orphan node to its maximum probability neighbor, where that probability must be larger than the default edge probability $\mathcal{R}_0$ provided by our default expert and by definition is less than 0.52. We then apply the Organic Layout option in the Cytoscape drawing program [SMO+03] to the graph to create a substrate for further visualization.

Oligonucleotide microarrays containing markers for over 12000 mouse genes are applied to 11 mammary tissue samples staged from virgin mouse to post-lactation (involution), using four replicate samples per timepoint. Timepoints include Virgin Mouse (at four days post-ovariectomy), Pregnancy (Day 1, 3, 7, 12, 17, and 19), Lactation (Day 1, 2, and 9) and Involution (Day 2). All displays of expression data in the following graphs show data per gene transformed by taking the average of the four sample replicates, calculating $\log_2$ ratios of the (replicate average) expression value in the sample for a given timepoint relative to the average across all timepoints, then normalizing to have zero mean and variance one across the timepoints. Thus a value of -1 represents a two-fold down-regulation of the gene with respect to the average while a value of 2 represents a four-fold up-regulation of the gene with respect to the average.

Nodes are animated according to magnitude and direction of differential gene expression. Nodes are colored based on the degree of differential expression of a gene in a sample at a given timepoint to the average for that gene across all 11 samples. We use red to indicate a gene is expressed above the mean whereas green indicates the gene is expressed below the mean. Nodes are sized based on the degree of differential expression such that larger nodes mean a greater magnitude of over- or under-expression.

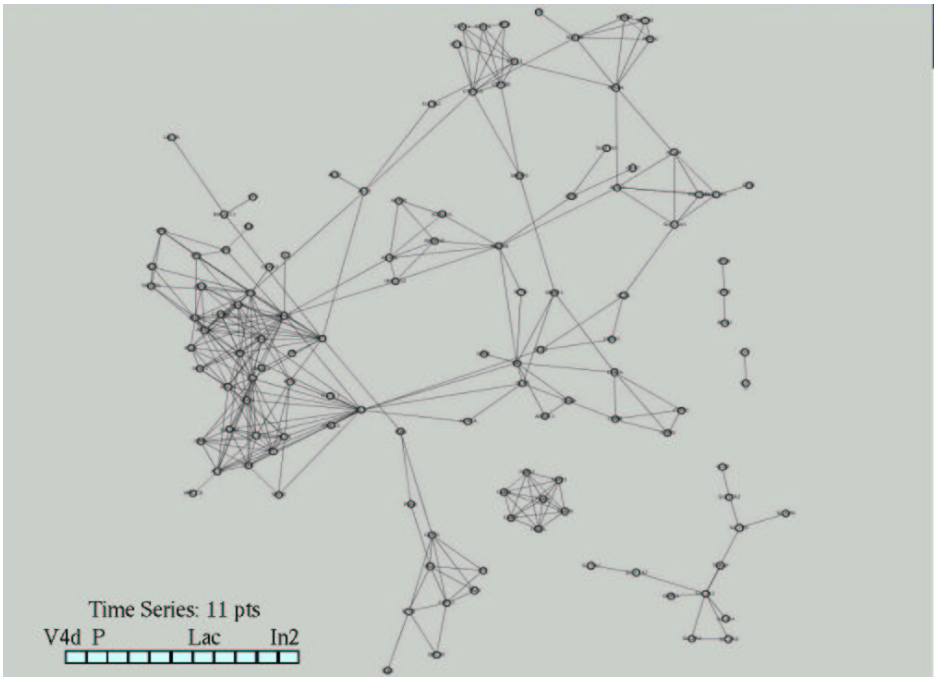


Figure G.1: Differential Expression Visualization for MG122 MOUSE Gene Set

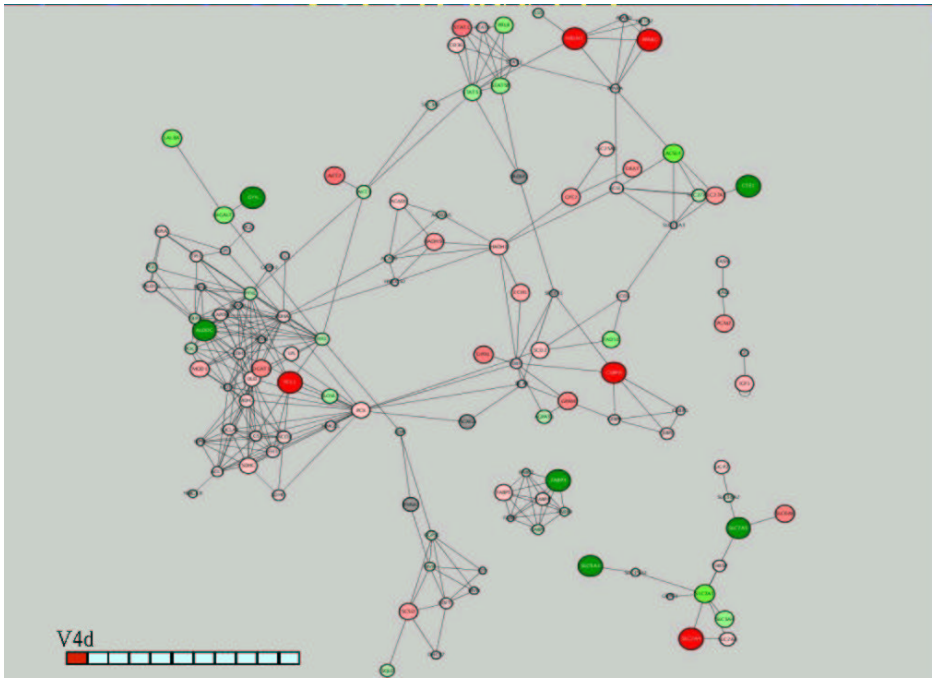


Figure G.2: Differential Expression Timepoint Virgin Mouse for MG122 MOUSE Gene Set

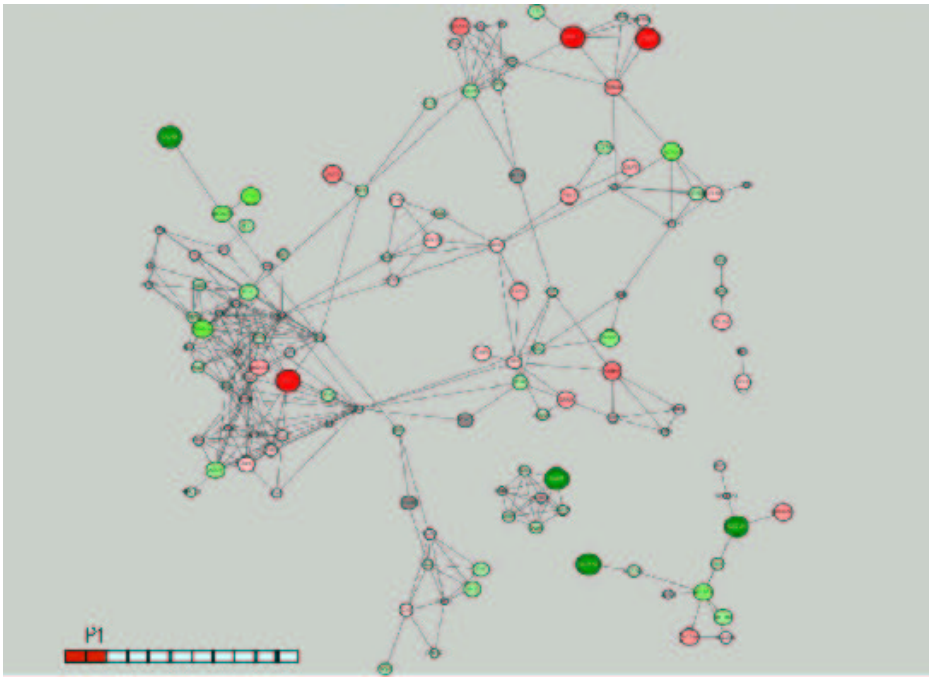


Figure G.3: Differential Expression Timepoint Pregnancy Day 1 for MG122 MOUSE Gene Set

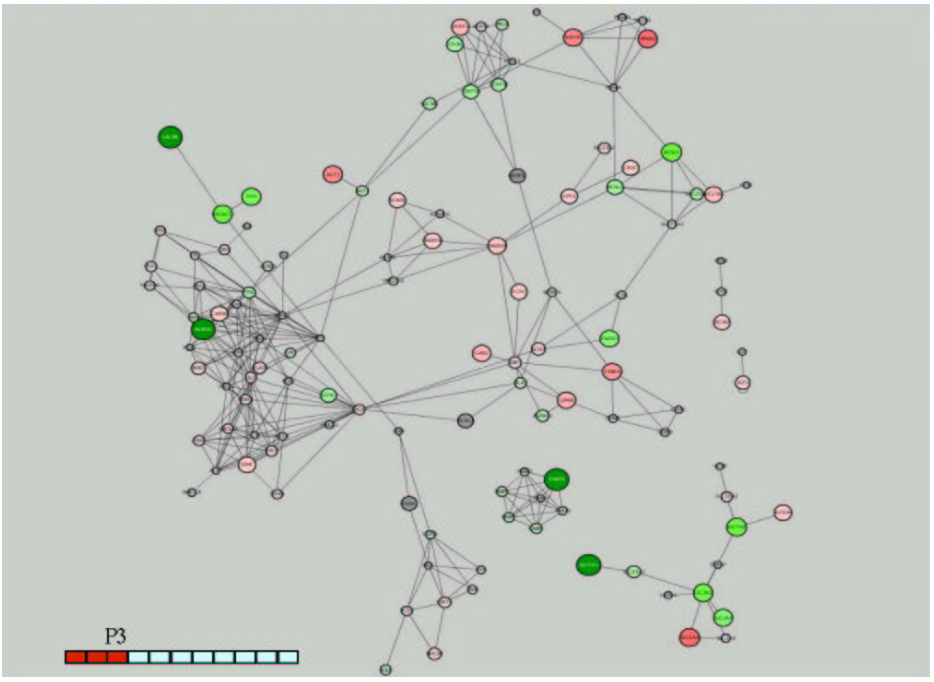


Figure G.4: Differential Expression Timepoint Pregnancy Day 3 for MG122 MOUSE Gene Set

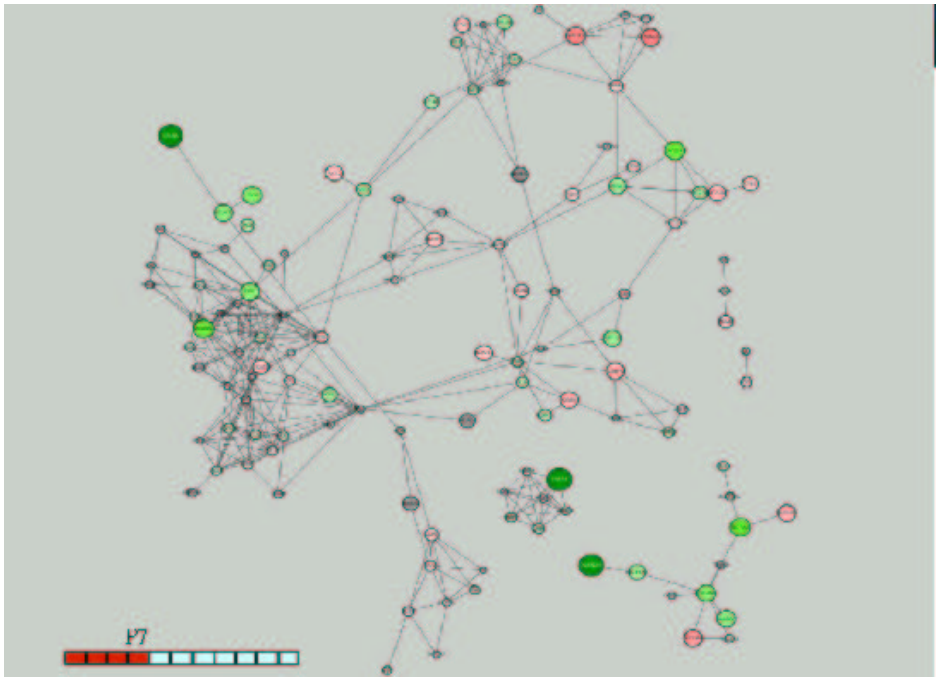


Figure G.5: Differential Expression Timepoint Pregnancy Day 7 for MG122 MOUSE Gene Set

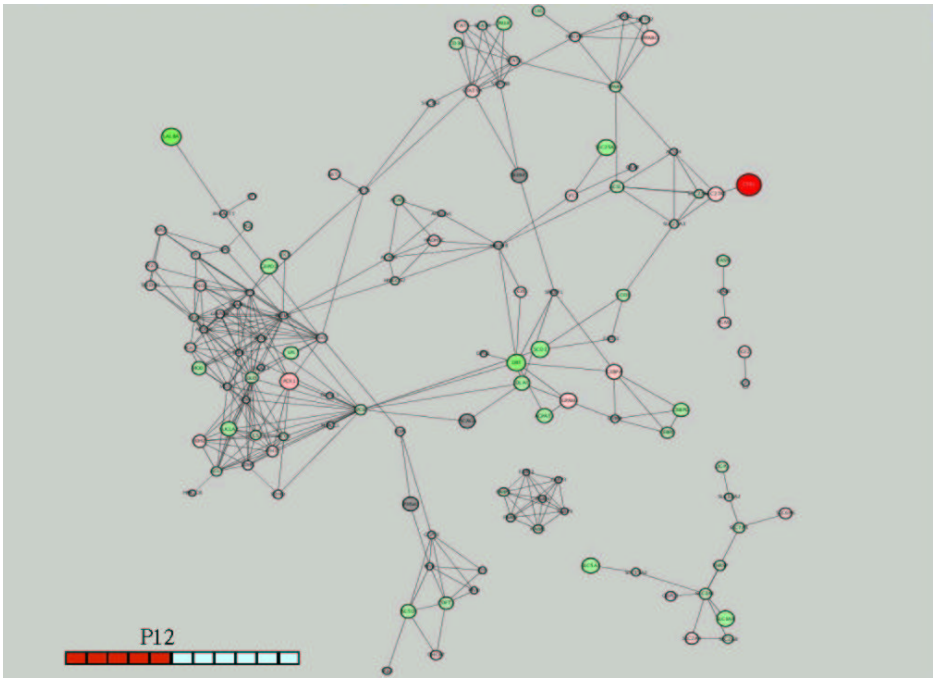


Figure G.6: Differential Expression Timepoint Pregnancy Day 12 for MG122 MOUSE Gene Set

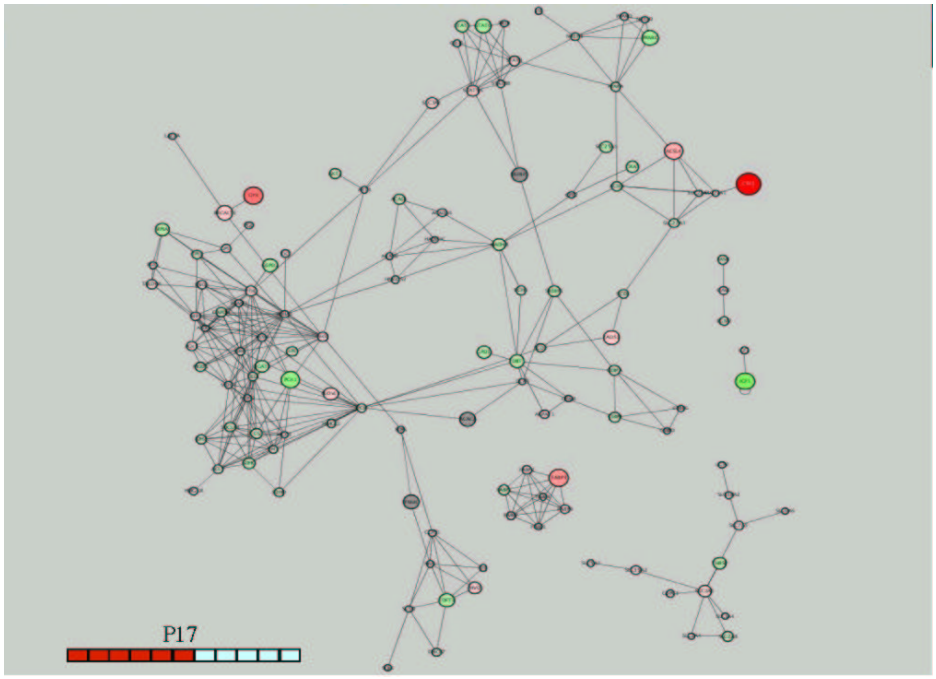


Figure G.7: Differential Expression Timepoint Pregnancy Day 17 for MG122 MOUSE Gene Set

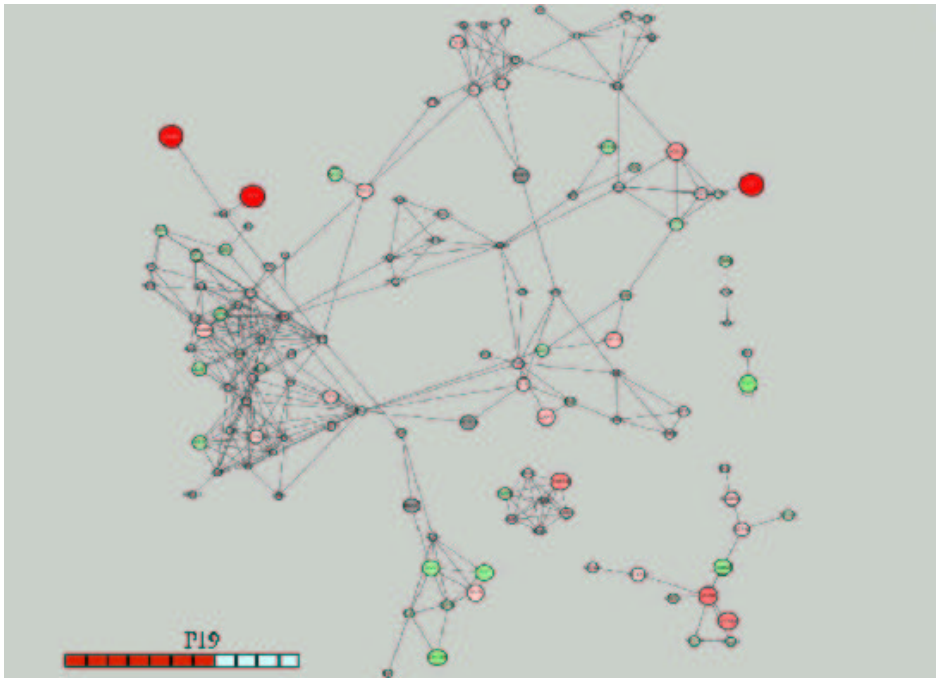


Figure G.8: Differential Expression Timepoint Pregnancy Day 19 for MG122 MOUSE Gene Set

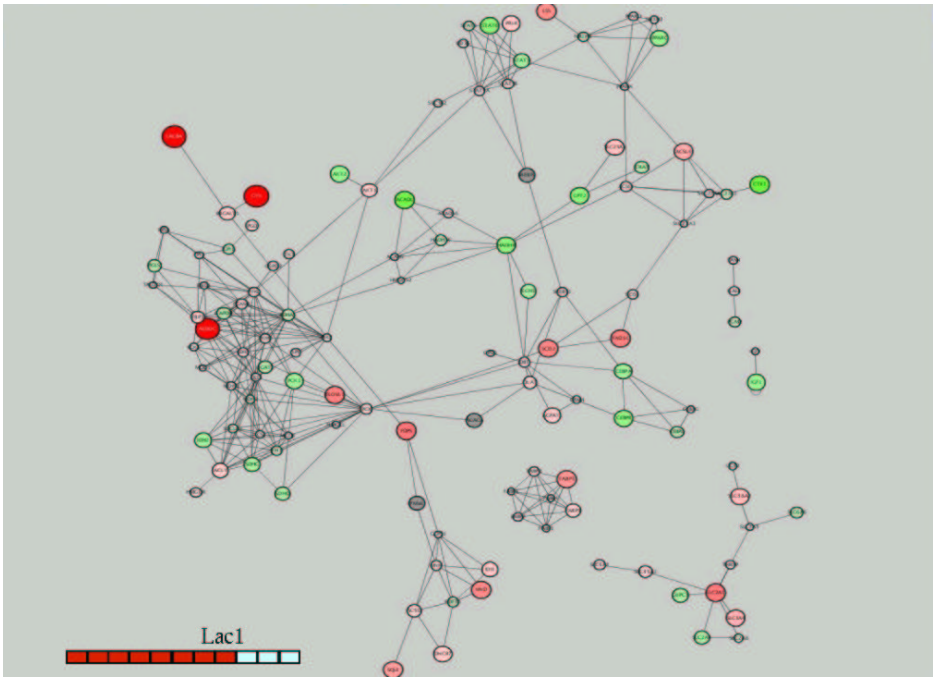


Figure G.9: Differential Expression Timepoint Lactation Day 1 for MG122 MOUSE Gene Set

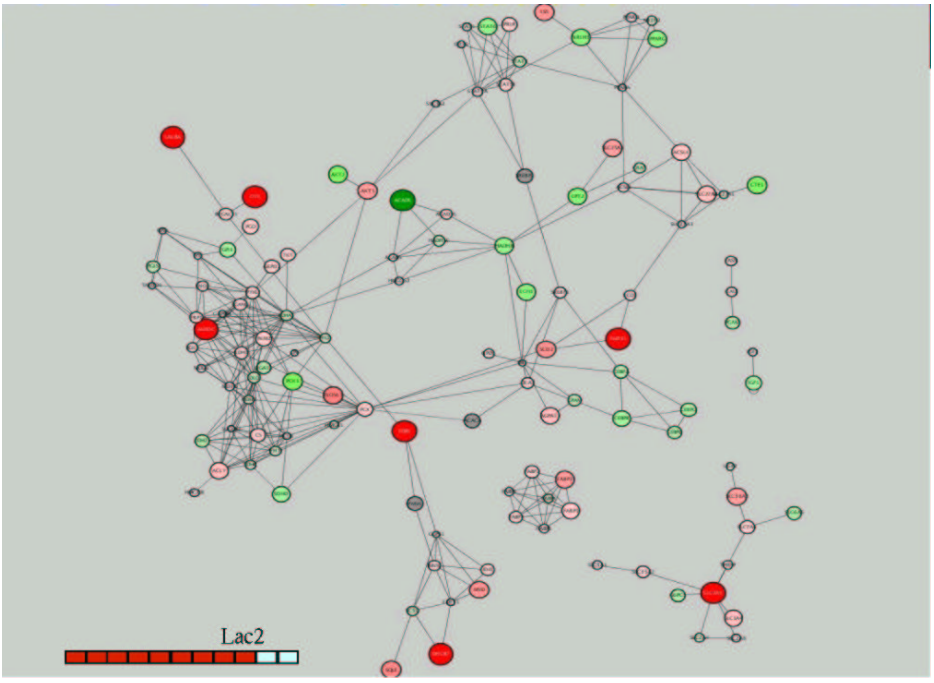


Figure G.10: Differential Expression Timepoint Lactation Day 2 for MG122 MOUSE Gene Set

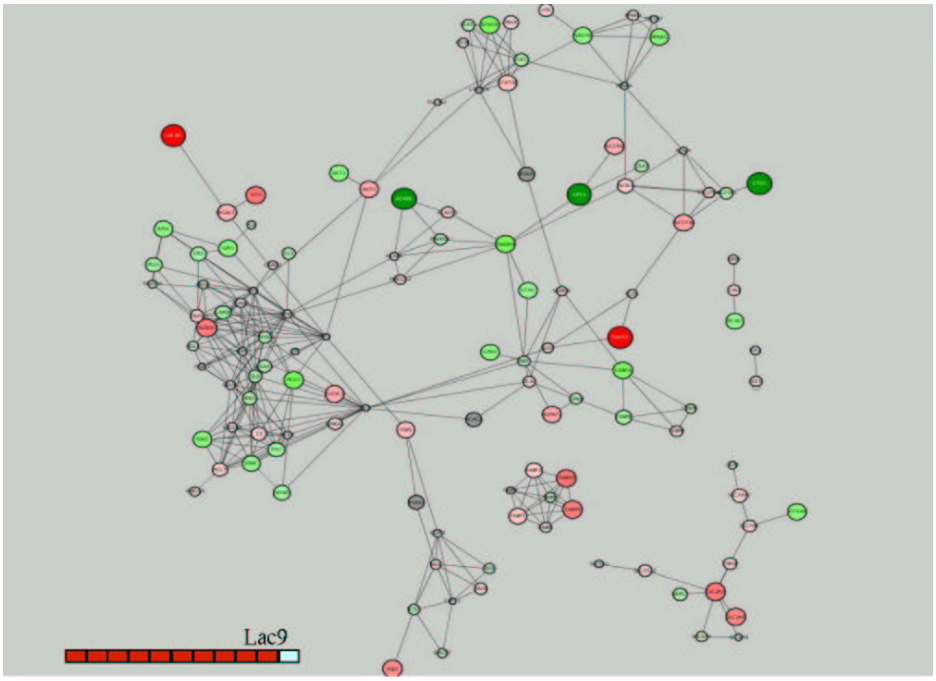


Figure G.11: Differential Expression Timepoint Lactation Day 9 for MG122 MOUSE Gene Set

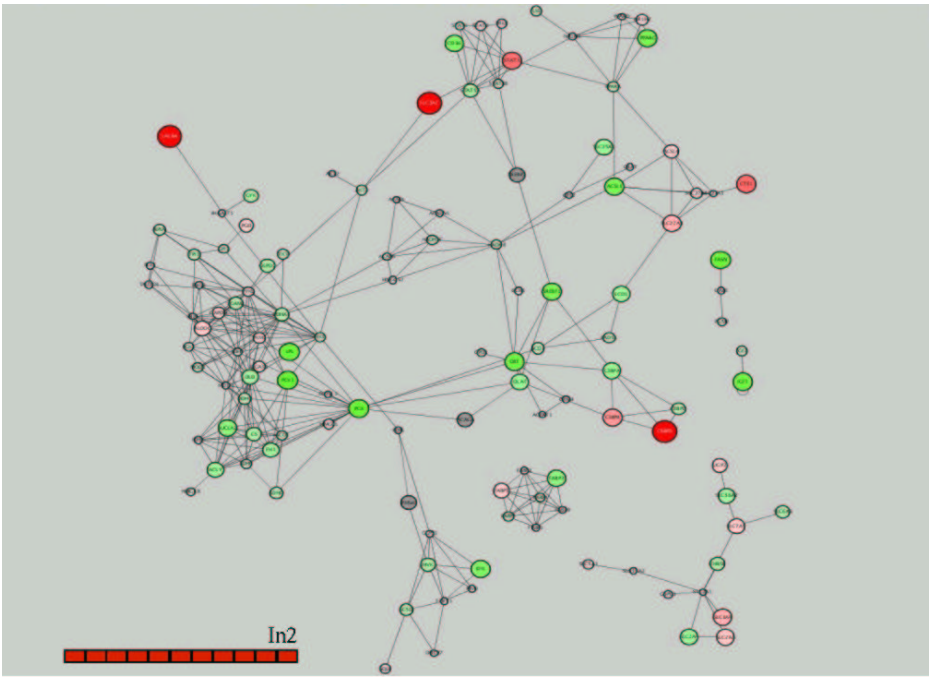


Figure G.12: Differential Expression Timepoint Involution Day 2 for MG122 MOUSE Gene Set

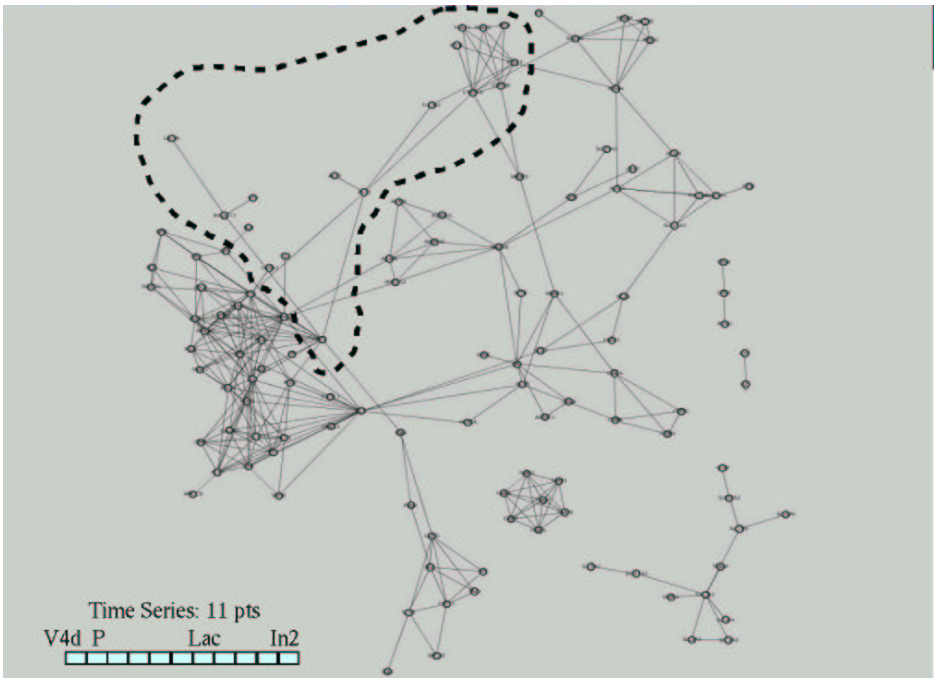


Figure G.13: Differential Expression Visualization for MG122 MOUSE (subgraph)

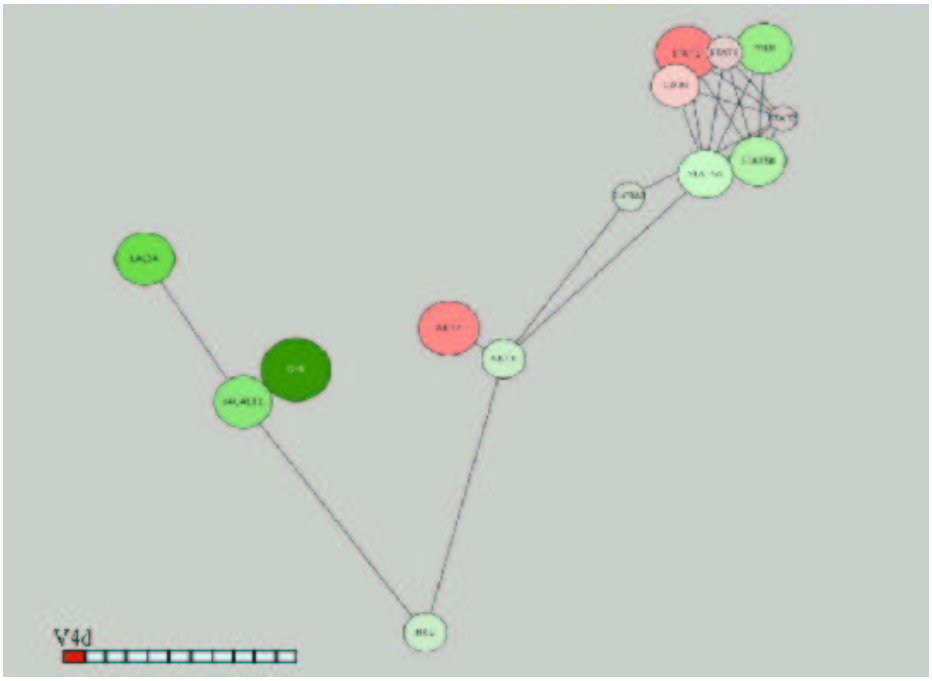


Figure G.14: Differential Expression Timepoint Virgin Mouse for MG122 MOUSE (subgraph)

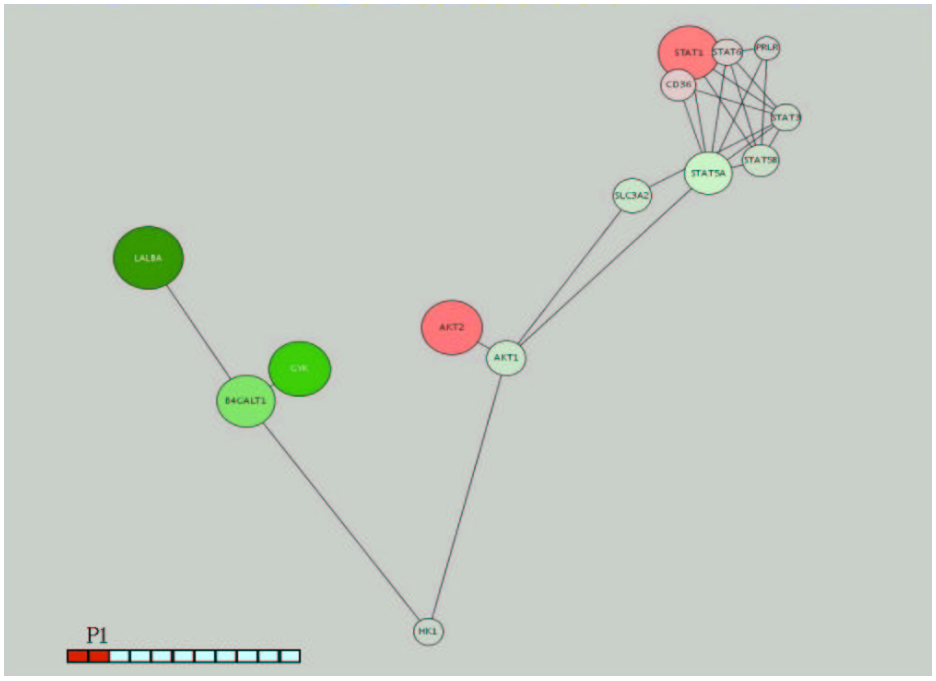


Figure G.15: Differential Expression Timepoint Pregnancy Day 1 for MG122 MOUSE (subgraph)

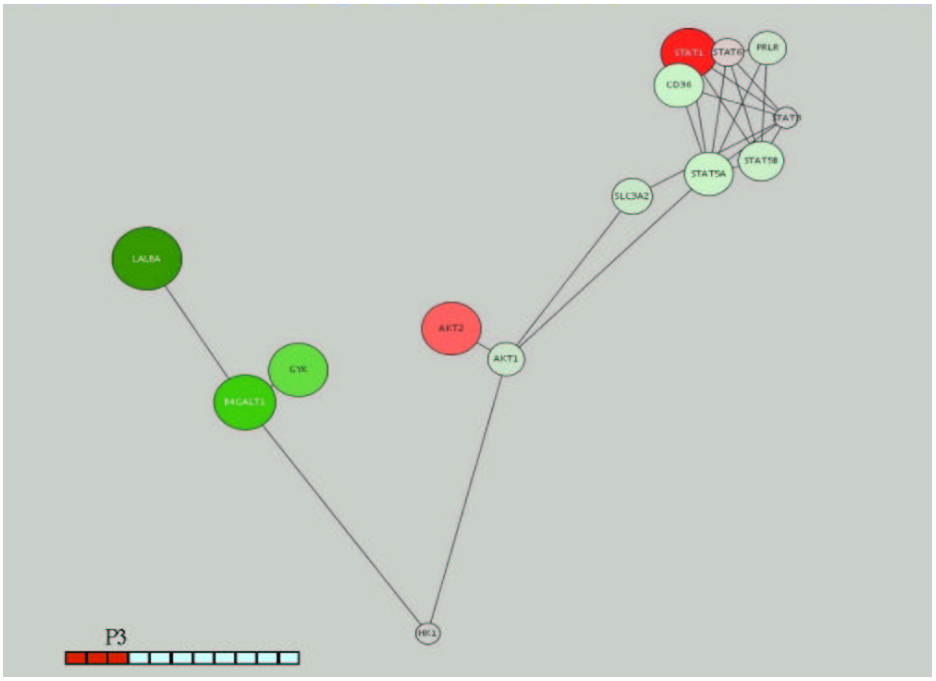


Figure G.16: Differential Expression Timepoint Pregnancy Day 3 for MG122 MOUSE (subgraph)

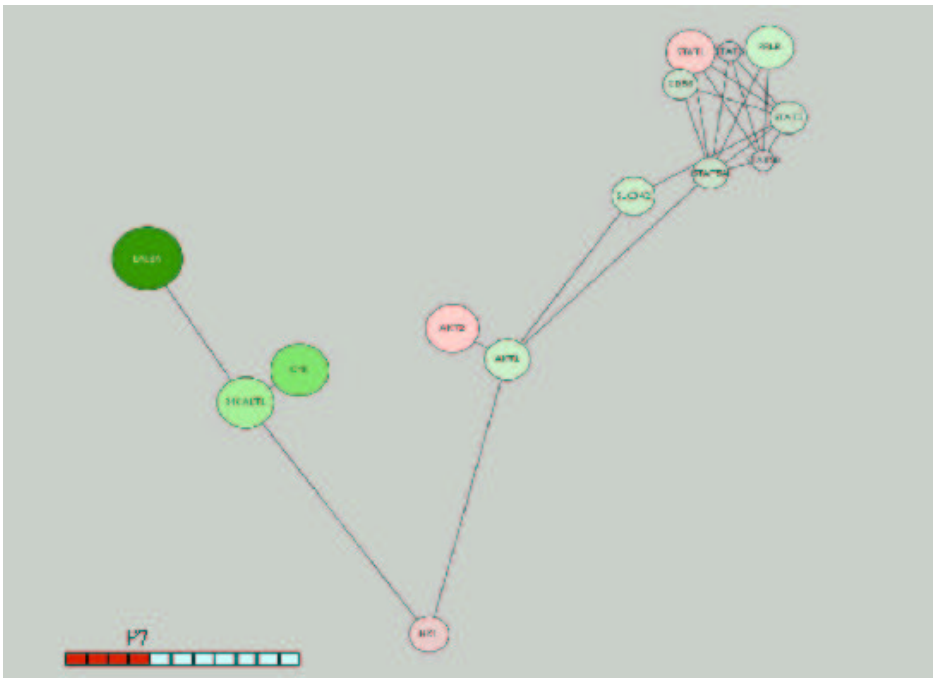


Figure G.17: Differential Expression Timepoint Pregnancy Day 7 for MG122 MOUSE (subgraph)

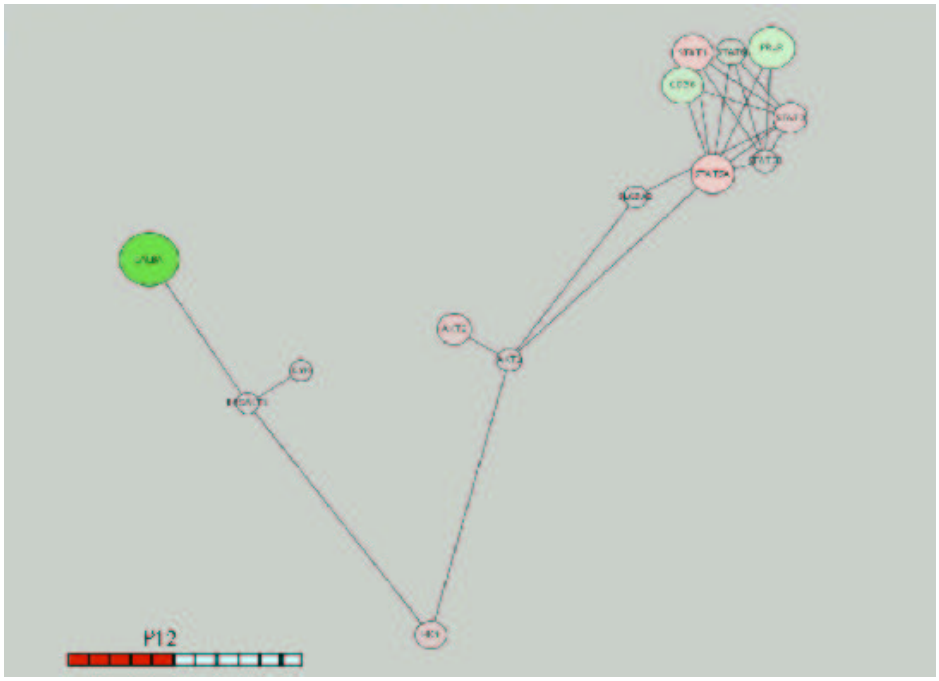


Figure G.18: Differential Expression Timepoint Pregnancy Day 12 for MG122 MOUSE (subgraph)

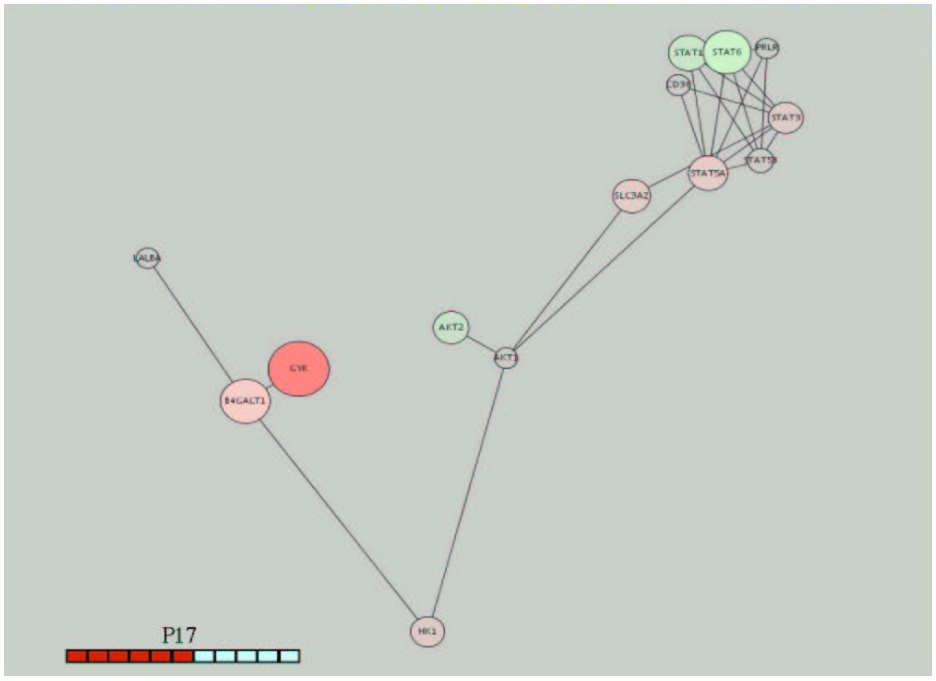


Figure G.19: Differential Expression Timepoint Pregnancy Day 17 for MG122 MOUSE (subgraph)

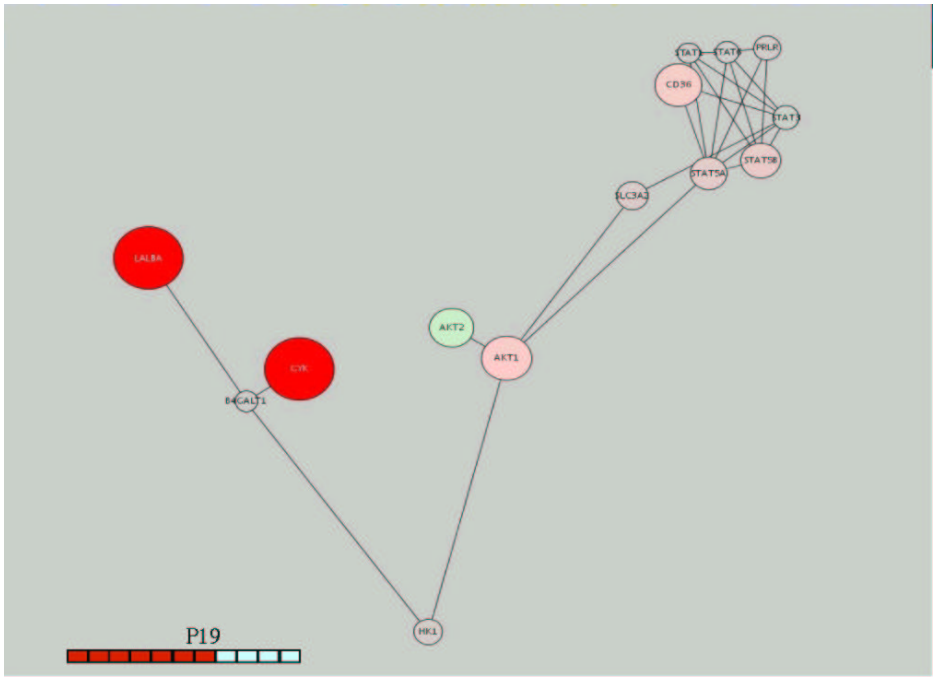


Figure G.20: Differential Expression Timepoint Pregnancy Day 19 for MG122 MOUSE (subgraph)

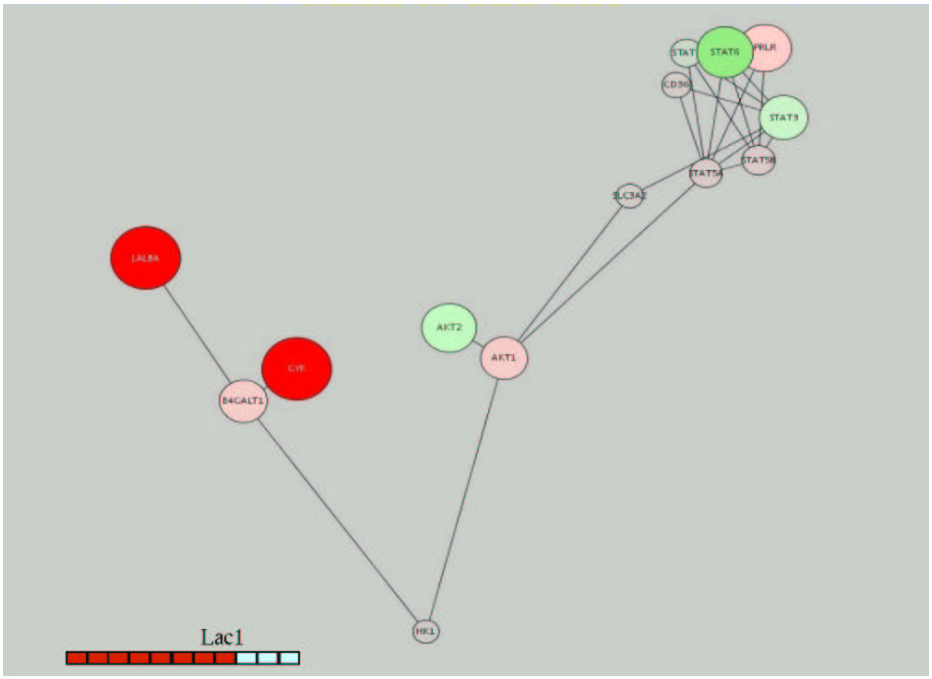


Figure G.21: Differential Expression Timepoint Lactation Day 1 for MG122 MOUSE (subgraph)

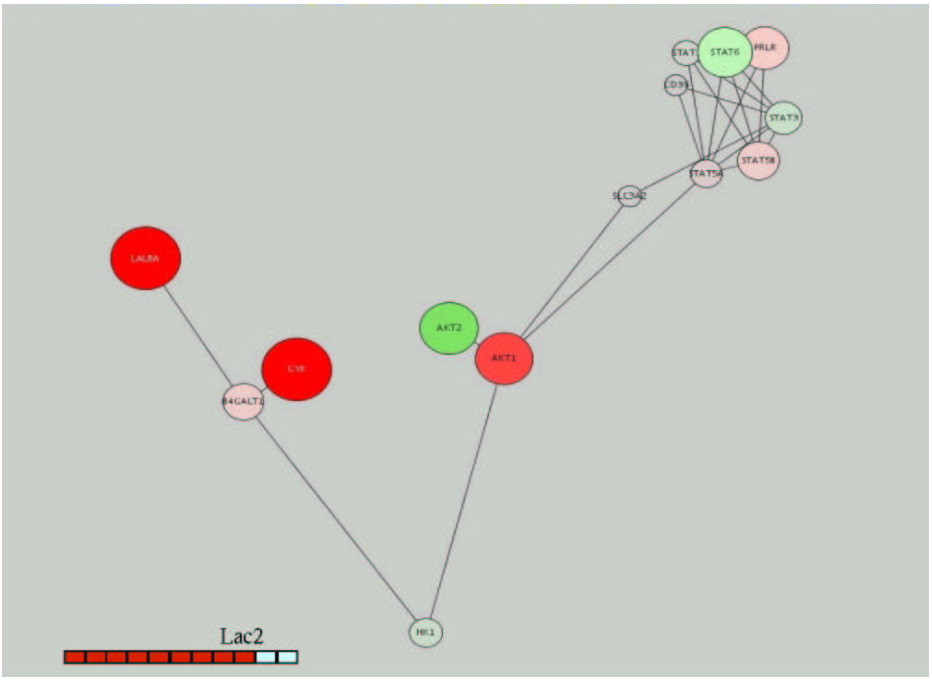


Figure G.22: Differential Expression Timepoint Lactation Day 2 for MG122 MOUSE (subgraph)

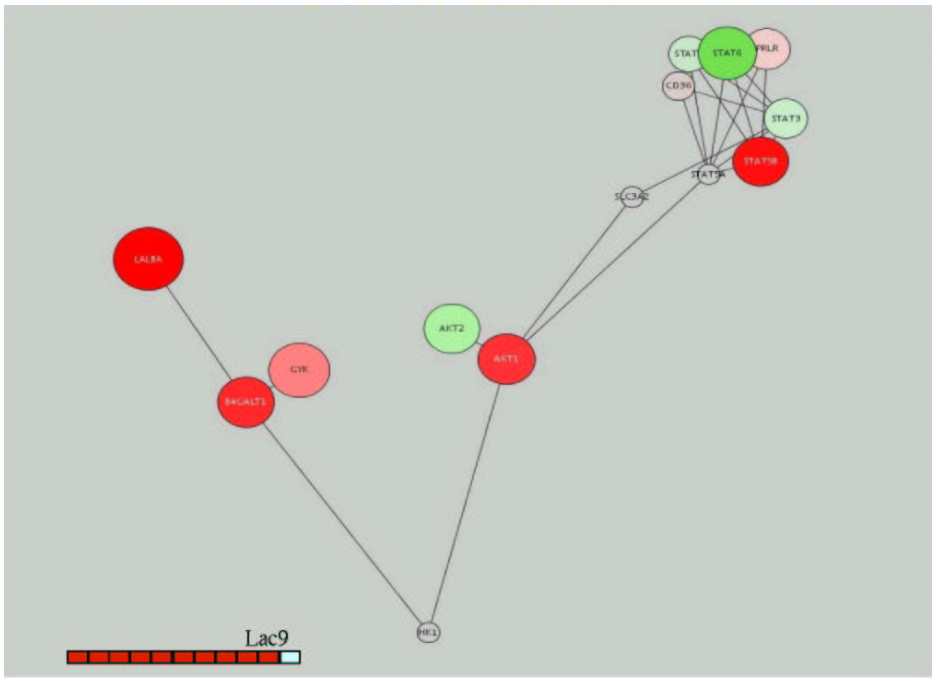


Figure G.23: Differential Expression Timepoint Lactation Day 9 for MG122 MOUSE (subgraph)

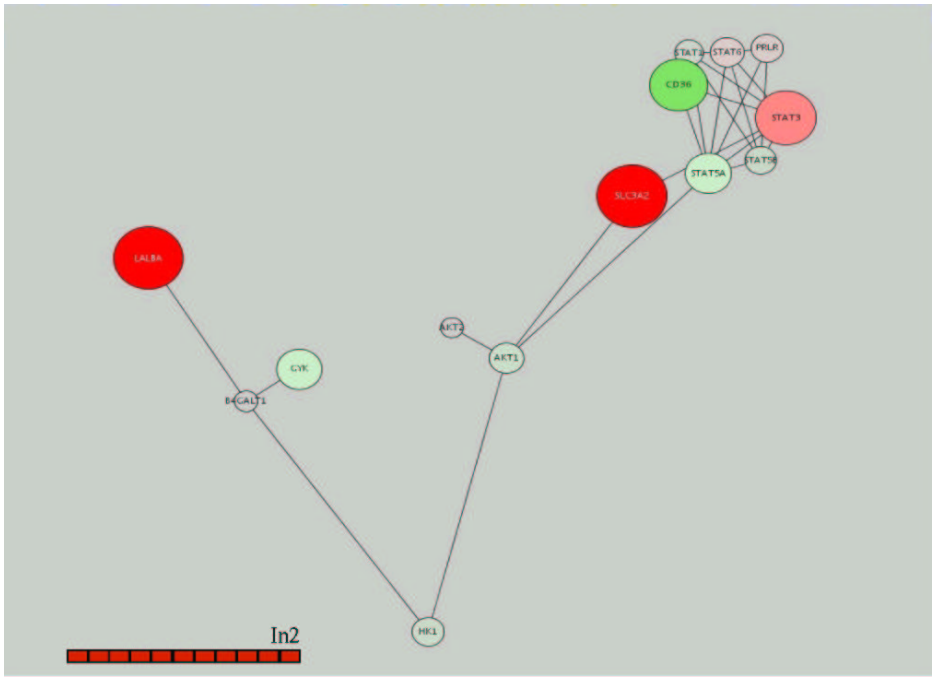


Figure G.24: Differential Expression Timepoint Involution Day 2 for MG122 MOUSE (subgraph)

Bibliography

[ABB+00] Michael Ashburner, Catherine A Ball, Judith A Blake, David Botstein, Heather Butler, J Michael Cherry, Allan P Davis, Kara Dolinski, Selina S Dwight, Janan T Eppig, Midori A Harris, David P Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C Matese, Joel E Richardson, Martin Ringwald, Gerald M Rubin, and Gavin Sherlock. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

[ABN+99] U Alon, N Barkai, DA Notterman, K Gish, S Ybarra, D Mack, and AJ Levine. 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 USA*, 96:6745–6750, 1999.

[AED+00] Ash Alizadeh, Michael Eisen, R Eric Davis, Chi Ma, Izidore Lossos, Andreas Rosenwald, Jennifer Boldrick, Hajeer Sabet, Truc Tran, Xin Yu, John Powell, Liming Yang, Gerald Marti, Troy Moore, James Hudson Jr, Lisheng Lu, David Lewis, Robert Tibshirani, Gavin Sherlock, Wing Chan, Timothy Greiner, Dennis Weisenburger, James Armitage, Roger Warnke, Ronald Levy, Wyndham Wilson, Michael Grever, John Byrd, David Botstein, Patrick Brown, and Louis Staudt. Distinct types of diffuse large B-cell lymphoma identifed by expression profiling. *Nature*, 403:503–511, 2000.

[AMK99] T Akutsu, S Miyano, and S Kuhara. Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Pacific Symposium on Biocomputing (PSB)*, volume 4, pages 17–28, 1999.

[AMK00] T Akutsu, S Miyano, and S Kuhara. Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, 16(8):727–34, 2000.

[Bad03] JS Bader. Greedily building protein networks with confidence. *Bioinformatics*, 19(15):1869–74, 2003.

[Bay01] Stephen D Bay. Multivariate discretization for set mining. *Knowledge and Information Systems*, 3(4):491–512, 2001.

[BCD+04] Alex Bateman, Lachlan Coin, Richard Durbin, Robert D Finn, Volker Hollich, Sam Griffiths-Jones, Ajay Khanna, Mhairi Marshall, Simon Moxon, Erik LL Sonnhammer, David J Studholme, Corin Yeats, and Sean R Eddy. The Pfam protein families database. *Nucleic Acids Research*, 32:D138–D141, 2004.

[BCRC04] JS Bader, A Chaudhuri, JM Rothberg, and J Chant. Gaining confidence in high-throughput protein interaction networks. *Nat Biotechnol*, 22(1):78–85, 2004.

[BDS00] Bikramjit Banerjee, Sandip Debnath, and Sandip Sen. Combining multiple perspectives. In *International Conference on Machine Learning (ICML)*, pages 33–40, San Francisco CA, 2000. Morgan Kaufmann.

[BDW+01] G Bader, I Donaldson, C Wolting, B Ouellette, T Pawson, and C Hogue. BIND–the biomolecular interaction network database. *Nucleic Acids Research*, 29(1):242–245, 2001.

[BF01] Yoseph Barash and Nir Friedman. Context-specific Bayesian clustering for gene expression data. In *Proceedings of the Fifth Annual International Conference on Computational Molecular Biology - RECOMB 01*, pages 12–21, 2001.

[BFOS84] L Breiman, J Friedman, R Olshen, and CJ Stone. *Classification and Regression Trees.* Chapman and Hall, 1984.

[BH01] Alexander Budanitsky and Graeme Hirst. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Proceedings of Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, June 2001.

[BH02] GD Bader and CW Hogue. Analyzing yeast protein-protein interaction data obtained from different sources. *Nat Biotechnol*, 20(10):991–7, 2002.

[BH03] GD Bader and CW Hogue. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics*, 4(1):2, 2003.

[BM98] CL Blake and CJ Merz. UCI repository of machine learning databases, 1998.

[BP99] Stephen D Bay and Michael J Pazzani. Detecting change in categorical data: Mining contrast sets. In *Knowledge Discovery and Data Mining*, pages 302–306, 1999.

[Bra99] Matthew Brand. Structure learning in conditional probability models via and entropic prior and parameter extinction. *Neural Computation*, 11(5):1155–1182, 1999.

[BSCC89] I Beinlich, G Suermondt, R Chavez, and G Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proc. 2nd European Conf. on AI and Medicine*, Berlin, 1989. Springer–Verlag.

[BSK05] A Battle, E Segal, and D Koller. Probabilistic discovery of overlapping cellular processes and their regulation. *J Comput Biol*, 12(7):909–27, 2005.

[Bun91] Wray Buntine. Theory refinement on Bayesian networks. In *UAI*, pages 52–60, 1991.

[BV98] Tom Brijs and Koen Vanhoof. Cost sensitive discretization of numeric attributes. In *Principles of Data Mining and Knowledge Discovery*, pages 102–110, 1998.

[BY98] James O Berger and Ruoyang Yang. A catalog of noninformative priors. http://www.stat.missouri.edu/ bayes/catalog.ps, December 1998.

[CA04] Noan-Minh Chau and Margaret Ashcroft. Akt2: a role in breast cancer metastasis. *Breast Cancer Res*, 6(1):55–57, 2004.

[CAB+98] J Michael Cherry, Caroline Adler, Catherine Ball, Stephen A Chervitz, Selina S Dwight, Erich T Hester, Yankai Jia, Gail Juvik, TaiYun Roe, Mark Schroeder, Shuai Weng, and David Botstein. SGD: Saccharomyces Genome Database. *Nucleic Acids Research*, 26(1):73–80, 1998.

[Cat91] J Catlett. On changing continuous attributes into ordered discrete attributes. In *Proceedings of the European Working Session on Learning*, pages 164–178, 1991.

[CCW+98] Raymond J Cho, Michael J Campbell, Elizabeth A Winzeler, Lars Steinmetz, Andrew Conway, Lisa Wodicka, Tyra G Wolfsberg, Andrei E Gabrielian, David Landsman, David J Lockart, and Ronald W Davis. A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell*, 2:65–73, 1998.

[CDE+98] S Chu, J DeRisi, M Eisen, J Mulholland, D Botstein, P O Brown, and I Herkowitz. The transcriptional program of sporulation in budding yeast. *Science*, 282:699–705, 1998.

[CH91] Gregory F Cooper and Edward Herskovits. A Bayesian method for constructing Bayesian belief networks from databases. In *UAI*, pages 86–94, 1991.

[CH92] Gregory F Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[CH97] David Maxwell Chickering and David Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29(2–3):181–212, 1997.

[CHC99] T Chen, HL He, and GM Church. Modeling gene expression with differential equations. In *Pacific Symposium on Biocomputing (PSB)*, volume 4, pages 29–40, 1999.

[Chi96] David Maxwell Chickering. Learning equivalence classes of Bayesian network structures. In *UAI*, pages 150–157, 1996.

[CTC+01] Han Cho, Joanne L Thorvaldsen, Qingwei Chu, Fei Feng, and Morris J Birnbaum. Akt1/PKB is required for normal growth but dispensable for maintenance of glucose homeostasis in mice. *J Biol Chem*, 276(42):38349–38352, 2001.

[CW99] RT Clemen and RL Winkler. Combining probability distributions from experts in risk analysis. *Risk Analysis*, 19(2):187–203, 1999.

[CW03] Richard WE Clarkson and Christine J Watson. Microarray analysis of the involution switch. *Journal of Mammary Gland Biology and Neoplasia*, 8(3):309–319, 2003.

[DeG70] Morris H DeGroot. *Probability and Statistics*. Addison Wesley, 1970.

[DGT02] F Denis, R Gilleron, and M Tommasi. Text classification from positive and unlabeled examples. In *The 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2002*, 2002.

[DHHB05] Harold J Drabkin, Christopher Hollenbeck, David P Hill, and Judith A Blake. Ontological visualization of protein-protein interactions. *BMC Bioinformatics*, 6:29, 2005.

[dHIK+03] MJL de Hoon, S Imoto, K Kobayashi, N Ogasawara, and S Miyano. Inferring gene regulatory networks from time-ordered gene expression data of Bacillus subtilis using differential equations. In *Pacific Symposium on Biocomputing (PSB)*, volume 8, pages 17–28, 2003.

[DIB97] Joseph L DeRisi, Vishwanath R Iyer, and Patrick O Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278:680–686, 1997.

[DKS95] James Dougherty, Ron Kohavi, and Mehran Sahami. Supervised and unsupervised discretization of continuous features. In *International Conference on Machine Learning*, pages 194–202, 1995.

[DL93] A Dawid and S Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Annals of Statistics*, 21:1272–1317, 1993.

[DLMP03] Richard Dybowski, Kathryn B Laskey, James W Myers, and Simon Parsons. Introduction to the special issue on the fusion of domain knowledge with data for decision support. *Journal of Machine Learning Research*, 4:293–294, 2003.

[DLR77] A Dempster, N Laird, and D Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39:1–38, 1977.

[dM93] Thierry Van de Merckt. Decision trees in numerical attribute spaces. In *International Joint Conference on AI (IJCAI)*, pages 1016–1021, 1993.

[DMHK95] I Dubchak, I Muchnik, SR Holbrook, and S Kim. Prediction of protein folding class using global description of amino acid sequence. *Proceedings of the National Academy of Sciences USA (PNAS)*, 92:8700–8704, 1995.

[DSB+01] Becky L Drees, Bryan Sundin, Elizabeth Brazeau, Juliane P Caviston, Guang-Chao Chen, Wei Guo, Keith G Kozminski, Michelle W Lau, John J Moskow, Amy Tong, Laura R Schenkman, Amos McKenzie III, Patrick Brennwald, Mark Longtine, Erfei Bi, Clarence Chan, Peter Novick, Charles Boone, John R Pringle, Trisha N Davis, Stanley Fields, and David G Drubin. A protein interaction map for cell polarity development. *Journal of Cell Biology*, 154(3):549–571, 2001.

[DSV+02] KD Dahlquist, N Salomonis, K Vranizan, SC Lawlor, and BR Conklin. GenMAPP, a new tool for viewing and analyzing microarray data on biological pathways. *Nature Genetics*, 31(1):19–20, 2002.

[DSXE02] CM Deane, L Salwinski, I Xenarios, and D Eisenberg. Protein interactions: two methods for assessment of the reliability of high throughput observations. *Mol Cell Proteomics*, 1(5):349–56, 2002.

[DTSC04] M Deng, Z Tu, F Sun, and T Chen. Mapping gene ontology to proteins based on protein-protein interaction data. *Bioinformatics*, 20(6):895–902, 2004.

[DWFS99] P D'haeseleer, X Wen, S Fuhrman, and R Somogyi. Linear modeling of mRNA expression levels during CNS development and injury. In *Pacific Symposium on Biocomputing (PSB)*, volume 4, pages 41–52, 1999.

[EBK+05] Janan T Eppig, Carol J Bult, James A Kadin, Joel E Richardson, and Judith A Blake. The mouse genome database (MGD): from genes to mice, a community resource for mouse biology. *Nucleic Acids Res*, 33:D471–D475, 2005.

[EIKO99] Anton J Enright, Ioannis Iliopoulos, NIkos C Kyrpides, and Christos A Ouzounis. Protein interaction maps for complete genomes based on gene fusion events. *Nature*, 402:86–90, 1999.

[Fel98] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge MA, 1998.

[FG96] Nir Friedman and Moises Goldszmidt. Learning Bayesian networks with local structure. In *UAI*, pages 252–262, 1996.

[FGKP99] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning Probabilistic Relational Models. In *IJCAI*, 1999.

[FI93] UM Fayyad and KB Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *International Joint Conference on AI (IJCAI)*, pages 1022–1027, 1993.

[FK03] Nir Friedman and Daphne Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1–2):95–125, 2003.

[FLNP00] N Friedman, M Linial, I Nachman, and D Pe'er. Using Bayesian networks to analyze expression data. *J Comput Biol*, 7(3-4):601–620, 2000.

[FNP99] Nir Friedman, Iftach Nachman, and Dana Pe'er. Learning Bayesian network structure from massive datasets: The sparse candidate algorithm. In *UAI*, pages 206–215, 1999.

[FR91] Thomas MJ Fruchterman and Edward M Reingold. Graph drawing by force-directed placement. *Software Practice and Experience*, 21(11):1129–1164, 1991.

[Fri98] Nir Friedman. The Bayesian structural EM algorithm. In *UAI*, pages 129–138, 1998.

[FW99] Eibe Frank and Ian H Witten. Making better use of global discretization. In *International Conference on Machine Learning*, pages 115–123, 1999.

[GBK+02] Anne-Claude Gavin, Markus Bösche, Roland Krause, Paola Grandi, Martina Marzioch, Andreas Bauer, Jörg Schultz, Jens M Rick, Anne-Marie Michon, Cristina-Maria Cruciat, Marita Remor, Christian Höfert, Malgorzata Schelder, Miro Brajenovic, Heinz Ruffner, Alejandro Merino, Karin Klein, Manuela Hudak, David Dickson, Tatjana Rudi, Volker Gnau, Angela Bauch, Sonja Bastuck, Bettina Huhse, Christina Leutwein, Marie-Anne Heurtier, Richard R Copley, Angela Edelmann, Erich Querfurth, Vladimir Rybin, Gerard Drewes, Manfred Raida, Tewis Bouwmeester, Peer Bork, Bertrand Seraphin, Bernhard Kuster, Gitte Neubauer, and Giulio Superti-Furga. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 415:141–147, 2002.

[GCN+02] Guri Giaever, Angela M Chu, Li Ni, Carla Connelly, Linda Riles, Steeve Véronneau, Sally Dow, Ankuta Lucau-Danila, Keith Anderson, Bruno André, Adam P Arkin, Anna Astromoff, Mohamed El Bakkoury, Rhonda Bangham, Rocio Benito, Sophie Brachat, Stefano Campanaro, Matt Curtiss, Karen Davis, Adam Deutschbauer, Karl-Dieter Entian, Patrick Flaherty, Francoise Foury, David J Garfinkel, Mark Gerstein, Deanna Gotte, Ulrich Güldener, Johannes H Hegemann, Svenja Hempel, Zelek Herman, Daniel F Jaramillo, Diane E Kelly, Steven L Kelly, Peter Kötter, Darlene LaBonte, David C Lamb, Ning Lan, Hong Liang, Hong Liao, Lucy Liu, Chuanyun Luo, Marc Lussier, Rong Mao, Patrice Menard, Siew Loon Ooi, Jose L Revuelta, Christopher J Roberts, Matthias Rose, Petra Ross-Macdonald, Bart Scherens, Greg Schimmack, Brenda Shafer, Daniel D Shoemaker, Sharon Sookhai-Mahadeo, Reginald K Storms, Jeffrey N Strathern, Giorgio Valle, Marleen Voet, Guido Volckaert, Ching yun Wang, Teresa R Ward, Julie Wilhelmy, Elizabeth A Winzeler, Yonghong Yang, Grace Yen, Elaine Youngman, Kexin Yu, Howard Bussey, Jef D Boeke, Michael Snyder, Peter Philippsen, Ronald W Davis, and Mark Johnston. Functional profiling of the Saccharomyces cerevisiae genome. *Nature*, 418(6896):387–391, 2002.

[GHG] Aaron Gabow, Lawrence Hunter, and Debra Goldberg. Supplementing graph-theoretic protein function prediction methods with literature co-occurrence data. In review, contact first author for availability.

[GN00] Emden R Gansner and Stephen C North. An open graph visualization system and its applications to software engineering. *Software – Practice and Experience*, 30(11):1203–1233, 2000.

[GSK+00] Audrey P Gasch, Paul T Spellman, Camilla M Kao, Orna Carmel-Harel, Michael B Eisen, Gisela Storz, David Botstein, and Patrick O Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell*, 11:4241–4257, 2000.

[GST+99] T Golub, D Slonim, P Tamayo, C Huard, M Gaasenbeek, J Mesirov, H Coller, M Loh, J Downing, M Caligiuri, C Bloomfield, and E Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[GSTK01] Lise Getoor, Eran Segal, Ben Taskar, and Daphne Koller. Probabilistic models of text and link structure for hypertext classification. In *IJCAI Workshop on Text Learning: Beyond Supervision*, 2001.

[GZ86] C Genest and JV Zidek. Combining probability distributions: A critique and an annotated bibliography. *Statistical Science*, 1(1):114–148, 1986.

[Har75] John A Hartigan. *Clustering algorithms*. Wiley, 1975.

[Har01] Alexander J Hartemink. *Principled Computational Methods for the Validation and Discovery of Genetic Regulatory Networks*. PhD thesis, Massachusetts Institute of Technology, 2001.

[HB00] Ian Holmes and William J Bruno. Finding regulatory elements using joint likelihoods for sequence and expression profile data. In *Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 202–210, 2000.

[HBKS00] S Hampson, P Baldi, D Kibler, and S Sandmeyer. Analysis of yeast's ORF upstream regions by parallel processing, microarrays and computational methods. In *Intelligent Systems for Molecular Biology (ISMB)*, pages 190–201, 2000.

[Hec95] David Heckerman. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, 1995.

[HGC95] David Heckerman, Dan Geiger, and David M Chickering. Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[HGH+02] Yuen Ho, Albrecht Gruhler, Adrian Heilbut, Gary D Bader, Lynda Moore, Sally-Lin Adams, Anna Millar, Paul Taylor, Keiryn Bennett, Kelly Boutilier, Lingyun Yang, Cheryl Wolting, Ian Donaldson, Soren Schandorff, Juanita Shewnarane, Mai Vo, Joanne

Taggart, Marilyn Goudreault, Brenda Muskat, Cris Alfarano, Danielle Dewar, Zhen Lin, Katerina Michalickova, Andrew R Willems, Holly Sassi, Peter A Nielsen, Karina J Rasmussen, Jens R Andersen, Lene E Johansen, Lykke H Hansen, Hans Jespersen, Alexandre Podtelejnikov, Eva Nielsen, Janne Crawford, Vibeke Poulsen, Birgitte D Sorensen, Jesper Matthiesen, Ronald C Hendrickson, Frank Gleeson, Tony Pawson, Michael F Moran, Daniel Durocher, Matthias Mann, Christopher W V Hogue, Daniel Figeys, and Mike Tyers. Systematic identification of protein complexes in Saccharomyces cerevisiae by mass spectrometry. *Nature*, 415:180–183, 2002.

[HGJY01] Alexander J Hartemink, David K Gifford, Tommi S Jaakkola, and Richard A Young. Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In *Pacific Symposium on Biocomputing (PSB)*, 2001.

[HGJY02] Alexander J Hartemink, David K Gifford, Tommi S Jaakkola, and Richard A Young. Combining location and expression data for principled discovery of genetic regulatory network models. In *Pacific Symposium on Biocomputing (PSB)*, 2002.

[HMA02] Dongqing Huang, Jason Moffat, and Brenda Andrews. Dissection of a complex phenotype by functional genomics reveals roles for the yeast cyclin-dependent protein kinase Pho85 in stress adaptation and cell integrity. *Mol Cell Biol*, 22(14):5076–5088, 2002.

[HMJ+00] T Hughes, M Marton, AR Jones, C Roberts, R Stoughton, C Armour, H Bennett, E Coffey, H Dai, Y He, MJ Kidd, and AM King. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–126, 2000.

[HML+04] Henning Hermjakob, Luisa Montecchi-Palazzi, Chris Lewington, Sugath Mudali, Samuel Kerrien, Sandra Orchard, Martin Vingron, Bernd Roechert, Peter Roepstorff, Alfonso Valencia, Hanah Margalit, John Armstrong, Amos Bairoch, Gianni Cesareni, David Sherman, and Rolf Apweiler. IntAct: an open source molecular interaction database. *Nucleic Acids Research*, 32:D452–D455, 2004.

[Hof93] K Hoffgen. Learning and robust learning of product distributions. In *Workshop on Computational Learning Theory (COLT)*, pages 77–83, 1993.

[HSL+99] E Hartuv, A Schmitt, J Lange, S Meier-Ewert, H Lehrachs, and R Shamir. An algorithm for clustering cDNAs for gene expression analysis. In *RECOMB*, pages 188–197, 1999.

[HSL+04] N Hulo, CJA Sigrist, V Le Saux, PS Langendijk-Genevaux, L Bordoli, A Gattiker, E De Castro, P Bucher, and A Bairoch. Recent improvements to the PROSITE database. *Nucleic Acids Research*, 32:134–137, 2004.

[HSvMB03] Martijn A Huynen, Berend Snell, Christian von Mering, and Peer Bork. Function prediction and protein networks. *Current Opinion in Cell Biology*, 15:191–198, 2003.

[HZZL02] Daniel Hanisch, Alexander Zien, Ralf Zimmer, and Thomas Lengauer. Co-clustering of biological networks and gene expression data. *Bioinformatics*, 18:S145–S154, 2002.

[IER+99] Vishwanath R Iyer, Michael B Eisen, Douglas T Ross, Greg Schuler, Troy Moore, Jeffrey CF Lee, Jeffrey M Trent, Louis M Staudt, James Hudson Jr, Mark S Boguski, Deval Lashkari, Dari Shalon, David Botstein, and Patrick O Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283:83–87, 1999.

[IGM02] S Imoto, T Goto, and S Miyano. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. In *Pacific Symposium on Biocomputing (PSB)*, volume 7, pages 175–186, 2002.

[ITR+01] Trey Ideker, Vesteinn Thorsson, Jeffrey A Ranish, Rowan Christmas, Jeremy Buhler, Jimmy K Eng, Roger Bumgarner, David R Goodlett, Ruedi Aebersold, and Leroy Hood. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 292:929–933, 2001.

[JC97] JJ Jiang and DW Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *International Conference on Research in Computational Linguistics (ROCLING)*, pages 19–33, 1997.

[Jef61] Harold Jeffreys. *Theory of Probability*. Oxford University Press, 1961.

[JEMF05] Ariel Jaimovich, Gal Elidan, Hanah Margalit, and Nir Friedman. Towards an integrated protein-protein interaction network. In *Lecture Notes in Computer Science*, volume 3500, pages 14–30, 2005.

[JFS95] Charles E Jacobs, Adam Finkelstein, and David H Salesin. Fast multiresolution image querying. *Computer Graphics*, 29(Annual Conference Series):277–286, 1995.

[JJLL02] Gary Johnson, Robert Jotte, Razvan Lapadat, and Sonia Leach. Human adenocarcinoma data. Unpublished data, contact first author for availability, 2002.

[JOB03] H Jeong, Z N Oltvai, and A L Barabási. Prediction of protein essentiality based on genomic data. *ComplexUs*, 1:19–28, 2003.

[JYG+03] R Jansen, H Yu, D Greenbaum, Y Kluger, NJ Krogan, S Chung, A Emili, M Snyder, JF Greenblatt, and M Gerstein. A Bayesian networks approach for predicting protein-protein interactions from genomic data. *Science*, 302(5644):449–53, 2003.

[KAH+02] Anuj Kumar, Seema Agarwal, John A Heyman, Sandra Matson, Matthew Heidtman, Stacy Piccirillo, Lara Umansky, Amar Drawid, Ronald Jansen, Yang Liu, Kei-Hoi Cheung, Perry Miller, Mark Gerstein, G Shirleen Roeder, and Michael Snyder. Subcellular localization of the yeast proteome. *Genes and Development*, 16(6):707–719, 2002.

[KGH+06] Minoru Kanehisa, Susumu Goto, Masahiro Hattori, Kiyoko F Aoki-Kinoshita, Masumi Itoh, Shuichi Kawashima, Toshiaki Katayama, Michihiro Araki, , and Mika Hirakawa. From genomics to chemical genomics: new developments in KEGG. *Nucleic Acids Research*, 34:D354–D357, 2006.

[KKK97] Shuichi Kawashima, Toshiaki Katayama, and Minoru Kanehisa. Construction of molecular interaction database and searching for similar pathways. *Genome Informatics*, 8:298–299, 1997.

[Kru94] R Krumlauf. Analysis of gene expression by northern blot. *Mol Biotechnol*, 2(3):227–42, 1994.

[KS96] Ron Kohavi and Mehran Sahami. Error-based and entropy-based discretization of continuous features. In *Knowledge Discovery and Data Mining (KDD)*, pages 114–119, 1996.

[KSL+99] Greg A Keim, Noam M Shazeer, Michael L Littman, Sushant Agarwal, Catherine M Cheves, Joseph Fitzgerald, Jason Grosland, Fan Jiang, Shannon Pollard, and Karl Weinmeister. PROVERB: The probabilistic cruciverbalist. In *AAAI/IAAI*, pages 710–717, 1999.

[KT06] Jin-Dong Kim and Jun'ichi Tsujii. Corpora and their annotation. In Sophia Ananiadou and John McNaught, editors, *Text Mining for Biology and Biomedicine*, pages 179–211. Artech House, London, 2006.

[KvMB03] R Krause, C von Mering, and P Bork. A comprehensive set of protein complexes in yeast: mining large scale protein-protein interaction screens. *Bioinformatics*, 19(15):1901–8, 2003.

[Lau92] S. Lauritzen. Propagation of probabilities, means, and variances in mixed graphical association models. *Journal of the American Statistical Association*, 87(420):1098–1108, 1992.

[LBG80] Y Linde, A Buzo, and RM Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28:84–95, 1980.

[LBU04] Phillip P Le, Amit Bahl, and Lyle H Ungar. Using prior knowledge to improve genetic network reconstruction from microarray data. *In Silico Biology*, 4(3):335–353, 2004.

[LCC+05] CY Lin, CL Chen, CS Cho, LM Wang, CM Chang, PY Chen, CZ Lo, and CA Hsiung. hp-DPI: Helicobacter pylori database of protein interactomes–embracing experimental and inferred interactions. *Bioinformatics*, 21(7):1288–90, 2005. Erratum in: Bioinformatics 2005 Jun 15;21(12):2932.

[LDB+96] David J Lockhart, Helin Dong, Michael C Byrne, Maximillian T Follettie, Michael V Gallo, Mark S Chee, Michael Mittmann, Chunwei Wang, Michiko Kobayashi, Heidi Horton, and Eugene L Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 14:1675–1680, 1996.

[LFS98] S Liang, S Fuhrman, and R Somogyi. REVEAL: A general reverse engineering algorithm for inference of geneticnetwork architectures. In *Pacific Symposium on Biocomputing (PSB)*, volume 3, pages 18–29, 1998.

[Lin98] D Lin. An information-theoretic definition of similarity. In *International Conference on Machine Learning (ICML)*, San Francisco, CA, 1998. Morgan Kaufmann.

[LK03] S Letovsky and S Kasif. Predicting protein function from protein/protein interaction data: a probabilistic approach. *Bioinformatics*, 19 Suppl 1:I197–I204, 2003.

[LL03] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, pages 448–455, 2003.

[LLBB01] Jason D Lieb, Xiaole Liu, David Botstein, and Patrick O Brown. Promoter-specific binding of Rap1 revealed by genome-wide maps of protein-DNA association. *Nature Genetics*, 28:327–334, 2001. Erratum in Nat Genet 2001 Sep;29(1):100.

[LLYL02] Bing Liu, Wee Sun Lee, Philip S Yu, and Xiaoli Li. Partially supervised classification of text documents. In *ICML*, pages 387–394, 2002.

[LM98] Huan Liu and Hiroshi Motoda. *Feature selection for knowledge discovery and data mining*. Kluwer Academic Publishers, 1998.

[LPIS00] GJ Lord, E Pardo-Igúzquiza, and IM Smith. A practical guide to wavelets for metrology. Technical report, NPL Report CMSC 02/00, 2000.

[LRR+02] Tong Ihn Lee, Nicola J Rinaldi, François Robert, Duncan T Odom, Ziv Bar-Joseph, Georg K Gerber, Nancy M Hannett, Christopher T Harbison, Craig M Thompson, Itamar Simon, Julia Zeitlinger, Ezra G Jennings, Heather L Murray, D Benjamin Gordon, Bing Ren, John J Wyrick, Jean-Bosco Tagne, Thomas L Volkert, Ernest Fraenkel, David K Gifford, and Richard A Young. Transcriptional regulatory networks in Saccharomyces cerevisiae. *Science*, 298(5594):799–804, 2002.

[LSBG03] PW Lord, RD Stevens, A Brass, and CA Goble. Semantic similarity measures as tools for exploring the Gene Ontology. In *Pacific Symposium on Biocomputing (PSB)*, pages 601–612, 2003.

[LW00] Marcus-Christopher Ludl and Gerhard Widmer. Relative unsupervised discretization for association rule mining. In *PKDD 00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 148–158, 2000.

[MC01] Pedrito Maynard–Reid II and Urszula Chajewska. Aggregating learned probabilistic beliefs. In *UAI*, pages 354–361, 2001.

[MCM00] EJ Moler, ML Chow, and IS Mian. Analysis of molecular profile data using generative and discriminative models. *Physiol Genomics*, 4:109–126, 2000.

[MFG+02] HW Mewes, D Frishman, U Guldener, G Mannhaupt, K Mayer, M Mokrejs, B Morgenstern, M Munsterkotter, S Rudd, and B Weil. MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, 30:31–34, 2002.

[MGR95] David Madigan, Jonathan Gavrin, and Adrian E Raftery. Eliciting prior information to enhance the predictive performance of Bayesian graphical models. *Communications in Statistics: Theory and Methods*, 24:2271–2292, 1995.

[MNB+04] N Majewski, V Nogueira, P Bhaskar, P Coy, J Skeen, K Gottlob, N Chandel, C Thompson, R Robey, and N Hay. Hexokinase-mitochondria interaction mediated by Akt is required to inhibit apoptosis in the presence or absence of Bax and Bak. *Molecular Cell*, 16(5):819–830, 2004.

[MPT+99] Edward M Marcotte, Matteo Pellegrini, Michael J Thompson, Todd O Yeates, and David Eisenberg. A combined algorithm for genome-wide prediction of protein function. *Nature*, 402:83–86, 1999.

[MTO+01] Y Maki, D Tominaga, M Okamoto, S Watanabe, and Y Eguchi. Development of a system for the inference of large scale genetic networks. In *Pacific Symposium on Biocomputing (PSB)*, volume 6, pages 446–458, 2001.

[Mur01] Kevin Murphy. Learning bayes net structure from sparse data sets. Technical report, Comp Sci Div, UC Berkeley, 2001.

[MVR+01] Lisa R Matthews, Philippe Vaglio, Jérôme Reboul, Hui Ge, Brian P Davis, James Garrels, Sylvie Vincent, and Marc Vidal. Identification of potential interaction networks using sequence-based searches for conserved protein-protein interactions or interologs. *Genome Research*, 11(12):2120–6, 2001.

[MWJ99] Kevin P Murphy, Yair Weiss, and Michael I Jordan. Loopy belief propagation for approximate inference: an empirical study. In *UAI*, pages 467–475, 1999.

[MYC+02] Joseph C Mellor, Itai Yanai, Karl H Clodfelter, Julian Mintseris, and Charles DeLisi. Predictome: a database of putative functional links between proteins. *Nucleic Acids Research*, 30(1):306–309, 2002.

[NAY+00] MT Nevalainen, TJ Ahonen, H Yamashita, V Chandrashekar, A Bartke, PM Grimley, GW Robinson, L Hennighausen, and H Rui. Epithelial defect in prostates of stat5a-null mice. *Lab Invest*, 80(7):993–1006, 2000.

[NJA+05] Elena Nabieva, Kam Jim, Amit Agarwal, Bernard Chazelle, and Mona Singh. Whole-genome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics*, 21(Suppl.1):i302–i310, 2005.

[NWK00] John RS Newman, Ethan Wolf, and Peter S Kim. A computationally directed screen identifying interacting coiled coils from Saccharomyces cerevisiae. *Proceedings of the National Academy of Sciences USA*, 97(24):13203–13208, 2000.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[PGK05] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105, 2005.

[PKK+05] Sungman Park, Donghwa Kim, Satoshi Kaneko, Kristen M Szewczyk, Santo V Nicosia, Hua Yu, Richard Jove, and Jin Q Cheng. Molecular cloning and characterization of the human AKT1 promoter uncovers its up-regulation by the Src/Stat3 pathway. *J Biol Chem*, 280(47):38932–38941, 2005.

[PNA+03] Suraj Peri, J Daniel Navarro, Ramars Amanchy, Troels Z Kristiansen, Chandra Kiran Jonnalagadda, Vineeth Surendranath, Vidya Niranjan, Babylakshmi Muthusamy, TKB Gandhi, Mads Gronborg, Nieves Ibarrola, Nandan Deshpande, K Shanker, HN Shivashankar, BP Rashmi, MA Ramya, Zhixing Zhao, KN Chandrika, N Padma, HC Harsha, AJ Yatish, MP Kavitha, Minal Menezes, Dipanwita Roy Choudhury, Shubha Suresh, Neelanjana Ghosh, R Saravana, Sreenath Chandran, Subhalakshmi Krishna, Mary Joy, Sanjeev K Anand, V Madavan, Ansamma Joseph, Guang W Wong, William P Schiemann, Lily Huang Stefan N Constantinescu, Roya Khosravi-Far, Hanno Steen, Muneesh Tewari, Saghi Ghaffari, Gerard C Blobe, Chi V Dang, Joe GN Garcia, Jonathan Pevsner, Ole N Jensen, Peter Roepstorff, Krishna S Deshpande, Arul M Chinnaiyan, Ada Hamosh, Aravinda Chakravarti, and Akhilesh Pandey. Development of Human Protein Reference Database as an initial platform for approaching systems biology in humans. *Genome Research*, 13:2363–2371, 2003.

[PREF01] Dana Pe'er, Aviv Regev, Gal Elidan, and Nir Friedman. Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 17 Suppl.1:S215–S224, 2001.

[PW99] David M Pennock and Michael P Wellman. Graphical representations of consensus belief. In *UAI*, pages 531–540, 1999.

[PWJ04] N Przulj, DA Wigle, and I Jurisica. Functional topology in a network of protein interactions. *Bioinformatics*, 20(3):340–8, 2004.

[Qui93] J Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[Res95] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *International Joint Conference on AI IJCAI*, pages 448–453, 1995.

[Rig02] Carsten Riggelsen. Induction of Bayesian networks with a priori domain knowledge. Master's thesis, Utrecht University, 2002.

[RMH+03] Michael C Rudolph, James L McManaman, Larry Hunter, Tzulip Phang, and Margaret C Neville. Functional development of the mammary gland: Use of expression profiling and trajectory clustering to reveal changes in gene expression during pregnancy, lactation, and involution. *Journal of Mammary Gland Biology and Neoplasia*, 8(3):287–307, 2003.

[RMP+] Michael C Rudolph, James L McManaman, Tzulip Phang, Steven M Anderson, Tanya Russell, Douglas Kominsky, Natalie Serkova, and Margaret C Neville. Metabolic regulation in a lipid synthesizing machine: The lactating mouse. *Physiol Genomics*, to appear.

[RRS96] S Rabaseda, R Rakotomalala, and M Sebban. A comparison of some contextual discretization methods. *Information Sciences*, 92:137–157, 1996.

[RSL+00] S Raychaudhuri, JM Stuart, X Liu, PM Small, and RB Altman. Pattern recognition of genomic features with microarrays: site typing of *mycobacterium tuberculosis* strains. In *Intelligent Systems for Molecular Biology (ISMB)*, pages 286–295, 2000.

[RTS95] MC Riedy, EA Timm Jr, and CC Stewart. Quantitative RT-PCR for measuring gene expression. *Biotechniques*, 18(1):70–4,76, 1995.

[SA96] R Srikant and R Agrawal. Mining quantitative association rules in large relational tables. In *ACM SIGMOD International Conference on Management of Data*, pages 1–12, 1996.

[SBR+06] C Stark, BJ Breitkreutz, T Reguly, L Boucher, A Breitkreutz, and M Tyers. BioGRID: a general repository for interaction datasets. *Nucleic Acids Research*, 34:D535–9, 2006.

[SDKZ02] I Shmulevich, ER Dougherty, S Kim, and W Zhang. Probabilistic boolean network: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, 18:261–74, 2002.

[SDLC93] DJ Spiegelhalter, AP Dawid, SL Laurtizen, and RG Cowell. Bayesian analysis in expert systems. *Statistical Science*, 8:219–283, 1993.

[SDS95] Eric J Stollnitz, Tony D DeRose, and David H Salensin. Wavelets for computer graphics: A primer part 2. *IEEE Computer Graphics and Applications*, 15(4):75–85, 1995.

[SFK+01] Harukazu Suzuki, Yoshifumi Fukunishi, Ikuko Kagawa, Rintaro Saito, Hiroshi Oda, Toshinori Endo, Shinji Kondo, Hidemasa Bono, Yasushi Okazaki, and Yoshihide Hayashizaki. Protein-protein interaction panel using mouse full-length cDNAs. *Genome Research*, 11(10):1758–1765, 2001.

[SIK+05] Roded Sharan, Trey Ideker, Brian P Kelley, Ron Shamir, and Richard M Karp. Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. *Journal of Computational Biology*, 12(6):835–846, 2005.

[SKS05] Amos Tanay Israel Steinfeld, Martin Kupiec, and Ron Shamir. Integrative analysis of genome-wide experiments in the context of a large high-throughput data compendium. *Molecular Systems Biology*, 2005.

[SL03] MP Samanta and S Liang. Predicting protein functions from redundancies in large-scale protein interaction networks. *Proc Natl Acad Sci U S A*, 100(22):12579–83, 2003.

[SLP02] J Shrager, P Langley, and A Pohorille. Guiding revision of regulatory models with expression data. In *Pacific Symposium on Biocomputing (PSB)*, pages 486–497, 2002.

[SMO+03] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13:2498–2504, 2003.

[SPR+03] Thomas Schlitt, Kimmo Palin, Johan Rung, Sabine Dietmann, Michael Lappe, Esko Ukkonen, and Alvis Brazma. From gene networks to gene function. *Genome Research*, 13:2568–2576, 2003.

[SS05] E Segal and R Sharan. A discriminative model for identifying spatial cis-regulatory modules. *J Comput Biol*, 12(6):822–34, 2005.

[SSH+96] M Schena, D Shalon, R Heller, A Chai, PO Brown, and RW Davis. Parallel human genome analysis: Microarray-based expression monitoring of 1000 genes. *Proceedings of the National Academy of Sciences USA*, 93:10614–10619, 1996.

[SSH03] R Saito, H Suzuki, and Y Hayashizaki. Construction of reliable protein-protein interaction networks with a new interaction generality measure. *Bioinformatics*, 19(6):756–63, 2003.

[SSZ+98] Paul T Spellman, Gavin Sherlock, Michael O Zhang, Vishwanath R Iyer, Kirk Anders, Michael B Eisen, Patrick O Brown, David Botstein, and Bruce Futcher. Comprehensive identification of the cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9:3273–3297, 1998.

[STG+01] Eran Segal, Ben Taskar, Audrey Gasch, Nir Friedman, and Daphne Koller. Rich probabilistic models for gene expression. *Bioinformatics*, 17 Suppl.1:S1–S10, 2001.

[Str89] Kevin Struhl. Molecular mechanisms of transcriptional regulation in yeast. *Annual Review of Biochemistry*, 58:1051–1077, 1989.

[TDO+03] OG Troyanskaya, K Dolinski, AB Owen, RB Altman, and D Botstein. A Bayesian framework for combining heterogeneous data sources for gene function prediction (in Saccharomyces cerevisiae). *Proc Natl Acad Sci U S A*, 100(14):8348–53, 2003.

[TKB+03] Yoshinori Tamada, SunYong Kim, Hideo Bannai, Seiya Imoto, Kousuke Tashiro, Satoru Kuhara, and Satoru Miyano. Estimating gene networks from gene expression data by combining Bayesian network model with promoter element detection. *Bioinformatics*, 19:ii227–ii236, 2003.

[TS01] Amos Tanay and Ron Shamir. Computational expansion of genetic networks. *Bioinformatics*, 17 Suppl.1:S270–S278, 2001.

[TT98] D Thieffry and R Thomas. Qualitative analysis of gene networks. In *Pacific Symposium on Biocomputing (PSB)*, volume 3, pages 77–88, 1998.

[VBJ+00] J Vilo, A Brazma, I Jonassen, A Robinson, and E Ukkonen. Mining for putative regulatory elements in the yeast genome using gene expression data. In *Intelligent Systems for Molecular Biology (ISMB)*, pages 384–394, 2000.

[VFMV03] A Vazquez, A Flammini, A Maritan, and A Vespignani. Global protein function prediction from protein-protein interaction networks. *Nat Biotechnol*, 21(6):697–700, 2003.

[vMHJ+03] Christian von Mering, Martijn Huynen, Daniel Jaeggi, Steffen Schmidt, Peer Bork, and Berend Snel. STRING: a database of predicted functional associations between proteins. *Nucleic Acids Research*, 31(1):258–261, 2003.

[vMJS+05] Christian von Mering, Lars J Jensen, Berend Snel, Sean D Hooper, Markus Krupp, Mathilde Foglierini, Nelly Jouffre, Martijn A Huynen, and Peer Bork. STRING: known and predicted protein-protein associations, integrated and transferred across organisms. *Nucleic Acids Research*, 33:D433–D437, 2005.

[vMKS+02] C von Mering, R Krause, B Snel, M Cornell, SG Oliver, S Fields, and P Bork. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417(6887):399–403, 2002.

[vSWR00] EP van Someren, LFA Wessels, and MJT Reinders. Linear modeling of genetic networks from experimental data. In *Intelligent Systems for Molecular Biology*, pages 355–366, 2000.

[VZVK95] VE Velculescu, L Zhang, B Vogelstein, and KW Kinzler. Serial analysis of gene expression. *Science*, 270(5235):484–487, 1995.

[WABD04] Haiying Wang, Francisco Azuaje, Olivier Bodenreider, and Joaquín Dopazo. Gene expression correlation and gene ontology-based similarity: An assessment of quantitative relationships. In *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 25–31, 2004.

[War63] Joe H Ward. Hierarchical grouping to optimize an objective function. *Journal of American Statistical Association*, 58(301):236–244, 1963.

[WCH+00] E Wingender, X Chen, R Hehl, H Karas H, I Liebich, V Matys, T Meinhardt, M Pruss, I Reuter, and F Schacherer. TRANSFAC: an integrated system for gene expression regulation. *Nucleic Acids Research*, 28:316–319, 2000.

[WFM+98] Xiling Wen, Stefanie Fuhrman, George S Michaels, Daniel B Carr, S Smith, JL Barker, and Roland Somogyi. Large-scale temporal gene expression mapping of central nervous system development. *Proceedings of the National Academy of Sciences*, 95(1):334–339, 1998.

[WGR+02] Hannes Wettig, Peter Grünwald, Teemu Roos, Petri Myllymäki, and Henry Tirri. Supervised naive bayes parameters. In *Intelligence, the art of natural and artificial and artificial: Proceedings of the 10th Finnish Artificial Intelligence Conference*, pages 72–83, 2002.

[WKB05] Cecily J Wolfe, Isaac S Kohane, and Atul J Butte. Systematic survey reveals general applicability of guilt-by-association within gene coexpression networks. *BMC Bioinformatics*, 6:227, 2005.

[WLW+02] Jamie L Wooldridge, Sonia M Leach, Kristin A Wallick, Bifeng Gao, Mark W Geraci, and Frank J Accurso. Microarray determination of gene expression in cystic fibrosis and non-cystic fibrosis whole lung. Unpublished, available from last author, 2002.

[WM97] SC Weller and NC Mann. Assessing rater performace without a gold standard using consensus theory. *Medical Decision Making*, 17(1):71–79, 1997.

[XSD+02] I Xenarios, L Salwinski, XJ Duan, P Higney, S Kim, and D Eisenberg. DIP: The Database of Interacting Proteins. a research tool for studying cellular networks of protein interactions. *Nucleic Acids Research*, 30:303–305, 2002.

[YBW+03] Gaël Yvert, Rachel B Brem, Jacqueline Whittle, Joshua M Akey, Eric Foss, Erin N Smith, Rachel Mackelprang, and Leonid Kruglyak. Trans-acting regulatory variation in Saccharomyces cerevisiae and the role of transcription factors. *Nature Genetics*, 35:57–64, 2003.

[YDD01] Itai Yanai, Adnan Derti, and Charles DeLisi. Genes linked by fusion events are generally of the same functional category: A systematic analysis of 30 microbial genomes. *Proceedings of the National Academy of Sciences USA*, 98(14):7940–7945, 2001.

[YMD02] Itai Yanai, Joseph C Mellor, and Charles DeLisi. Identifying functional links between genes using conserved chromosomal proximity. *Trends in Genetics*, 18(4):176–179, 2002.

[YTC02] C Yoo, V Thorsson, and GF Cooper. Discovery of causal relationships in a gene regulation pathway from a mixture of the experimental and observational DNA microarray data. In *Pacific Symposium on Biocomputing (PSB)*, 2002.

[ZGSD03] Daniel E Zak, Gregory E Gonye, James S Schwaber, and Francis J Doyle III. Importance of input perturbations and stochastic gene expression in the reverse engineering of genetic regulatory networks: Insights from an identifiability analysis of an in silico network. *Genome Research*, 13:2396–2405, 2003.

[ZKZL00] A Zien, R Küffner, R Zimmer, and T Lengauer. Analysis of gene expression data with pathway scores. In *Intelligent Systems for Molecular Biology*, pages 407–417, 2000.

[ZMQ+02] A Zanzoni, L Montecchi-Palazzi, M Quondam, G Ausiello, M Helmer-Citterich, and G Cesareni. MINT: a Molecular INTeraction database. *FEBS Letters*, 513(1):135–140, 2002.