A Computational Framework for the Analysis of Multi-Species Microarray Data

Yong Lu

CMU-CS-08-155
September 2008

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Ziv Bar-Joseph, Co-Chair
Roni Rosenfeld, Co-Chair
Eric Xing
Gerard J. Nau, University of Pittsburgh
Jerry Xiaojin Zhu, University of Wisconsin, Madison

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was sponsored by the National Science Foundation under grants CAREER-0448453 and ITR-0225656, National Institute of Health under grant NO1 AI-5001, and the Pennsylvania Department of Health. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. government.

Keywords: Cell Cycle, Immune Response, Microarray, Markov Random Field

Abstract

In this thesis I present algorithms for the analysis of microarray expression data from multiple species. These algorithms are used to identify core genes in two biological systems, the cell cycle and the immune response.

With data generated from high throughput biological experiments, it is now becoming possible to study organisms at the systems level. One of the first questions facing researchers is the identification of the core components of biological subsystems within an organism. This task is made difficult by the high levels of experimental and biological noise associated with these experiments. To address these problems I introduce a new computational framework for combining data from multiple species, for both improving prediction accuracy and identifying important subsets of genes involved in a given system. The computational framework is based on Markov random fields which allow the integration of microarray and sequence data from multiple species. Applying this framework to study cell cycle regulated genes, I have identified genes representing the core machinery of the cell cycle. These findings are supported by both complementary high-throughput data and motif analysis. In addition, I apply this computational framework to study immune response in human and mouse. I show that by using Gaussian random fields instead of discrete Markov random fields we are able to achieve better accuracy in predicting immune response genes. Finally, we identify a list of immune response genes that are conserved between cell types and species for further experimental study.

To Tao and my parents

Acknowledgments

First of all, I am deeply grateful to my advisors, Profs. Ziv Bar-Joseph and Roni Rosenfeld, for their thoughtful guidance during my five years at Carnegie Mellon. Roni skillfully introduced me to the exciting area of computational biology. Ziv taught me a great deal about how to tackle large complex problems. Both of them are incredibly supportive and have such an immensely positive impact on my life.

I would like to thank my committee members, Profs. Eric Xing, Jerry Nau, and Jerry Zhu, for carefully reading my thesis and giving me invaluable feedbacks. I also benefited much from discussions with Drs. Itamar Simon, Takis Benos, and Shawn Mahony.

I want to thank the (former) members of the Systems Biology Group: Jason Ernst, Yanjun Qi, Yanxin Shi, Henry Lin, Tony Gitter, Guy Zinman, and Peter Huggins. It is always refreshing to learn from and exchange ideas with them. Not to mention it was fun to hang out together when we went to a conference.

Carnegie Mellon is an exciting place for machine learning research. In addition to learning from Profs. Zoubin Ghahramani, Carlos Guestrin, John Lafferty, Tom Mitchell, Andrew Moore, Larry Wasserman, and Eric Xing in class, I also benefited from discussions with Steve Hanneke, Yan Liu, Luo Si, Kyung-Ah Sohn, Rong Yan, Jian Zhang, Yi Zhang, and many other fellow graduate students.

From time to time, I asked questions on Zephyr, a wonderful place you can get valuable suggestions on everything from fixing a broken AFS connection to finding a local restaurant. I am especially grateful to the gurus on Zephyr, Jeffrey Baird, Darrell Kindred, Ray Link, Rajesh Balan, Francisco Pereira, and many others, for their help.

Badminton has been a great pastime for me at Carnegie Mellon, and I get to know many friends on the badminton court. I especially enjoyed playing with (and against) Chen-Ling Chou, Hung-Chih Lai, Gaofei He, Yizhang Yang, and Ning Yao, who share my passion for this fascinating sport. It was often the most reinvigorating and relaxing moment of the week when we played in the Skibo Gym.

I am fortunate to have the opportunity to interact with other great minds across the campus. I am continuously impressed by Prof. Ajay Kalra's insightful remarks

on human behavior. Prof. Baohong Sun keeps me amazed by her ability to take care of two young kids while staying productive in research. I would also like to thank Jian Ni and Meng Zhu at the Tepper School for their friendship.

I am indebted to my parents, Junbao Lu and Minhua Jin, for raising me and their support during my stay at Carnegie Mellon. My interest in computational biology is at least partly due to hearing their experience as a forensic pathologist and a pharmacist.

Finally, I would like to thank my wonderful wife, Tao. Without her, this would not be possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

1.1.1 Systems Biology and Conservation in Biological Systems

With data generated from high throughput biological experiments, it is now becoming possible to study organisms at the systems level. Systems biology studies the dynamics and interactions between the components of a biological system. One of the first questions facing researchers is to identify the components, or biological subsystems, within an organism.

One important approach for this task is to identify conserved genes between species related at suitable evolutionary distance. Under evolutionary selection pressure, proteins essential for survival are more likely to stay the same or similar to their ancestors. In fact, comparative studies of eukaryotic species have revealed conservation at multiple levels, despite the fact that they were separated by speciation events millions of years ago [Holm and Sander, 1996, Hardie et al., 1998]. For example, the control mechanism regulating the onset of mitosis is common to all eukaryote cells [Nurse, 1990].

Proteins and RNAs (including miRNA, siRNA, etc) are the two major workhorses in the cell. Our discussion will focus on proteins, but in principle it is also applicable to RNAs. The function of a protein is determined by its structure, which is in turn determined by its amino acid or nucleic acid sequence. Currently, it is much more expensive and time-consuming to determine the structure than the sequence of a new protein, and it is very hard to computationally predict structures based on sequences. Therefore, when researchers want to study the functions of unknown proteins, they often have to rely on sequences directly.

Functional annotation of proteins can serve as an example. When people find a new protein and want to determine its function, they can look for known proteins

with similar sequences and hypothesize that they have the same function [Bork et al., 1998, Wilson et al., 2000]. In practice, sequence similarity is usually a fairly good indicator of structural and functional conservation, but there are also counterexamples where proteins with similar structures have low sequence similarity [Holm and Sander, 1996].

In addition to sequences that encode proteins, non-coding sequences are also found to be conserved between some species [Duret et al., 1993, Dubchak et al., 2000, Kellis et al., 2003, Xie et al., 2005]. Parts of the non-coding regions, e.g. cis-regulatory sequences, are believed to play an important role in transcriptional regulation, so the conservation of non-coding sequences may imply the conservation of interaction networks between species.

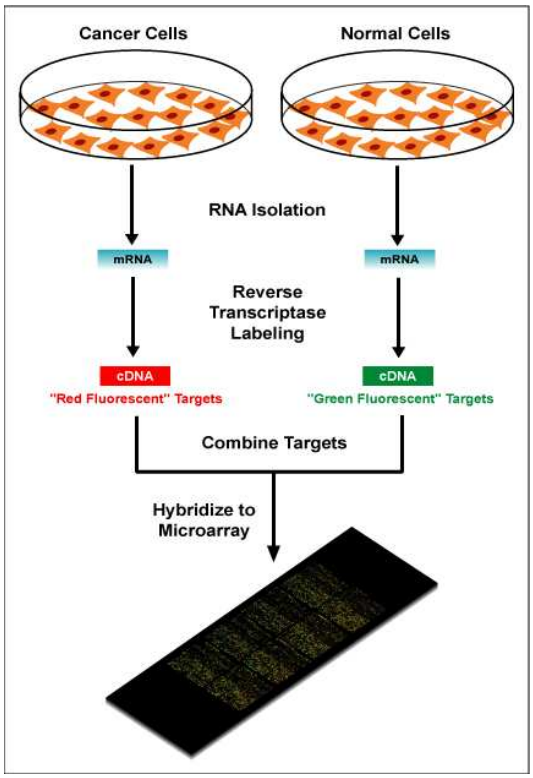
1.1.2 DNA Microarrays

A DNA microarray is an array of short single stranded DNA segments ("probes") printed densely on a solid surface, e.g. glass or plastic [Schena et al., 1995]. It can be used to analyze the gene expression profiles for thousands of genes simultaneously. Due to its small format and high density, a few microliters are enough for detection of target genes [Schena et al., 1995].

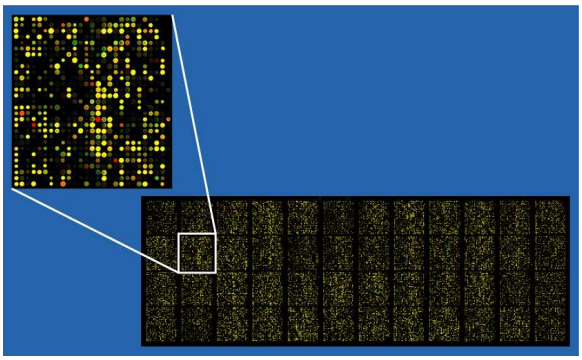
Microarray technology is based on the complementarity property of DNA and RNA molecules. In short, DNA or RNA sequences are made up of four types of bases, and two of them are complementary to the other two, i.e. they are able to match and bind to each other. Therefore it is possible to identify a target gene transcript using a "probe" sequence complementary to the target. By attaching thousands of well-designed probe sequences on a microarray and matching them to the gene transcripts in a sample, we are able to identify all the genes expressed in the sample.

To make microarrays work, we need a way to quantify how much DNA is bound to each probe. This is done by tagging all the DNA in the sample by fluorescent material. After hybridization, a process to match sample DNA with the probes, unmatched DNA is removed and the matched DNA can be quantified by measuring the intensity of fluorescence.

There are two major types of microarrays, one is spotted microarrays [Schena et al., 1995], and the other is oligonucleotide microarrays [Lockhart et al., 1996]. In spotted microarrays, two samples to be compared are labeled with two different fluorophores. They are mixed and hybridized to a single microarray and the ratio of fluorescent intensity is used to quantify the change of gene expression levels. It is easy to observe up- or down-regulation using spotted microarrays, but it's hard to know the absolute levels. The second type of microarrays usually requires only one DNA sample and is calibrated by control probes on the microarrays. Therefore



(a)



(b)

Figure 1.1: (a): A schematic diagram for spotted microarrays. (b): a scanned microarray image. (Source: Wikipedia).

multiple microarrays are needed for cross sample comparison, but it is possible to observe absolute expression values.

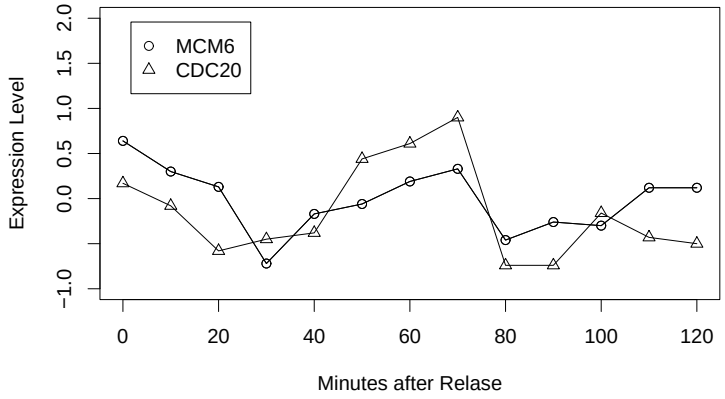
1.1.3 Potential of Data Integration

Following the completion of genome sequencing projects for multiple model species [Goffeau et al., 1996, Arabidopsis Genome Initiative, 2000, Adams et al., 2000, Venter et al., 2001, Waterston et al., 2002, Stein et al., 2003], it is possible to carry out comparative studies at the whole genome scale [Ureta-Vidal et al., 2003]. The major principle of comparative genomics is as follows: common traits of two organisms are the result of functionally conserved proteins or RNAs, which are encoded by DNA sequences conserved between the species since their last common ancestor. Conversely, proteins and RNAs responsible for species specific traits are encoded by divergent DNA sequences [Hardison, 2003].

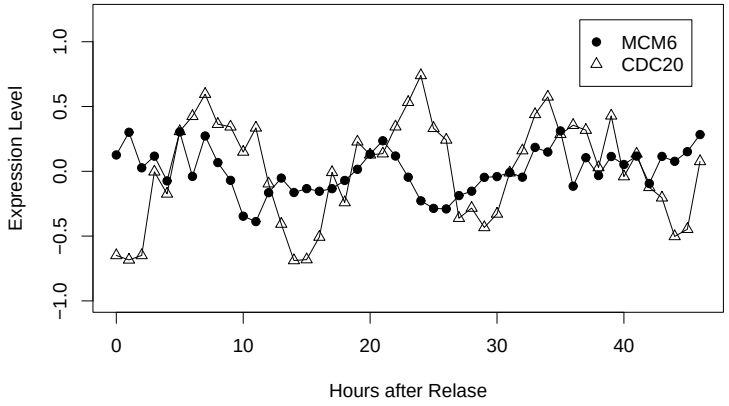
Comparative genomics has shown to be a very promising field and sequence comparison has become a standard tool when looking for homology between genes. In many applications, the BLAST algorithm [Altschul et al., 1990, 1997] is used to search for sequence similarity, and a cut-off score is used to determine the homology relations.

However, there is an inherent limitation to sequence-based methods. Sequences only provide static information about the organisms, while in many cases what we want to understand is the dynamic aspects of the proteins. We argue that it is beneficial to integrate static information with data measuring dynamic properties of the biological system. For example, gene expression microarrays are able to measure the expression level of all genes in a cell at a given time point. Using multiple microarrays at different time points, we can obtain expression time series for every gene in the genome (Figure 1.2). This information is very useful for understanding changes of biological systems over time [Wodicka et al., 1997, Brown and Botstein, 1999, Debouck and Goodfellow, 1999], and is complementary to the static information from sequences.

On the other hand, data produced from dynamic experiments are often confounded by a number of factors. For example, measurement of gene expression is affected by both biological noise and technical noise. The former may be due to the intrinsic property of gene expression networks [Rao et al., 2002], variation of the global pool of house keeping genes, and fluctuations in environmental conditions that affect all genes [Pedraza and van Oudenaarden, 2005]. The latter may be caused by fluctuations in probe, target and array preparation, in the hybridization process, and effects resulting from image processing [Schuchhardt et al., 2000]. As a result, gene expression is best modeled by random variables. Furthermore, if we assume homologous genes are more likely to have similar expression profiles,



(a)



(b)

Figure 1.2: (a) Gene expression time series for two budding yeast cycling genes, MCM6 and CDC20 in a block-release experiment. (b) Gene expression time series in block-release experiment for human cycling genes Mcm6 and Cdc20, which are homologous to the two genes in (a). The cell cycle period is approximately 80 minutes in (a) and 15 hours in (b). (Data from Spellman et al. [1998] and Whitfield et al. [2002]).

we may be able to take advantage of correlation between expression of homologs, and integrate measurements from different species to derive more accurate results. In this case, static sequence data are able to complement noisy dynamic measurements, enabling us to better interpret experimental observations. For example, Figure 1.2 shows the time series for two budding yeast genes, MCM6 and CDC20, and two homologous human genes, Mcm6 and Cdc20. By examining the time series, it is obvious that budding yeast CDC20 and MCM6 are periodically expressed. Human Cdc20 is also periodically expressed, although with less amplitude or more fluctuation. However, it is much harder to tell whether human Mcm6 is cycling or not. But if we know these genes are homologous to each other, it will boost our confidence that Mcm6 is also a cycling gene. Indeed, human Mcm6 encodes a protein involved in DNA replication, and its expression is regulated by the cell cycle [Dalton and Whitbread, 1995].

1.2 Gene Expression Programs

There are many biological processes within a living cell. In order to maintain a healthy state, all these activities must be regulated at multiple levels, e.g. the transcription, transportation, and degradation of proteins. Transcription of genes, or gene expression, is the first stage of protein assembly, and is highly regulated in all species. In fact, if the principle of parsimony is applicable here, then the regulation of protein assembly should happen at the earliest possible stage, because it can save energy and nutrients which may be vital for the survival in extreme conditions. On the other hand, there is also the need to be able to respond in time to environmental changes, and it may require the regulation of some genes to happen in later stages.

1.2.1 Cell Cycle

The cell cycle, the process in which cells divide, is the most basic process in all cellular organisms. A cell dies if the cell cycle goes wrong and it cannot replicate itself. In higher eukaryotes, cells become cancerous when they lose control and keep dividing. Many genes related to the cell cycle are regulated in a periodic way, and it is likely that they are periodically expressed, peaking at the stage where required.

To identify periodically expressed genes, cells in a population are first arrested at the same stage of the cell cycle. After released from arrest, the now synchronized cells are profiled by microarray experiments at multiple time points. By examining the expression time series, it is possible to identify genes whose expression level

changes with the cell cycle. However, there are borderline cases where it is hard to decide whether a gene is cycling or not (Figure 1.2 (b)). By combining sequence homology and cycling expression, we may be able to recover a more coherent set of cycling genes.

Some of the cycling genes are conserved across species during evolution. By comparing lists of cycling genes from several species, including yeast, plants, and humans, we can derive a core set of cycling genes, as well as cycling genes specific to a single species. It is very likely these core cycling genes make up an essential part of the cell cycle machinery, while species-specific cycling genes are responsible for other recurrent activities developed only in that species.

1.2.2 Immune Response

The immune system is developed in higher eukaryotes to protect the host from pathogens. There are many types of immune cells, including specialized cells that are capable of recognizing and responding to alien substances. For example, macrophages can engulf and digest alien particles, a process called phagocytosis. Dendritic cells are another type of immune cells that can process antigens and present fragments to "train" and activate other cells (T cells and B cells) in the immune system. Dendritic cells are very important for the host to develop antigen specific immunity.

Upon contact with infectious agents, receptor proteins in immune cells are activated, and they in turn activate related pathways leading to proper response to protect the host. Toll-like receptors (TLRs) are one of such receptors and are highly conserved between species [Lemaitre et al., 2003, Beutler, 2004]. There are several types of TLRs, leading to different pathways. To better understanding the molecular mechanism of the immune response, one important challenge is to identify genes participating in these pathways. Because genes that are differentially expressed after infection are very likely to play a role in the immune response, it is useful to first identify these genes.

Genes in immune response are constantly under negative selection pressure, so they tend to be more conserved. Therefore by incorporating sequence information, we will be able to better identify essential genes in the immune response pathways.

In contrast to the cell cycle expression program, which is probably common to all type of cells in an organism, the expression program in immune cells varies between different infectious agents. Moreover, there are multiple types of immune cells, and they have different responses to the same infectious agent.

1.3 Functional Analysis of Gene Expression Programs

As biological knowledge is being accumulated, it is desirable to have a convenient way to incorporate it into the analysis of new biological findings. Functional annotation projects, such as Gene Ontology (GO) [Ashburner et al., 2000], provide a controlled vocabulary that can be used to describe the properties of genes. Using the vocabulary, people can summarize known information about genes and store the annotations into databases, which can be used for future analysis.

When studying gene expression programs, it is often the case that a subset of genes are observed to be significantly different from the rest, e.g. being induced, repressed, or in general differentially expressed. It is very useful if one can quickly characterize the gene set based on existing knowledge, if it contains genes that have already been studied. This can also be used for functional assignment for unknown genes [Zhou et al., 2002]. One approach is to look for functional annotations that are "enriched" in the set of observed genes. These "enriched" annotations can be regarded as a qualitative summary of the experimental outcome. This type of enrichment analysis is also helpful for other computational analysis of gene expression programs, where it can provide quick feedback by summarizing the computational results.

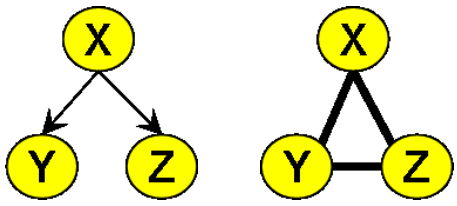
1.4 Graphical Models for Data Integration and Functional Analysis

Probabilistic graphical models have been proposed to deal with many large scale statistical learning and inference problems. In a graphical model, random variables are represented by nodes, and the dependency structure is represented by edges between nodes. The main idea of graphical models is to capture the conditional independencies between random variables, and it gives rise to many efficient algorithms for learning and inference.

There are two major classes of graphical models. One class is directed models, or Bayes Networks, and the other is undirected models, or Markov random fields. In a directed model, nodes are connected by directed edges, and the conditional probability is encoded directly on the edges. Figure 1.3 (a) shows a simple directed model.

In a undirected model, we define potential functions on cliques of the graph, and the joint probability is expressed by the product of potential function divided by a normalization constant. Figure 1.3 (b) shows an example of a simple undirected model of three nodes.

There exist efficient algorithms to estimate parameters from data, and compute



(a) (b)

Figure 1.3: Two types of graphical models. (a) is a directed model of three nodes, in which Y and Z are conditionally independent given X. Conditional probability $P(Y|X)$ and $P(Z|X)$ are defined on edge $\overrightarrow{XY}$ and $\overrightarrow{XZ}$. (b) is an undirected model of three nodes, in which we define *potential functions* $\psi_1(Y,Z)$, $\psi_2(Z,X)$, and $\psi_3(X,Y)$ on the three edges. The joint probability is represented by the product of all potential functions: $P(X,Y,Z) \propto \psi_1(Y,Z)\psi_2(Z,X)\psi_3(X,Y)$.

the posterior of random variables given the data the estimated parameters, making it feasible for dealing with large datasets of thousands of variables.

Our basic idea for integrating data is to use correlation information from sequence similarity to "connect" genes and the experimental observations into a graph. Since the relation of sequence similarity has no direction, it is natural to apply undirected models to this problem. In the simplest form, we use nodes to represent the functional label of genes, e.g. whether a gene is a cell cycle regulated gene. We use potential functions on edges to model correlation information from sequence homology, and use node potential to model information from other experiments, e.g. microarray measurements.

For the functional analysis problem, we are going to adopt a view where the observed gene set is "generated" from latent biological processes. The goal is then to identify these latent processes. In this case, a directed graphical model is more suitable because it is straight-forward to represent the generative process by directed relations.

1.5 Previous Work

There are a number of previous papers on combining sequence and expression data to study similarities in expression between different species. For example, Bergmann et al. [2004] clustered data from six different species to identify modules of genes that are co-expressed. Stuart et al. [2003] identified 'metagenes', a group of homologous genes from four different species (one gene from each species), and then used correlation coefficients to link metagenes forming a co-expression network.

Our approach differs from these papers in several important aspects. First, unlike prior work that relied on clustering to identify groups of co-expressed genes under a wide range of conditions, our approach uses a *classification* framework to achieve a different goal: identifying a set of conserved systems genes. Second, prior work only looked at pairwise expression similarities, whereas our algorithm utilizes the complete graph topology to propagate information. Finally, previous papers used sequence similarity as a binary value (similar or not). In contrast, our framework uses the extent of this similarity to determine edge weights. The higher the similarity the greater the importance of neighboring genes for determining the final label assignment. More specific studies have been carried out for some biological systems and we expand on those in the relevant chapters.

Enrichment analysis has become an increasingly popular method for analyzing gene sets. Perhaps the most commonly used method of GO enrichment analysis is based on computing a p-value using the hypergeometric distribution (the hypergeometric method or the "Classic" method). Although it is widely used, there still exist some unsolved challenges. For example, the functional categories in Gene Ontology are organized into a hierarchical structure, while the hypergeometric method assumes they are independent. This assumption leads to underestimation of p-values and the hypergeometric method often returns very redundant results. There are a number of efforts that try to address this problem Grossmann et al. [2006], Alexa et al. [2006]. In my thesis, I am going to introduce a new method that explicitly takes into account the hierarchical structure of GO, and dramatically improves on existing methods.

1.6 Contribution

In my thesis, I propose a new framework to integrate sequence and microarray data, and use it to identify genes in the two expression programs, the cell cycle and the immune response.

The major contribution of this thesis is a principled framework for integrating high-throughput biological datasets. It allows combining correlated datasets across different species and/or cell types for more accurate analysis of gene expression programs. In addition, this thesis presents a generative model for functional analysis of gene expression programs.

I believe the algorithms and methods introduced in this thesis will help researchers better utilize the rich information in the ever-growing amount of high-throughput datasets. I also believe generative models can be a powerful tool for solving other problems in the analysis of biological data, especially when more knowledge of the underlying biological mechanisms becomes available.

1.7 Organization

This thesis develops probabilistic models in a framework that combines multi-species microarray data and applies the models to two gene expression programs. Because the analysis of gene expression programs usually results in gene sets, we first develop a tool for functional analysis of gene sets in Chapter 2, which we will use in the subsequent chapters. In Chapter 3, we develop a probabilistic model to combine gene expression time series and protein sequence data and apply it to identify a core set of cell cycle genes. In Chapter 4, we develop an improved model to combine gene expression data and sequence data, and apply it to identify innate immune response genes conserved between two cell types and/or in human and mouse. The last chapter outlines potential application of the framework we have developed to other areas.

Chapter 2

A Generative Model for Gene Function Enrichment Analysis

While the main goal of this thesis is to develop models for analysis of multi-species microarray data, in this chapter we first develop a tool, which we will use in the following chapters to analyze functional enrichment in various gene sets identified by our models. Unlike other existing tools for the same task, in many cases our tool is able to characterize the functionality of a gene set with much less redundancy. In the following sections, we will introduce the probabilistic model for our tool, and compare its performance with several other methods.

2.1 Introduction

High-throughput experiments in molecular biology are generating large quantities of data, which enable researchers to study biological systems, such as gene expression programs. In many cases these datasets are in the form of lists of genes. For example, it can be a set of differentially expressed genes, or the targets of a transcription factor. However, due to the size of the lists it is often difficult to manually inspect them to functionally characterize the experimental outcome. To overcome this challenge, researchers increasingly rely on computational analysis using curated databases of functional annotations. These include the Gene Ontology (GO) [Ashburner et al., 2000] and the MIPS [Mewes et al., 2002] database, among others. In these databases genes are annotated by standardized terms, summarizing existing knowledge of the genes. For example, in Gene Ontology categories are used to indicate a gene's known functions or related biological processes.

While using curated functional databases to analyze high-throughput experiments has led to some success, there are many problems remain to be solved.

One challenge is multiple hypotheses testing, because GO contains thousand of categories which are all tested for enrichment for the same gene set [Ernst and Bar-Joseph, 2006]. Another challenge lies in the fact that the categories to which genes are assigned are not independent, making it hard to determine if a set of identified significant categories are indeed different functional outcomes, or rather a redundant view of the same biological process. For example, GO categories are organized into a hierarchy with more general categories close to the root and more specific categories at the bottom. Genes annotated by a specific term are implicitly annotated to all parent terms, resulting in highly overlapping categories. In addition, many genes are assigned to multiple categories that do not share a directed path in the GO hierarchy, resulting in overlapping categories that cannot be detected using the hierarchical structure. In both cases, the dependency between categories make it hard to identify the most informative functional annotations. In fact, when using GO to compute hypergeometric p-values, which is the most common method used [Fischer et al., 2006], researchers often recover several redundant categories as the top hits (see Table 2.2) which both mask other important categories and make it hard to determine the most relevant category.

2.2 Prior Work

The problems caused by dependency of the GO categories have been recognized and a few methods were developed to address them. One of the first attempts was the use of 'GO Slim' (http://www.geneontology.org/GO.slims.shtml), a leaner version of GO containing a manually picked small set of categories (130 of the current $\sim 24,000$ categories in GO) with a small overlap between them. While useful, this method only retains the general categories and does not provide more specific ones which are often most interesting to biologists. Other attempts were proposed by a few recent papers. Grossmann et al. [2006] adjust the p-value for a specific category by taking into account the immediately more general terms (the parents). This can often lead to the removal of false positives, since some of the more specific categories are eliminated if their parent category is determined to be significant. Alexa et al. [2006] proposed two algorithms to correct the p-values for a specific GO term. The first algorithm, 'Elim', tests the enrichment of each GO category in a gene set by examining the GO hierarchy in a bottom-up order. Once a GO category is determined to be significant, all genes associated with it are removed in the following analysis of its ancestral (more general) categories. The other algorithm, 'Weight', uses a similar strategy but rather than completely removing genes in significant categories it down-weights them for the remaining categories.

2.3 Our Method: Probabilistic Generative Model

While these methods are more powerful, they only utilize local information in the graph structure (parent-child or bottom-up). Thus, they cannot account for longer range relationships and global dependencies such as highly overlapping categories that do not share a directed path. In addition, all the above-mentioned methods return a (sometimes long) list of GO categories with their p-values requiring the user to select a cutoff in order to further analyze the resulting list.

Our approach is different. From a biological point of view, one of the goals of using functional databases is to identify a set of biological processes related to the specific study. Thus, it would be natural to use a generative model to globally identify the set of significant GO categories and processes that 'generated' the observed list. Our goal is to identify a (preferably small) set of categories that together account for the set of genes observed. Since many experiments study complicated responses involving several processes, the categories can come from different locations and levels in the hierarchy. However, highly overlapping categories will not be selected since one of them is often enough to explain the subset of the genes belonging to these categories.

We applied our method, which we term GenGO (GENerative GO analysis), to analyzing the GO hierarchy for yeast and humans. We used a controlled analysis (in which subsets of categories are selected and the goal is to recover the (hidden) categories), microarray expression data and ChIP-chip data for both species. GenGO was able to drastically reduce the false positive rates, even after statistical correction. As we show, GenGO consistently outperforms both the original hypergeometric method and the methods considering only local structural dependencies, in some cases dramatically so.

2.4 The Activation Graph for GO Categories

We developed a generative model to identify a subset of active GO categories. When designing the method we placed special emphasis on simplicity and speed. GO analysis is often an interactive process in which users change their lists, or analyze multiple lists (for example different gene clusters or different targets of transcription factors). Thus for a method to be successful it should be computable in a reasonable time to allow interactive analysis.

To explain our method, one can think of this problem in terms of a bi-partite graph representing the relationships between GO categories and genes (Figure 2.1). Nodes on the left side of the graph represent GO categories and nodes on the right represent all genes annotated in that species. We connect a gene node with a GO

node by an edge if and only if the gene is annotated to belong to that GO category. We denote genes that were identified in the experiment as 'ON' or active and genes that were not identified as 'OFF' or inactive. Similarly, when a biological process (corresponding to a specific GO category) is active, we represent it by setting its GO node to 'ON' and when it is inactive, we set its state to 'OFF'.

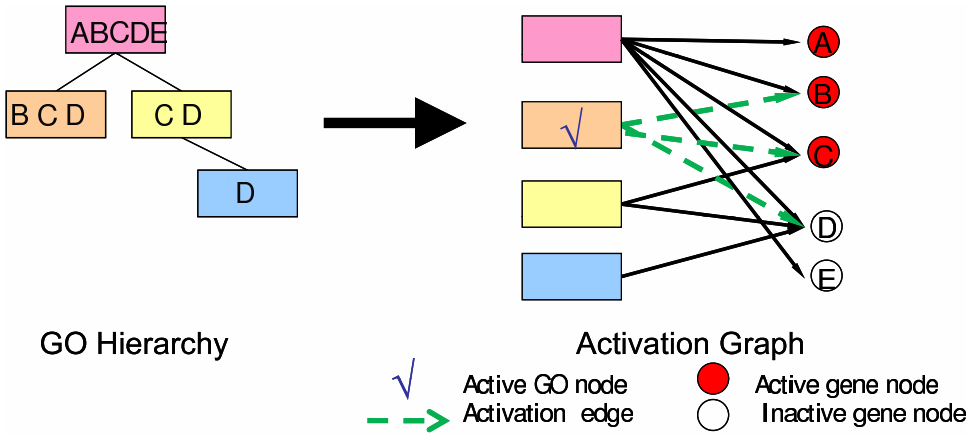


Figure 2.1: Construction of an activation graph. Left: A diagram showing a GO hierarchy of four categories and the five genes annotated by these categories (letters in each rectangle). Because of the true path rule, each gene annotated by a category in the GO hierarchy is also annotated by all its parent categories. Right: The activation graph corresponding to this GO hierarchy when observing three of the genes (A,B,C). In this graph, we connect a gene node with a GO node if and only if the gene is annotated by that GO category. For this set of genes the active category is determined to be the orange category. Note that due to noise there is a gene that is selected even though it does not belong to the active category (A). Noise is also responsible for the fact that a gene belonging to the active category is not selected (D).

To find this set we define a probabilistic model on the activation graph (containing both gene and GO nodes). The model accounts for noise in the experimental and GO data. We develop an algorithm that identifies active GO categories by maximizing the likelihood of this model conditioned on the set of active genes. The final outcome is a small subset of active GO nodes that together explains the set of active genes. We describe the model in details in the following sections.

2.5 Probabilistic Model for Activation Graphs

We assume a generative model for gene activation. In this model we first select a subset of GO categories and activate all genes in these categories. Next, a random process (representing noise, errors in GO assignments and partial knowledge) inactivates, with probability $1 - p$, genes in each of the selected categories and activates, with probability q, genes in categories that were not selected leading to the observed gene set. Given a list of active (selected) genes and a set of active GO categories, we can define the following sets

- A_g active gene nodes connected to at least one active GO node
- A_n active gene nodes not connected to any active GO nodes
- I inactive gene nodes
- S_g edges connecting nodes in I with active GO nodes
- S_n edges connecting nodes in I with inactive GO nodes

Using these symbols we define the following log-likelihood function which we would like to maximize:

$$L(C|p, q, G) = |A_g| \log p + |A_n| \log q + |S_g| \log(1 - p) + |S_n| \log(1 - q) - \alpha|C| \tag{2.1}$$

where G is the set of active (selected) gene nodes (the input), C is the set of active GO nodes, and $|X|$ represents the size of the X group (A_g, A_n etc.). This function captures our generative model. With probability p genes belonging to active categories would remain active (A_g). With probability q genes that do not belong to any active category would be activated (A_n). Similarly, with probability $1 - p$ genes in active categories will become inactive (S_g) and with probability $1 - q$ genes in inactive categories will remain inactive (S_n). The last term in the likelihood function penalizes the size of the set of active GO categories ($|C|$) so that the model will prefer a smaller set of categories when explaining the selected set of genes. The hyperparameter α is a positive number controlling the penalization.

The above likelihood model is a function of the selected set of active GO categories (denoted by C). In the next section we present an algorithm for finding such a set that maximizes this likelihood. We also present a method for optimizing the values for the noise parameters p and q. Once the algorithm terminates we compute a p-value score for each of the selected categories using hypergeometric distribution and return an ordered list of selected categories to the user.

2.5.1 Optimization by Greedy Search

Given an input list of active genes from an experiment, we would like to determine a set of active GO categories (C) that maximizes the likelihood function (Eq 2.1). This is an NP-hard problem (e.g. it can be shown that the Maximal Set Cover problem can be reduced to it). Thus, we use a simple and fast greedy search algorithm to look for a local maximum of the likelihood function.

The algorithm is as follows (p and q are fixed in this part; they can either be optimized in an outer loop as we discuss below or set by the user in advance.).

Algorithm 1 (Find the best GO set for given parameters)

1. Initialize C_0 to be the empty set

2. At iteration i, we consider all possible one-step changes of the current set of active GO categories (C_i), and compare the likelihood of the resulting sets. Let

   $$t_1^i = \text{argmax}_{t \in C_i} L(C_i \backslash \{t\}) \text{ and } t_2^i = \text{argmax}_{t \in T \backslash C_i} L(C_i \cup \{t\}),$$

   where T is the set of all GO categories. Thus among all possible reductions of C_i, $C_i^- = C_i \backslash \{t_1^i\}$ has the highest likelihood. Similarly, among all possible expansions of C_i, $C_i^+ = C_i \cup \{t_1^i\}$ has the highest likelihood.

3. If the likelihood of C_i^- is higher than that of both C_i^+ and C_i, let $C_{i+1} = C_i^-$ and go to step 2.

4. If the likelihood of C_i^+ is higher than the likelihood of C_i, let $C_{i+1} = C_i^+$ and go to step 2. Otherwise go to the next step.

5. return C.

It is important to note that including more GO categories will not necessarily lead to improved likelihood and thus the algorithm above does not overfit the data. The reason is that there is an associated penalty if the category added includes genes that were not selected. Adding a category for which many of its genes were not selected or if they were selected they are already explained by other selected categories will usually lead to reduction in the likelihood.

Once the algorithm terminates, we use the set of active categories as the final result. For these categories we compute a p-value using the hypergeometric distribution and return the list, ordered by the p-value significance score, to the user. Corrected p-values can also be computed either by using the Bonferroni correction or by carrying out randomization tests [Ernst and Bar-Joseph, 2006].

2.5.2 Optimizing Parameters

There are two parameters in our model, p and q. p is the probability that an active GO node will activate a gene belonging to that GO category and q is the probability that a gene node becomes active without being activated by any GO node. A higher p means a higher participation rate of the related genes in the biological process, and/or less uncertainty in the activation relation between a GO node and the related go nodes. A higher q means a larger portion of the genes are allowed to be explained by background noise or errors in the current ontology. p and q can be set manually according to the estimation of noise level. These two parameters can also be optimized by maximizing the log likelihood defined previously. The algorithm is as follows:

Algorithm 2 (Find the best GO set by learning parameters p and q)

1. Initialization. Set $p_0 = 0.5$, $q_0 = |G|/|R|$, where G is the set of active genes, and R is the reference set.

2. Carry out steps in Algorithm 1, using p_i and q_i.

3. Based the solution found in the previous step, we compute the maximum likelihood estimation of p and q:

$$p_{i+1} = \frac{|A_g|}{|A_g| + |S_g|}$$
$$q_{i+1} = \frac{|A_n|}{|A_n| + |S_n|}$$

4. if $\max(|p_{i+1} - p_i|, |q_{i+1} - q_i|) \geq \epsilon$, go to step 2, otherwise stop. (ϵ is a small positive number to control convergence.)

Because both steps in Algorithm 1 and 2 only increase the likelihood, the algorithm above is guaranteed to converge to a local maximum.

The hyperparameter α can be chosen by experiments and we found it generally works well when setting $\alpha = 3.0$.

GO annotation data. Gene ontology files (release 2007-06) were downloaded from the Gene Ontology website (`ftp://ftp.geneontology.org/`). GO annotations for humans and yeast were extracted from the Gene2GO database, which was downloaded from the NCBI website (`ftp://ftp.ncbi.nlm.nih.gov/`) on Jun 26, 2007. GO categories were filtered such that only those with at

least 5 genes would be used. In this study, we focused on the Biological Process categories, but our methodology is also applicable to Cellular Component and Molecular Function categories.

Precision/Recall curves. Precision/Recall plots were done using the ROCR package in R (`http://www.r-project.org/`). Each point in the precision/Recall curve corresponds to a score (or p-value) cutoff. The precision and the recall are defined as follows

$$\text{Precision} = \text{TP}/(\text{TP} + \text{FP})\text{Recall} = \text{TP}/(\text{TP} + \text{FN}),$$

where TP is the number of true positives (true active categories below the cutoff), FP is the number of false positives (inactive categories below the cutoff), and FN is the number of false negatives (true active categories above the cutoff).

Precision/recall curves are more informative than Receiver Operating Characteristic (ROC) curves when working with highly skewed datasets [Davis and Goadrich, 2006]. This is exactly the case when working with GO enrichment analysis in which the vast majority of categories are not expected to be enriched for any one dataset.

Comparison. For comparison with the Classic method we used the hypergeometric p-value analysis from STEM [Ernst and Bar-Joseph, 2006]. We used the Parent-Child method implemented by Ontologizer (`http://www.charite.de/ch/medgen/ontologizer/recomb06/index.html`), and the Weight and Elim methods implemented in the topGO package (release 1.2.1) in R (release 2.5.1). For both Classic and Parent-Child methods, p-values are computed with Bonferroni correction, which is a commonly used method for multiple testing correction.

In every GO analysis task we performed for a species, we used the whole set of annotated genes as the reference set. To generate the precision/recall curve for a method in a specific experiment, we followed the strategy in Grossmann et al. [2006] and accumulated all p-values from 100 random gene sets.

Ranking induced genes in amino acid starvation. For each yeast gene in the amino acid starvation experiment, we looked at its second highest expression level throughout the whole time series, and ranked all genes according by this value.

2.6 Results

2.6.1 Comparison by Selecting a Subset of Categories

We first tested our method (GenGO) using GO data for yeast and humans. We followed the same procedures in Grossmann et al. [2006] and Alexa et al. [2006]

for objective comparison of different GO analysis methods. For each species, 1, 2, or 5 GO categories were randomly selected as 'active', and a subset of genes associated with each active category were randomly picked (90% or 50% of genes in each of the selected categories). In addition, we randomly selected 1% or 15% of the remaining genes (from inactive categories) and combined the two sets from active and non-active categories to form the input to the GO analysis. Due to the large run time of some of the methods we were comparing to (Elim and Weight), for each experiment, 100 random sets were generated using the same parameters.

We used precision/recall curves to compare GenGO with four other methods (Materials and Methods). These included 'Classic' (hypergeometric test) and the three other methods listed above. The results are plotted in Figures 2.2 (yeast) and 2.3 (human). For all settings, the performance of GenGO dominates all other methods. When the noise level is low, the performance of GenGO is close to optimal (left columns in Figures 2.2 and 2.3). When the noise level is high, the performance drops for all methods, though GenGO is still the best. Even with high noise and multiple categories (as is the case for most real experiments) GenGO can achieve 80% precision for high recall levels (60%-80%). As for the other methods, in most cases 'Weight' is the second best and 'Classic' is usually the worst, indicating that all methods previously proposed for the task indeed improve upon the standard usage of GO.

Note that while the precision usually drops as the recall increases, there could be cases where the precision actually improves even though recall is increasing. For example, in Figure 2.2(a) the 'GenGO' method correctly assigns the lowest p-values to some of the selected categories, which results in a very high precision rate at low recall rates. However, when the recall increases to 0.1, due to some non-selected categories that are (incorrectly) assigned a low p-value, the precision drops to 0.9. As the recall continues to increase, the precision increases again because the method recovers the rest of the selected categories without picking up much non-selected categories.

2.6.2 Analysis of Noise Datasets

To test the ability of GenGO to overcome the multiple hypothesis testing problem, 5% and 10% of all human genes were randomly selected as a test set, and the five algorithms were run to identify significant categories. The procedure was repeated 100 times, and the percentages of sets without any significant GO categories (p-value < 0.001 with Bonferroni correction where applicable) are listed for each of the methods in Table 2.1. Even after correction the Classic method, which is the most commonly used, identified significant categories in all experiments. When 10% of genes were selected at random, all methods, except for GenGO identified

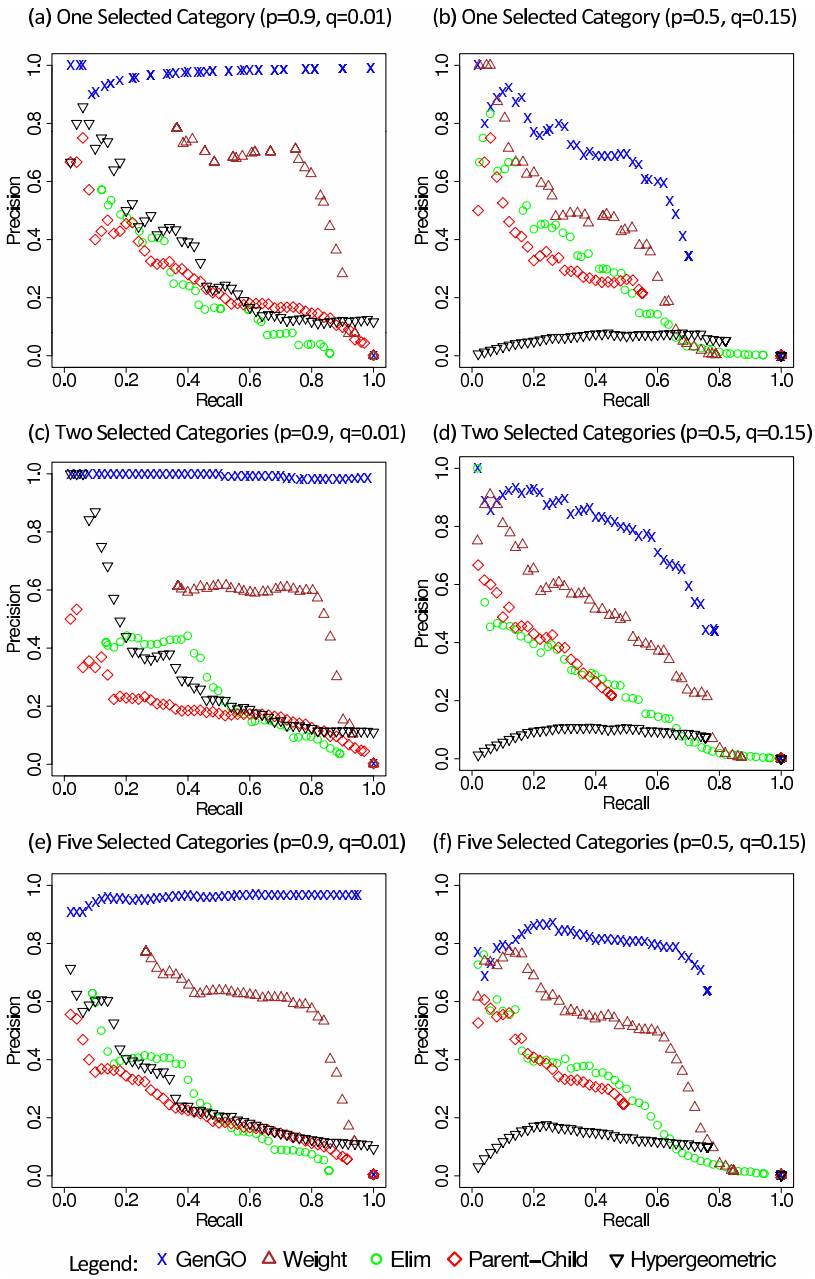


Figure 2.2: Comparison using GO for yeast. Performance comparison of five methods on data generated using the yeast GO database. We use p to represent the fraction of genes that are identified from an active GO category (true positive rate for a category, see Materials and Methods) and q to represent the fraction genes that are selected but do not belong to any active category. (a) Selecting one category with $p = 0.9$, $q = 0.01$ (b) Selecting one category with $p = 0.5$, $q = 0.15$. (c) and (d): same as (a) and (b) but using two categories. (e) and (f): same with five categories. Note that even when the noise is substantial (using 50% of genes in selected categories and 15% of all other genes, second column) GenGO is still able to accurately recover most of the correct categories.

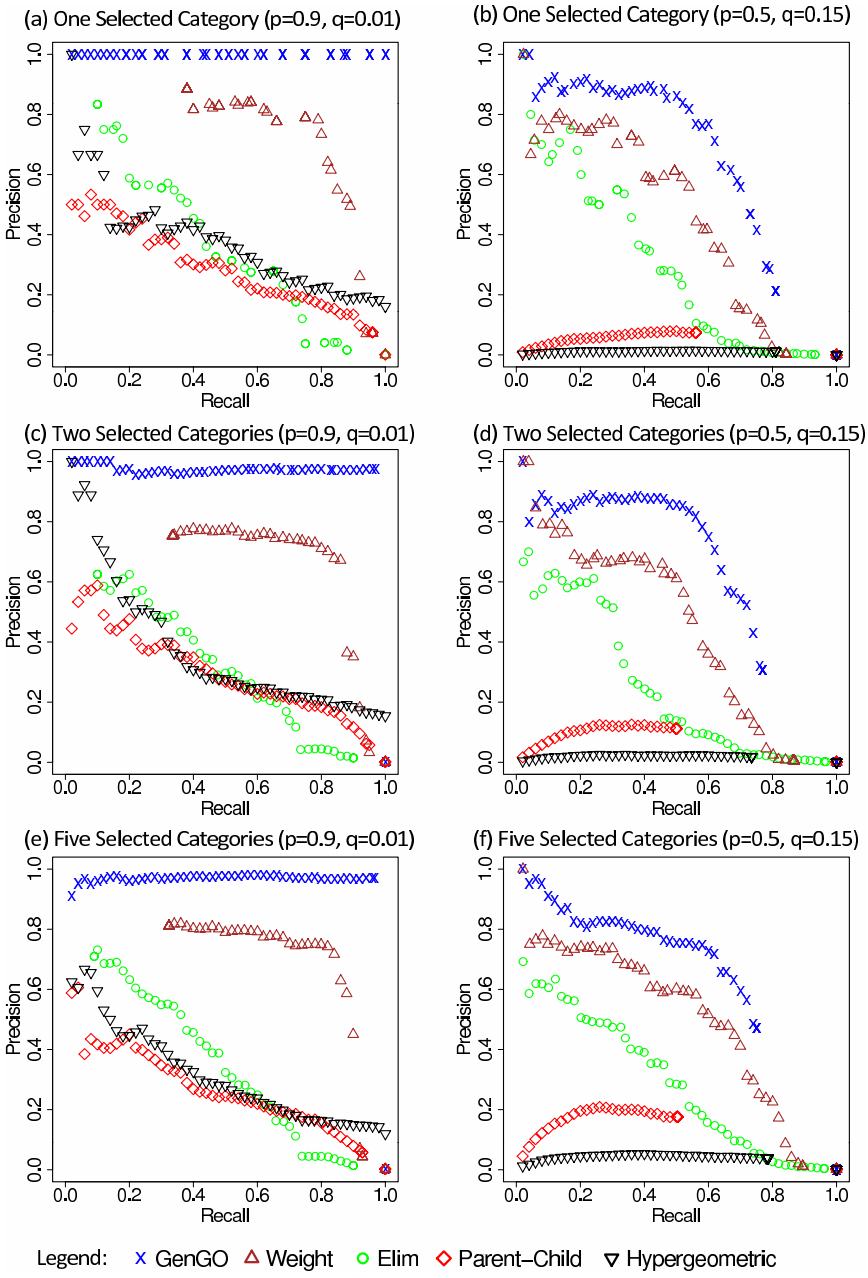


Figure 2.3: Performance comparison of five methods on data generated using human GO database. (a-f) same as in figure 2.2 for human GO data.

significant categories in at least 50% of the experiments. In contrast, GenGO was able to determine that no such significant category exists for more than 97% of tested noise sets.

Random Genes	Classic	Parent-Child	Elim	Weight	GenGO
1%	69%	100%	67%	64%	100%
5%	0%	83%	74%	71%	98%
10%	0%	7%	51%	44%	100%

Table 2.1: Analysis of random gene sets. 1%, 5%, and 10% of all human genes were randomly selected as a test set, and the five algorithms were run to identify significant categories. Categories were only selected if they achieved a p-value $<$ 0.001 following Bonferroni correction for multiple hypothesis testing. The procedure is repeated 100 times, and the percentages of sets without any significant GO categories are listed in the table. As can be seen, while GenGO correctly determined that there were no significant categories in more than 98% of tests, other methods identified much more erroneous categories in these experiments.

2.6.3 Comparison on Microarray Experiment for Yeast

Testing GenGO using real expression data is more challenging since the 'ground truth' is unknown in most cases. Still, when the biological condition is clearly defined, it is possible to determine whether a set of GO categories provides a good summary of the experimental setup.

Enrichment Analysis of Cell Cycle Genes

We have initially applied GenGO to analyze the well studied cell cycle expression dataset from Spellman et al. [1998]. We used the 800 genes determined to be cycling during the mitotic cell cycle in budding yeast. Figure 2.4 plots the location in the GO hierarchy of the top five categories identified by four of the five methods (see also Table 1 and Supplementary Figure 3). The results highlight the advantages of GenGO. For example, while both GenGO and Classic successfully identify "mitotic cell cycle" as the most significant category, the Classic method returns highly redundant categories including "mitotic cell cycle", "cell cycle process", and "cell cycle". The Parent-Child method [Grossmann et al., 2006] also returns redundant categories ("cell cycle process", and "cell cycle") though it does a better job in finding the more specific "microtubule-based process" which is related to cytoskeleton changes during cell cycle progression [Spellman et al., 1998]. Both Elim and Weight fail to identify the most appropriate category for this data

(cell cycle) though they do identify a number of relevant specific categories. In contrast, GenGO contains both the correct high level categories ('cell cycle' and 'cell division') as well as more specific categories ("chromatin assembly or disassembly") that play an important role in DNA replication and chromosome segregation. Note that cell division here is not redundant with cell cycle. While "cell cycle" describes the different phases of the cell cycle, their regulation, and checkpoints, 'cell division' refers to the process of separation of daughter cells following the cell cycle.

Classic	Parent-Child	Elim	Weight	GenGO
mitotic cell cycle	cell cycle	microtubule nucleation	microtubule nucleation	mitotic cell cycle
DNA replication	cell cycle process	mitotic sister chromatid cohesion	mitotic sister chromatid cohesion	DNA replication
cell cycle	DNA metabolic process	mitotic spindle organization and biogenesis	DNA strand elongation during DNA replication	microtubule-based process
cell cycle process	microtubule-based process	DNA replication initiation	mitotic spindle organization and biogenesis	cell division
DNA-dependent DNA replication	DNA replication	telomere maintenance via recombination	telomere maintenance via recombination	chromatin assembly or disassembly

Table 2.2: Top five GO categories identified by different methods from the list of periodically expressed yeast genes during the mitotic cell cycle.

Enrichment Analysis of Amino Acid Starvation Response Genes

We repeated the above analysis using the top 500 induced genes in amino acid starvation experiments [Gasch et al., 2000]. Only GenGO and Weight correctly identified "amino acid biosynthetic process" as the most significant category (Table 2.3 and Figures 2.5- 2.9). The next significant category identified by GenGO is "sulfur metabolic process". It includes genes required in recycling sulfur metabolites,

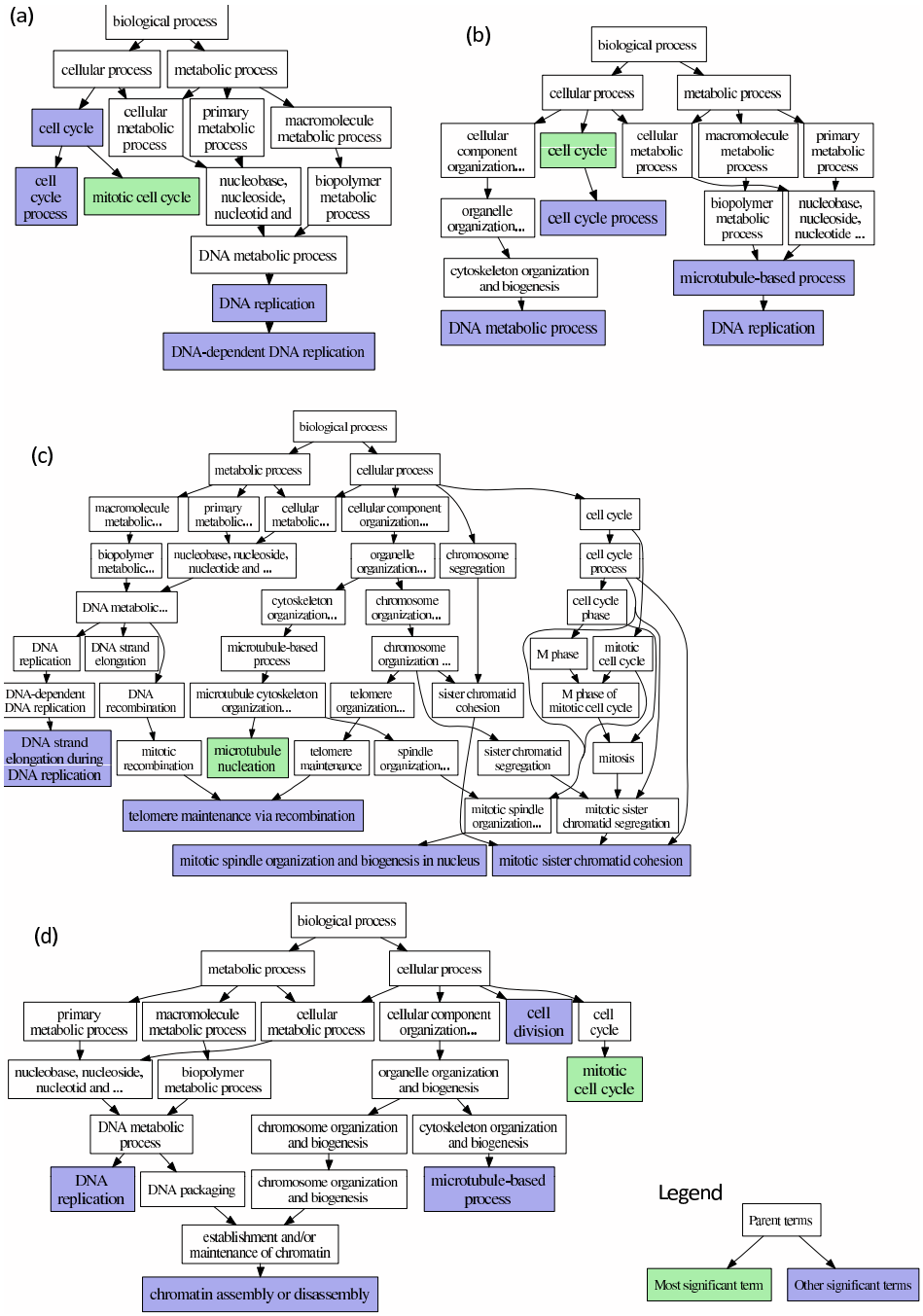


Figure 2.4: Comparison of top five GO categories identified in the yeast cell cycle genes Spellman et al. [1998] by four methods. (a): top five GO categories identified using the Classic method (hypergeometric p-value) are highlighted. Green represents the most significant category identified. The five categories represent highly redundant view of only two biological processes, as highlighted by the red circles. (b): Parent-Child method Grossmann et al. [2006]. (c): Weight method Alexa et al. [2006]. (d) GenGO.

which are known to be highly expressed under amino acid starvation [Thomas and Surdin-Kerjan, 1997]. In addition, an interesting finding by GenGO is “monosaccharide catabolic process”. During amino acid starvation, besides the lack of amino acid there is a cellular need to produce energy which is carried out mainly by this process [Natarajan et al., 2001]. Another category identified by GenGO, ”amino acid catabolic process”, describes the process that generates amino acids from existing proteins, which is a known consequence of amino acid starvation. In contrast, the categories identified by Elim are too specific: three of the five categories are subcategories of “amino acid biosynthetic process” and can be better summarized by the latter. The Classic method again identifies redundant categories: ”organic acid metabolic process”, ”carboxylic acid metabolic process”, and ”amino acid metabolic process”.

Classic	Parent-Child	Elim	Weight	GenGO
nitrogen compound metabolic process	nitrogen compound metabolic process	arginine biosynthetic process	amino acid biosynthetic process	amino acid biosynthetic process
carboxylic acid metabolic process	organic acid metabolic process	glutamate biosynthetic process	glutamate metabolic process	sulfur metabolic process
organic acid metabolic process	amino acid and derivative metabolic process	sulfate assimilation	sulfur amino acid metabolic process	amino acid catabolic process
amino acid metabolic process	amine metabolic process	transposition, RNA-mediated	main pathways of carbohydrate metabolic process	purine base metabolic process
amino acid and derivative metabolic process	cellular biosynthetic process	methionine biosynthetic process	glutamine family amino acid catabolic process	monosaccharide catabolic process

Table 2.3: Top five GO categories identified by different methods from the list of yeast genes induced following amino acid starvation [Gasch et al., 2000].

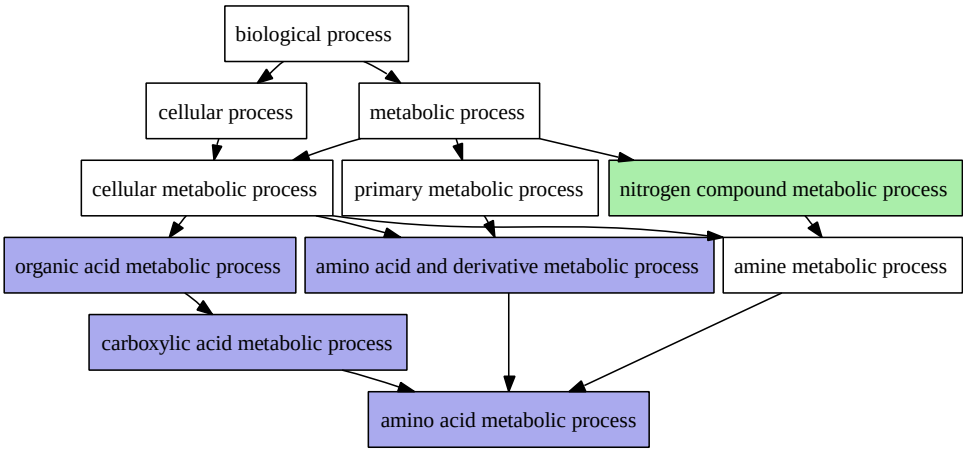


Figure 2.5: Top five categories identified by the hypergeometric method for yeast genes induced in amino acid starvation.

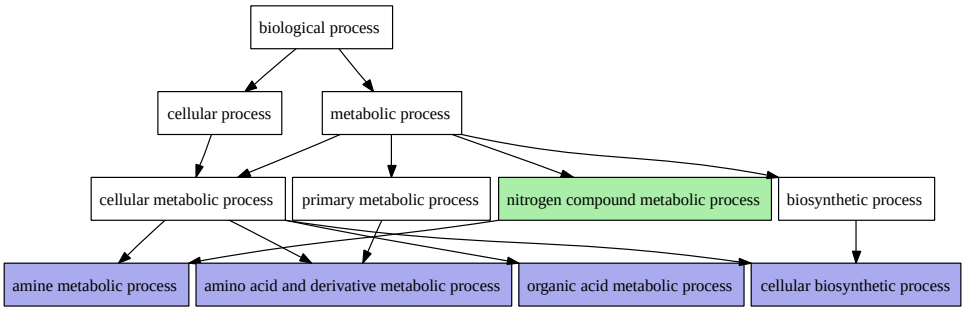


Figure 2.6: Top five categories identified by Parent-Child method for yeast genes induced in amino acid starvation.

2.6.4 Analysis of Human Expression Data

We repeated the analysis described above using human immune response experiments from [Nau et al., 2002]. 977 genes were identified as differentially expressed when host cells were exposed to one or more bacterial pathogens. For this set all methods have correctly identified “immune response” in the top two categories (Table 2.4). However, as was the case for yeast, the Classic method returned many redundant categories. Parent-Child returned two very general categories (‘biological process’ and ‘regulation of biology’) which do not provide insight into the set of genes. Interestingly both Elim and Weight identified ‘response to virus’ as one

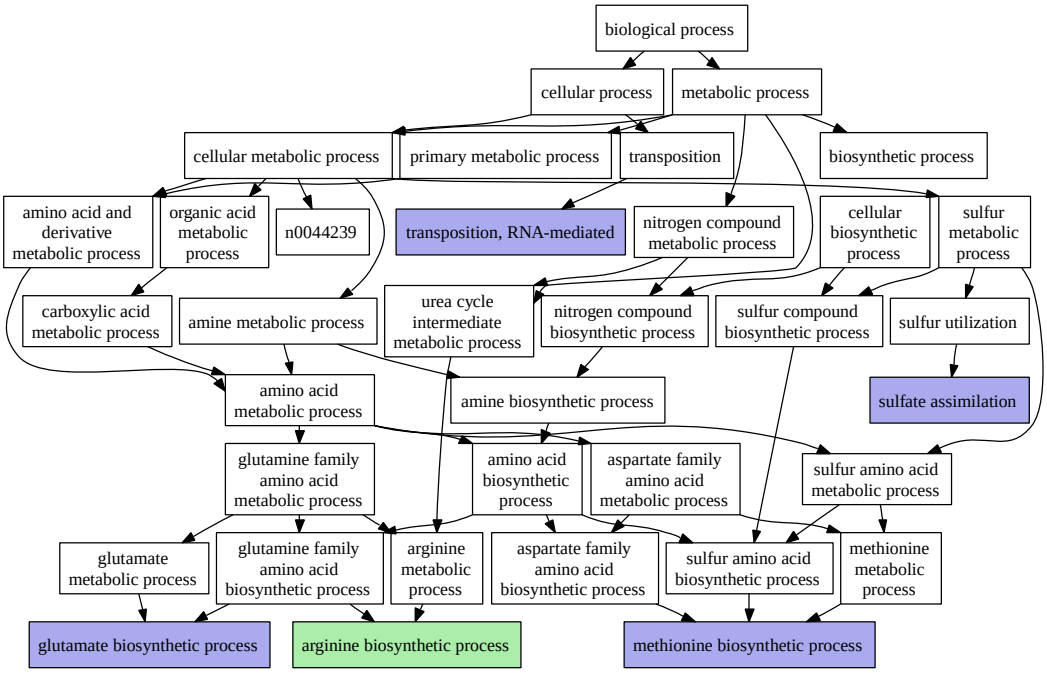


Figure 2.7: Top five categories identified by Elim for yeast genes induced in amino acid starvation.

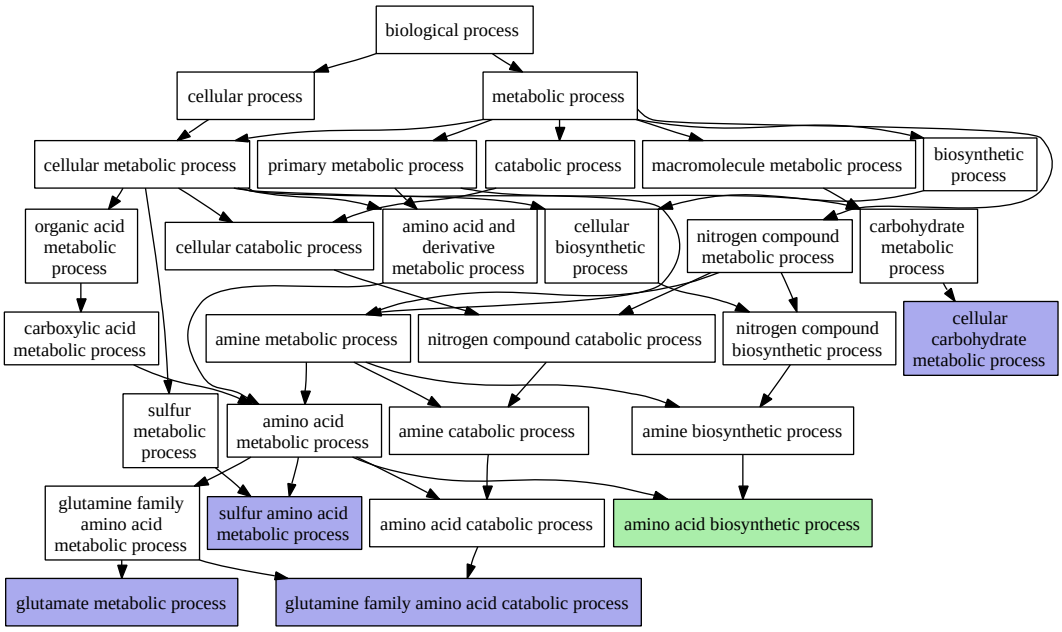


Figure 2.8: Top five categories identified by Weight for yeast genes induced in amino acid starvation.

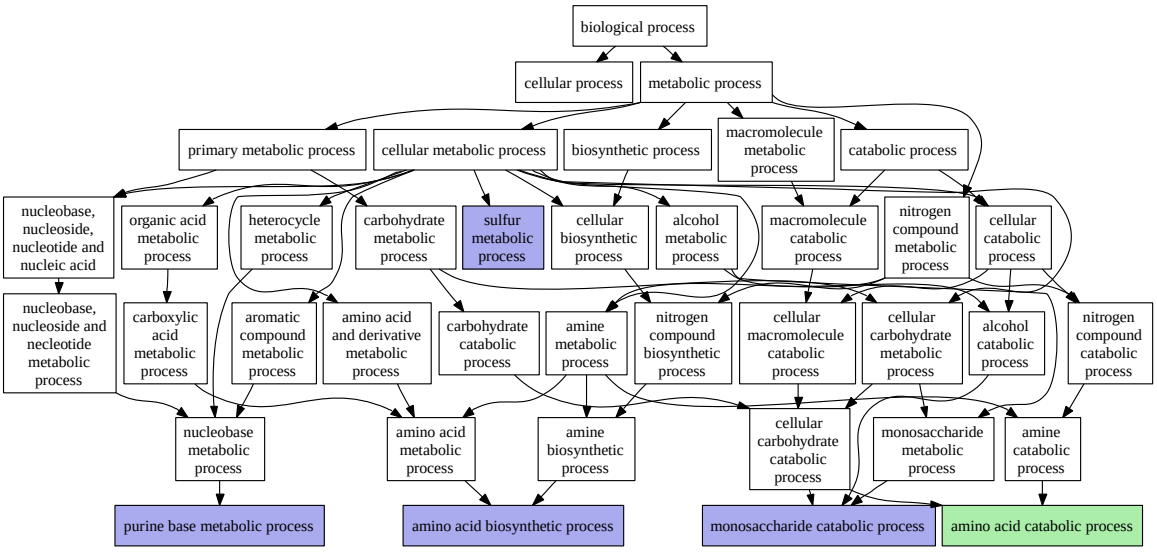


Figure 2.9: Top five categories identified by GenGO for yeast genes induced in amino acid starvation.

of the top five categories. Since only bacteria were used in this study this category should not have been identified. It was likely selected by these methods due to its overlap with the more general 'immune response' category. In addition to the 'immune response' and 'wound response' categories identified by GenGO it also identified 'taxis' which is clearly relevant due to the mobility need for macrophages during immune response [Jones, 2000]. GenGO also identified "regulation of apoptosis" which plays an important role in determining the drastically different fates for macrophages after infection [Grassmé et al., 2001, Navarre and Zychlinsky, 2000, Rojas, 1997]. The final category identified, "tRNA aminoacylation" is the process that joins an amino acid to its cognate tRNA, which is an important step in protein translation [Park et al., 2005].

Classic	Parent-Child	Elim	Weight	GenGO
immune response	biological process	immune response	immune response	immune response
immune system process	immune system process	inflammatory response	response to wounding	response to wounding
response to stress	response to stimulus	chemotaxis	cell proliferation	taxis
response to stimulus	cell proliferation	response to virus	chemotaxis	regulation of apoptosis
response to wounding	biological regulation	anti-apoptosis	response to virus	tRNA aminoacylation

Table 2.4: Top five GO categories identified from the list of human genes determined to be differentially expressed following exposure to bacteria.

2.6.5 Application to ChIP-chip Data Analysis

ChIP-chip [Harbison et al., 2004] is an experimental technique the combines CHromatin ImmunoPrecipitation with microarrays ("chip"), which can be used to identify the targets of transcription factors. These targets can later be used to shed light on the functional role of that factor, which can be done by using GO to determine the function of the resulting gene target set [Bar-Joseph et al., 2003]. We have compared the GO enrichment analysis of the different methods for the targets of transcription factors from yeast and human.

For yeast we have looked at Swi6, a cell cycle regulator of G1 transcription [Nasmyth and Dirick, 1991]. Table 2.5 presents the results of the five methods

for this factor. Except for Elim and Weight, which did not return 'cell cycle' in their top 5 hits, the three other methods correctly selected this as the top category for Swi6. However, the hypergeometric and parent-child again returned a set of redundant categories ('cell-cycle', 'cell cycle process'). In contrast, GenGO was able to balance the more detailed and more high level categories. Specifically it was the only one to correctly identify 'reproduction' as one of the top categories for Swi6, a role that is well documented [Leem et al.].

Classic	Parent-Child	Elim	Weight	GenGO
cell cycle	cell cycle	regulation of cyclin-dependent protein kinase activity	regulation of cyclin-dependent protein kinase activity	cell cycle
mitotic cell cycle	cell cycle process	G1/S-specific transcription in mitotic cell cycle	interphase of mitotic cell cycle	external encapsulating structure organization and biogenesis
regulation of progression through cell cycle	biological regulation	cell wall organization and biogenesis	regulation of progression through mitotic cell cycle	DNA replication
regulation of cell cycle	regulation of cellular process	axial bud site selection	axial bud site selection	reproduction
cell cycle process	regulation of cell cycle	positive regulation of DNA replication	cell wall organization and biogenesis	regulation of transcription

Table 2.5: Categories for Swi6 targets identified by ChIP-chip experiments.

We have also looked at the analysis of targets of E2F1, a human cell cycle regulator. Ren et al. [2002] have studied the targets of E2F1 and based on their detailed analysis determined in their title that "E2F integrates cell cycle progression with DNA repair, replication, and G2/M checkpoints". While all GO analysis meth-

ods correctly identified E2F1's role in controlling various aspects of the cell cycle, GenGO was only method to rank all three functions (replication, DNA repair and G2/M checkpoint) in its top 5 categories. (See Table 2.6).

Classic	Parent-Child	Elim	Weight	GenGO
DNA metabolic process	cell cycle process	cell division	DNA replication	DNA replication
cell cycle process	cell cycle	DNA replication	mitosis	Double-strand break repair
cell cycle	DNA metabolic process	DNA replication initiation	cell division	mitotic checkpoint
DNA replication	response to endogenous stimulus	mitosis	regulation of progression through cell cycle	mitotic sister chromatid segregation
cell cycle phase	regulation of cell cycle	regulation of cyclin-dependent protein kinase activity	DNA repair	G2/M transition of mitotic cell cycle

Table 2.6: Categories for Human E2F1 targets identified by ChIP-chip experiments.

2.7 Summary

The use of GO to analyze large datasets is rapidly becoming a standard procedure in many high throughput experimental studies. The ability to utilize decades of prior work that have been curated into a single database allow researchers to gain initial insight regarding their experiment and can often suggest novel hypothesis for follow-up work [Ihmels et al., 2002, Eisen et al., 1998]. However, in many cases the result of this GO analysis is a long list of significant categories. This makes it hard to interpret the results and determine what the most significantly enriched functions are in the selected set of genes.

In this chapter we described a generative model for identifying a small subset

of categories that, combined, explain the observed set of genes. The algorithm we presented maximizes a global likelihood function to achieve this task. Our results suggest that GenGO is effective in minimizing false positives while at the same time it can accurately balance the set of categories it returns, including both high level and specific categories. GenGO was shown to work very well on both simulated data and real data from a number of different experimental techniques and species. Unlike other methods it does not require an extra step for correcting for multiple hypothesis testing resulting in categories that are both significant and unique.

Chapter 3

Identification of Cell Cycle Regulated Genes

3.1 Overview

The cell is a dynamic system in which gene expressions are highly regulated. During the cell cycle, a cell goes through several stages to replicate its genetic material and organelles, and divide into two daughter cells. As a result, events related to this process are regulated with regard to the cell cycle. Especially, there are many genes expressed periodically, peaking at different stages of the cell cycle.

One of the first questions facing researchers is how to identify these cell cycle regulated genes, or cycling genes. Many methods for identifying cycling genes have been suggested. For example, Spellman et al. [1998] used Fourier transform to identify cycling genes in budding yeast. Wichert et al. [2004] presented statistical methods for identifying periodically expressed genes and applied them (separately) to human and yeast. Lu et al. [2004] and Bar-Joseph et al. [2004] presented methods for deconvolving yeast expression data in order to improve the identification of cycling genes. De Lichtenberg et al. [2005] used scores that look at the amplitude of the expression value peak as well as the peak in the Fourier spectrum around the cell cycle period. All of the above methods identify cell cycle genes by ranking genes in a *single* species according to a score computed from their expression time series.

With microarray data available for more species, researchers have started to study the conservation of the cell cycle expression. Surprisingly, Rustici et al. [2004] found the expression of cycling genes are not well conserved between two closely related species, budding and fission yeast. There could be many reasons for this discrepancy. One possibility is gene expression is not conserved, but an-

other possible explanation is lists for different species are derived using different methods, which have poor consistency.

In this chapter we present a method for combining experiments from multiple species. Our algorithm combines sequence and expression data to identify the set of cycling genes. By considering sequence information we can use paralogs and homologs to overcome noise and cutoff problems in individual species. By using expression data we can detect *functional* conservation, that is, sets of genes that are not only similar in sequence but also similar in function.

We use probabilistic graphical models, and in particular Markov random fields, to combine these data sources. We represent genes as nodes in the graph, with edges corresponding to sequence similarity as determined by a BLAST score. Each node (gene) is assigned an initial score which is determined by the expression experiment. Starting with this score we propagate information along the edges of the graph until convergence. Thus, if a node with a medium score is connected to a set of nodes with high scores, the information from the neighboring nodes can be used to elevate our belief in the assignment of this node, and vice versa.

3.2 The Model

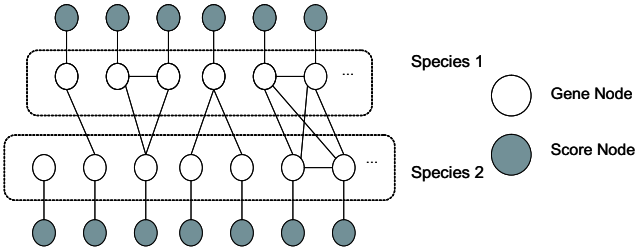


Figure 3.1: A graphical model for two species. Dark nodes are score nodes, representing the score derived from such experiments. The lighter nodes are gene nodes. Gene nodes are connected by edges if their sequence is similar.

We formulate the problem of assigning cyclic status to genes using similarity network models introduced in the previous chapter. There are two types of nodes in the graph we use for this problem (see Figure 3.1). The first represents genes and the second represents expression scores from the related cell cycle experiments. Edges between gene nodes correspond to sequence similarity, and carry a weight which depends on that similarity. These edges are used to capture the conditional dependencies of phylogenetically related genes. All edges between a gene node and its corresponding score node have the same weight and correspond to the gene nodes' potentials.

To generate the edges between potential homologous genes, we run BLAST between all pairs of genes in the two species. We insert an edge between two gene nodes (either belonging to the same species or to two different species) if their BLAST score is higher than a fixed threshold. We use a conservative cutoff such that we are fairly confident that when an edge is added to the graph, the two genes it connects are very likely to be homologous. While we use a cutoff to determine whether we place an edge or not, edges that are present in the graph are weighted based on their BLAST score. The resulting graph comprises of a set of connected components, as demonstrated in the diagram in Figure 3.1.

To represent the latent status of a gene (whether or not it is a cell cycle gene) we associate a hidden variable C_i with each gene node. $C_i = 1$ means that this gene is cell cycle regulated, otherwise $C_i = 0$.

Based on the definitions above, the joint probability distribution over the random variables C_i of this model is defined as follows [Pearl, 1988]

$$L = \frac{1}{Z} \prod_i \psi_i(C_i) \prod_{i,j} \psi_{ij}(C_i, C_j) \tag{3.1}$$

where $\psi_i(C_i)$ is the node potential function (derived from the score node), $\psi_{ij}(C_i, C_j)$ is the edge potential function, and Z is the partition function, i.e. the normalization term. Potential functions capture constraints on a single variable or between a pair of dependent variables. For example, if two gene nodes i and j are connected by an edge with a large weight, it is likely that they are functionally related. Thus, the potential function will penalize assignments that are different in the different nodes (e.g., setting C_i to 0 and C_j to 1). Below we discuss the cycling score and the potential function in detail.

3.2.1 Cycling Scores

A key to our algorithm is to apply a consistent scoring method to all species used. The method we use takes into account both the periodicity and the amplitude of the time series, and use the same method on all datasets.

Once such an expression score has been derived, each score node is assigned the corresponding gene's score, S_i. We assume that S_i is drawn from a mixture distribution. Specifically, we assume two different distributions (for each species): a cell cycle specific distribution, which applies to all genes that are cell cycle regulated, and a null, or background distribution which applies to all other genes.

An important practical issue is to choose the form of the two component distributions of the S_i scores. While the Gaussian distribution has been successfully applied to model expression values, here we are modeling scores that are derived from such values, and not the values themselves. In many cases, such scores are

derived by taking the max value of some transformation. Cell cycle score calculation involves taking the maximum peak of the expression time series or the Fourier transform and the resulting distribution often has a heavy tail and is more appropriately modeled as an Extreme Value Distribution (EVD). This heavy tail property is clearly noticeable in the scores assigned to known cycling genes as can be seen in Figure 3.2.

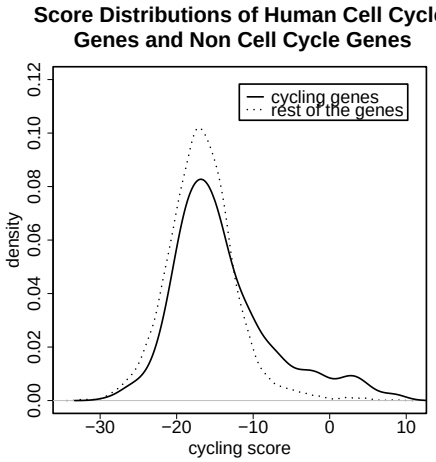


Figure 3.2: Empirical distribution for genes annotated as cycling in GO and the rest of the genes. As can be seen, these two distributions significantly overlap, making it hard to infer cyclic status from the expression score alone. The score distribution of the cell cycle genes has a heavy tail, and looks more like an Extreme Value Distribution than a normal distribution.

The EVD is defined using two parameters: location (a) and scale (b). Its PDF is given by:

$$p(x) = \frac{1}{b} e^{-\exp\left\{\frac{a-x}{b}\right\}} \cdot e^{\frac{a-x}{b}}$$

The location and scale parameters of EVD are similar to the mean and variance parameters of the Gaussian distribution. As in a Gaussian, they control the mode and the spread of the distribution, though they do not necessarily correspond to the mean and variance. Using the EVD mixture model we need to fit four parameters for each species a_0, b_0, a_1, b_1 where

$$\begin{aligned} S_i \mid C_i = 0 &\sim EVD(a_0, b_0) \\ S_i \mid C_i = 1 &\sim EVD(a_1, b_1) \end{aligned}$$

The values of these parameters are fitted to the score distributions using an EM-type algorithm. As with any EM algorithm, the initial guess plays an important role in reaching a good local maximum. To initialize the parameters for the null distribution we permute each of the original time series randomly to simulate the expression levels of non cell-cycle genes. Scores are calculated from these artificial expression data, and are subsequently used to estimate the parameters of the null score distribution. To initialize the score for cell-cycle genes, we compile a list of such genes that appear in the corresponding papers and use the scores of these genes to derive a maximum-likelihood estimate of the parameters.

3.2.2 Node Potential Function

The node potential function is defined using Bayes rule as

$$\begin{aligned}\psi_i(C_i) &= Pr(C_i|S_i) \\ &= \frac{Pr(S_i|C_i)Pr(C_i)}{Pr(S_i|C_i=0)Pr(C_i=0)+Pr(S_i|C_i=1)Pr(C_i=1)}\end{aligned}$$

Using the EVD mixture assumption, the potential function becomes

$$\begin{aligned}\psi_i(0) &= Pr(C_i=0|S_i) = \frac{t_{i0}}{t_{i0}+t_{i1}}, \\ \psi_i(1) &= Pr(C_i=1|S_i) = \frac{t_{i1}}{t_{i0}+t_{i1}}\end{aligned}$$

where

$$\begin{aligned}t_{i0} &= (1-P_c)\cdot\frac{1}{b_0}e^{-\exp\left\{\frac{a_0-S_i}{b_0}\right\}}e^{\frac{a_0-S_i}{b_0}} \\ t_{i1} &= P_c\cdot\frac{1}{b_1}e^{-\exp\left\{\frac{a_1-S_i}{b_1}\right\}}e^{\frac{a_1-S_i}{b_1}}\end{aligned}$$

and P_c is a prior probability for cycling genes in the species to which i belongs.

In practice, we require $b_0 = b_1$ so that the two score distributions have a similar spread. This guarantees that the posterior score will have the same ranking as the expression scores when there are no edges in the graph.

3.2.3 Edge Potential Functions

Our edge potential functions capture the a-priori functional similarity between gene pairs. This is based on our assumption regarding evolutionary conservation of gene functions, namely, that genes that are highly similar in sequence are likely to be similar in function. We use BLAST [Altschul et al., 1990] to determine sequence

similarity. As mentioned earlier, we do not transform these BLAST scores into binary features. Rather, we use the similarity score to determine the edge potential which penalizes contradictory assignments. The penalty is proportional to how close the two genes' sequences are.

For each query sequence, the BLASTALL program returns an E-value and a bit score S. The relation between them is $E = mn2^{-S}$ where m is the length of the query sequence and n is the length of the genome of the second species. Note that bit scores are not "symmetric" as they depend on the total genome length. To overcome this, and generate a single similarity score for pairs of genes we set the weight on edge (i, j) to

$$w_{ij} = \frac{1}{2}(b_{ij} + b_{ji})$$

where b_{ij} is the BLAST bit score of gene i against gene j. Using $w_{i,j}$ we define the edge potential as

$$\psi_{ij}(C_i, C_j) = 2^{-\lambda w_{ij}(C_i - C_j)^2}.$$

This potential function penalizes assignments that do not agree between connected nodes. λ is an externally specified parameter that controls the impact of edge potentials relative to the node potentials.

3.3 Learning the Parameters of Our Model

The model parameters we need to learn are the score distribution parameters of every species. We learn the score distribution parameters (a_0, b_0, a_1, b_1) in an iterative manner using an EM-style algorithm. We start with an informative guess for the score parameters, as mentioned above. Based on the score distributions we determine a posterior assignment to nodes using belief propagation, as we discuss below. Following convergence of the belief propagation algorithm we use the (soft) label assignments to update the score distribution parameters. We then repeat these steps by performing belief propagation again based on the updated score distributions and so forth until both the label assignment and score distribution parameters do not change anymore.

3.3.1 Iterative Step 1: Inference by Belief Propagation

To infer the node status variables C_i, we need to compute the marginal posterior label distribution on each gene node. This posterior is hard to compute directly because of the intractable normalization term Z in Formula (3.1). Fortunately, for these types of graphical models, we can use a standard belief propagation algorithm

for inference avoiding the direct calculation of the Z term [Pearl, 1988]. Note that our graph is loopy and thus the belief propagation algorithm is not guaranteed to converge to a global maximum. Still, as was shown in Yedidia et al. [2003], in practice these algorithms achieve good results in loopy networks as well.

The belief propagation algorithm consists of two steps: 'Message passing', where each node sends its current belief to all its neighbors, and 'belief update', where nodes update their belief based on the messages received. In our case the messages depend on the node's expression score and the belief of genes that are similar in sequence. The algorithm is summarized below.

1. 'Message passing'. The messages sent by node i to node j about its belief in an assignment of 1 to j is :

$$m_{i,j}(1) \leftarrow \sum_{k=0,1} (\psi_i(k)\psi_{ij}(k,1) \prod_{n \in N(i) \setminus j} m_{n,i}(k))$$

   Where $N(i)$ is the set of neighbors of node i in the graph. Intuitively, this message informs j about i's agreement with an assignment of 1 to j. In order to determine this, i takes into account its own belief (from its score node), the strength of the edge between i and j and the belief of i's neighbors about the right assignment to i. For the belief in a 0 assignment we simply replace every 1 with 0 in the above equation. Note that the weighting parameter λ is already incorporated into the edge potential function and so it is incorporated into the message as well.

2. 'Belief update'. The belief of i in an assignment of 1 is computed by setting:

$$b_i(1) = (1/v)\psi_i(1) \prod_{j \in N(i)} m_{j,i}(1)$$

   where v is a normalization constant to make beliefs sum to 1. As can be seen, i's belief depends on both its original score and the messages it received from its neighbors about what they 'believe' should be assigned to i.

3.3.2 Iterative Step 2: Updating the score distribution

Using the belief computed in the inference step, we update the score distribution parameters. Our goal is to maximize the auxiliary function $Q(\Theta, \Theta^{(g)})$, which is defined as the expected log likelihood of the complete data over the observed scores given the parameters $\Theta^{(g)} = (a_0^{(g)}, a_1^{(g)}, b^{(g)})$ at the g'th iteration.

We were unable to find a reference for deriving update rules for the EVD mixture distribution. We have thus derived these ourselves. In general, to derive an

update rule for this distribution we need to simplify the Q function and separate the parameters into two terms which can be maximized independently. If we require that $b_0 = b_1$, then for each species we have three parameters: two location parameters a_0 and a_1 and one scale parameter b. We can find the location parameters that maximize Q easily if we know b, but there is no close form solution for b. However, we can use numerical methods to solve for b. The final update rules for each species are as follows

$$a_l^{(g+1)} = \frac{1}{\beta} \log \frac{\sum_{i=1}^{N} P_{il}}{\sum_{i=1}^{N} e^{-\beta S_i} P_{il}}, \quad l = 0, 1$$
$$b^{(g+1)} = \frac{1}{\beta}$$

where N is the number of genes in that species, P_{il} represents $p(C_i = l|S_i, \Theta^g)$, $l = 0, 1$, and β is the root of the equation:

$$\frac{1}{\beta} = \frac{\sum_{l=\{0,1\}} \sum_{i=1}^{N} S_i P_{il}}{\sum_{l=\{0,1\}} \sum_{i=1}^{N} P_{il}} - \sum_{l=\{0,1\}} \left[\sum_{i=1}^{N} P_{il} \frac{\sum_{i=1}^{N} e^{-\beta S_i} S_i P_{il}}{\sum_{i=1}^{N} e^{-\beta S_i} P_{il}} \right] \Big/ \sum_{l=\{0,1\}} \sum_{i=1}^{N} P_{il} \qquad (3.2)$$

Equation (3.2) can be solved using linear line search since the reasonable range of β is not large. Note that the Newton-Raphson method does not work here, because the solution is very close to the local extrema of the function.

Our algorithm is summarized in Table 3.1.

3.4 Identification of Conserved Cell Cycle Genes

3.4.1 Simulated Data

To test our model using simulated data we first generated the graph structure from the two species as discussed before. We then generated labels (i.e. cycling or not) for nodes in the graph using a Gibbs sampler method that took into account previously assigned neighboring nodes when assigning labels to individual nodes.

After generating the labels we assigned scores to nodes. We used two (overlapping) score distributions, one for the nodes with $C_i = 1$ and the other for those with $C_i = 0$. In all experiments we used a fixed distribution for one species. However, each experiment used a different distribution for the second species. These distributions varied in their separability, ranging from highly separable to highly

Table 3.1: Algorithm for combining microarray expression data from multiple species.

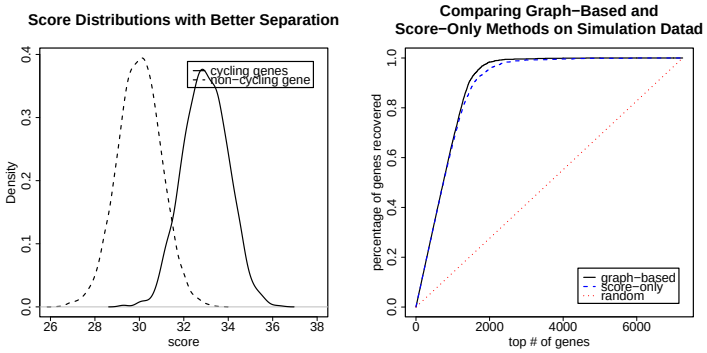
Input: 1. For each gene, expression score S_i 2. Graph structure (edge weights) **Output**: For each gene its posterior cycling status, C_i
Initialization: For each species compute estimates for a_0, a_1 and b using permutation analysis and original lists
Iterate until convergence: 1. Carry out Belief Propagation to determine a posterior C_i for each gene 2. Use the computed posterior to recompute the EVD parameters for the score distribution in each species

overlapping (see Figure 3.3). We have next hidden the node assignments, and used our algorithm to infer these assignments. We repeated this process 10 times for each set of score distributions.

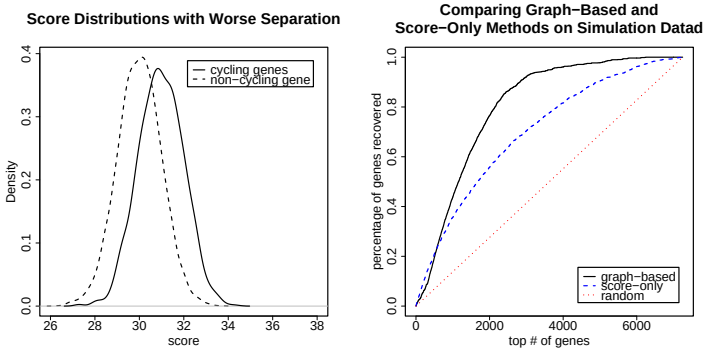
Figure 3.3 presents the results of two of these experiments. As can be seen, by relying on the graph structure we were able to improve the recovery of the true label assignments when compared to label assignments that are based on a cutoff of the score alone. As the separation between the two distributions became smaller the difference between the two methods became more apparent. For the less separable distributions our algorithm performed much better than the score only method by relying more heavily on the distribution of the other species.

These results indicate that under the evolutionary assumptions we stated in the introduction, our algorithm can improve the assignment of cycling genes and correctly recover more such genes.

It is worth noting that exact inference in general graphs is a NP-hard problem. Belief propagation is an efficient algorithm for approximate inference on graphs with loops. As a result, in addition to the noise in microarray measurements, the computed posterior probabilities are also affected by how good the approximation is.



(a) Score distribution (b) Recovery rate



(c) Score distribution (d) Recovery rate

Figure 3.3: Simulation results. 20% of the nodes were labeled with 1 and the rest were labeled with 0. (a) Score distribution and (b) Recovery rate for a well separated distribution. Both score based (dashed line) and graph based (solid line) methods were able to correctly recover the node assignments. (c) Score distribution and (d) Recovery rate for an overlapping score distribution. Note that while our graph based method can still achieve good precision and recall the score based method does significantly worse, especially for the higher recall rates (above 40%).

3.4.2 Identification of Cell Cycle Genes

To date, cell cycle expression was measured in more than six species. As mentioned above, the two most studied species are budding yeast and humans. Both provide access to a number of different validation sets, and are thus useful for comparison of our algorithm and score based methods.

We downloaded expression data from the corresponding websites for the budding yeast [Spellman et al., 1998] and human [Whitfield et al., 2002] cell cycle

papers. All protein sequences for genes in these species were downloaded from the NCBI ftp server (http://ftp.ncbi.nlm.nih.gov). We used BLASTALL [Altschul et al., 1990] to score all pairs of genes in both species.

Identifying Cycling Human Genes

To test the success of our algorithm for the task of identifying cycling human genes we used the GO human annotations. Of the 7254 human genes in the dataset we used, 498 were annotated by GO as cycling. We first ranked human genes using expression scores and the naive method mentioned above. Next, we ranked them using the posterior score computed by our algorithm.

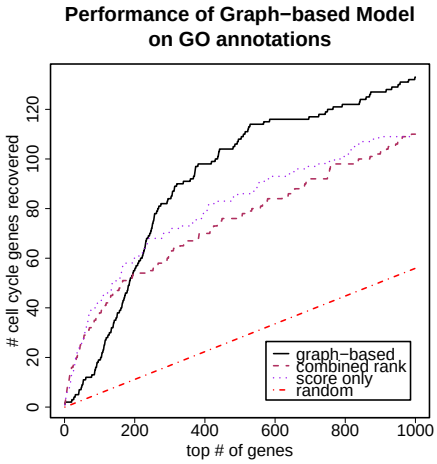


Figure 3.4: Identification of Human Cell Cycle Genes. The Y axis is the number of GO annotated human cell cycle genes in the top 1000 genes with highest posteriors. Our method (solid line) performs better than the score only method (dashed line) and the naive method for combining sequence and expression data (dotted line). Specifically for lower score thresholds our method achieves an improvement of over 20% over both other methods in terms of the number of accurately recovered cell cycle genes.

Figure 3.4 presents the precision recall curve for GO annotated cycling genes for the top ranked 1000 human genes. Based on the analysis in the original paper [Whitfield et al., 2002], roughly 1000 genes are determined to be cycling, which is why we focus on the top 1000. As can be seen, all three methods perform substantially better than a random ordering (dashed-dotted curve). Comparing our

method with a score based method we see that while at the very high expression score (bottom left) we do slightly worse, overall, and in particular for lower scores our algorithm provides results that are superior to score based methods. Specifically, for the top 1000 genes our algorithm was able to recover 23% more genes (135 vs. 110) when compared to both, the score only method and the naive method for combining sequence and expression data.

Note that while we relied on the GO list for this analysis, it is not complete. It is possible that there are many cycling genes which are not on that list. Thus, the recall rate is probably much higher than the one we report here. As we saw in Figure 3.2, there is substantial overlap between the expression score distributing of genes annotated as cycling and genes those do not belong to this category, making it hard for a score only method to identify a large set of cycling human genes. In contrast, our graph based method was able to partially overcome this problem by relying on the graph neighborhoods.

While our algorithm has achieved better performance, it also makes some errors. We show one of the false negatives, budding yeast CDC5, in Figure 3.5. From the expression time series, CDC5 is clearly a cycling gene. Its posterior probability falls below the threshold because of non-cycling homologs in its graph neighborhood. However, we should note that these cases are rare. For example, only two obviously cycling genes are not included in our list cycling genes in budding yeast, which means even if the false negatives is ten times the number, the false negative rate would still be less than 3%. On the other hand, it is harder to determine false positives because we don't have a list of non-cycling genes. We expect false positives would be less of a problem because our algorithm only elevates the posterior of genes with at least a border line cyclic score. In fact, the rank of most genes doesn't change much when we compare the result from our algorithm with the result based on cyclic scores alone (Figure 3.6).

Convergence of Loopy Belief Propagation

In general, belief propagation is not guaranteed to converge on a graph with loops. Several sufficient conditions for the algorithm to converge to a unique fixed point are known [Tatikonda and Jordan, 2002, Ihler et al., 2006, Mooij and Kappen, 2007]. For example, one sufficient condition that guarantees the convergence of Loopy belief propagation is

$$\max_t \sum_{u \in N(t)} \log d(\psi_{ut}) < 1 \tag{3.3}$$

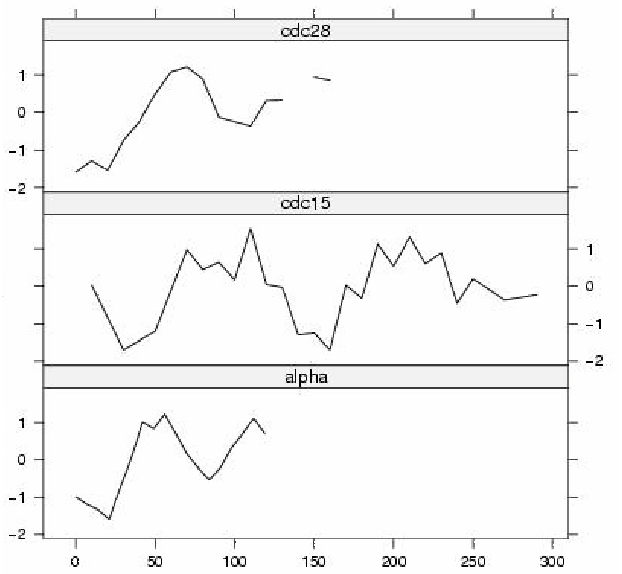


Figure 3.5: Example of a false negative cycling gene. Budding yeast CDC5 is a cycling gene, but was not recovered by our algorithm.

where

$$d^2(\psi_{ts}) = \sup_{a,b,c,d} \frac{\psi_{ts}(a,b)}{\psi_{ts}(c,d)}$$

is the dynamic range measure of potential function ψ_{st}. For our model, the dynamic range measure of edge potential function ψ_{st} is simply $\exp\{\lambda w_{st}\}$, where w_{st} is the edge weight, and λ is a non-negative hyper-parameter controlling how much a gene's cycling status is affected by its homologs' status. It can be seen that the graph we construct from sequence similarity satisfies the sufficient condition when λ is small (close to zero). Using condition in Eq 3.3, the algorithm is guaranteed to converge on the graph for four species when $0 \leq \lambda <\sim 7.7 \times 10^{-6}$. However, this condition is too conservative. Empirically, Loopy belief propagation converges on this graph when $\lambda < 0.01$, but may fail to converge with a larger λ. To put it into perspective, the λ we learned and showed to improve prediction accuracy is around 0.0005, which is within the range where Loopy belief propagation converges.

Comparison with Graph Cut Algorithm

Graph Cut is another popular method for learning labels in Markov random fields [Boykov et al., 1998, 2001], which can use both labeled and unlabeled data [Blum

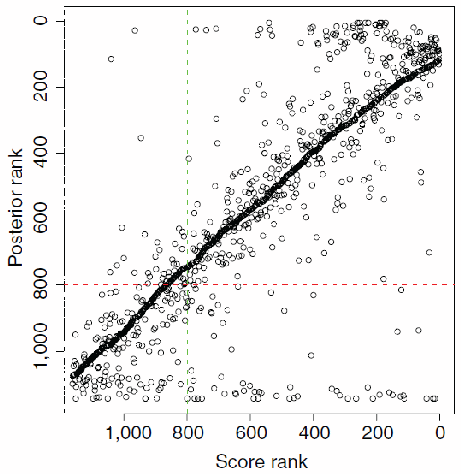


Figure 3.6: Comparison of expression score ranks and posterior ranks. The expression score rank and posterior rank for budding yeast genes. The x-axis is the expression score rank (the lower the rank the more cyclic the gene is determined to be by the scoring method) and the y-axis is the rank based on our method (again, the lower the better). As can be seen, the ranks for most of the genes do not change much. The red dashed line represents the posterior threshold used to select cycling genes, and the green dashed line is the corresponding threshold if only expression scores are used. Almost all genes that are elevated by our method to a cyclic status have a rather high cyclic expression score (though some are not as high as the cutoff for score alone, which is where the two methods differ).

and Chawla, 2001]. Similar to belief propagation, Graph Cut is an approximate inference algorithm that finds assignment of node labels that maximizes the likelihood. It works by looking for a set of edges with minimal total weight (a minimum-cut) that separates the positively labeled and negatively labeled nodes. After removing the cut, an unlabeled node is assigned a positive (negative) label if it is reachable from a positive (negative) node.

Tappen and Freeman [2003] compare Graph Cut and belief propagation on stereo vision problems, and in their study the results from both algorithms are comparable. Graph Cut is able to achieve lower energy on those problems than belief propagation, but empirically it does not imply better performance in recovering the ground truth. In another study by Mahamud [2006], the author shows belief propagation can achieve better performance. It is interesting to see whether Graph Cut can achieve better performance for our model. Both Graph Cut and belief propagation are polynomial-time algorithms. Standard max-flow algorithms can be used

to solve for th minimum-cut problem, and the worst case runtime complexity is $O(N^3)$, where N is the number of nodes in the graph [Boykov and Kolmogorov, 2004]. In contrast, the time complexity of belief propagation is $O(TN)$, where T is the number of iteration to converge [Mahamud, 2006]. Empirical study shows the speed of Graph Cut algorithms is efficient and comparable to belief propagation on vision problem where the Markov random field models are grid-structured. However it is not clear how fast the two algorithms can run on our problem.

For comparison, we apply the standard max-flow algorithm [Ford and Fulkerson, 1956] to learn the human cell cycle genes, using budding yeast cell cycle genes as labeled data. Because there is no labeled data for non-cycling genes, we randomly choose 10% genes from unlabeled genes to use as negative training data. From the results, we can see the Graph Cut algorithm can achieve similar precision and recall, but it runs much slower than the belief propagation algorithm.

Algorithm	Recall	Precision	Time
Graph Cut	0.14	0.26	37 min
Belief Propagation	0.14	0.27	1 min 22 sec

Table 3.2: Comparison of Graph Cut and belief propagation.

3.4.3 Identification of Groups of Orthologous Cycling Genes

By incorporating information from sequence similarity, we are able to identify a more consistent set of cycling genes. To further discover groups of orthologous cycling genes across species, we apply the Markov clustering (MCL) algorithm [Enright et al., 2002] to the graph of cycling genes. MCL has been shown to work well in detecting protein families, and it can handle the presence of multi-domain proteins in the graph. The resulting groups provide candidates for further conservation analysis in the next section. At the same time, by looking at the graph neighborhood represented by these groups, one can easily see the power of our algorithm to recover cycling genes with relatively weak expression scores. One such graph neighborhood is shown in Figure 3.7. Fission yeast nda3, a microtubule component, is a known cell cycle gene [Javerzat et al., 1996] . On the top of Figure 3.7 we plot the graph neighborhood of nda3. As can be seen, it contains many known cycling genes from the four species. On the bottom we plot the expression of nda3 in 8 different fission yeast cell cycle datasets. As can be seen, in at least some of these conditions nda3 seems to be cycling (the right panel). However, either because its expression levels are low in the other experiments or because of other experimental problems, it does not seem to be cycling in the other conditions. Using expression data alone, we would not assign a cyclic status to this gene. However, because of

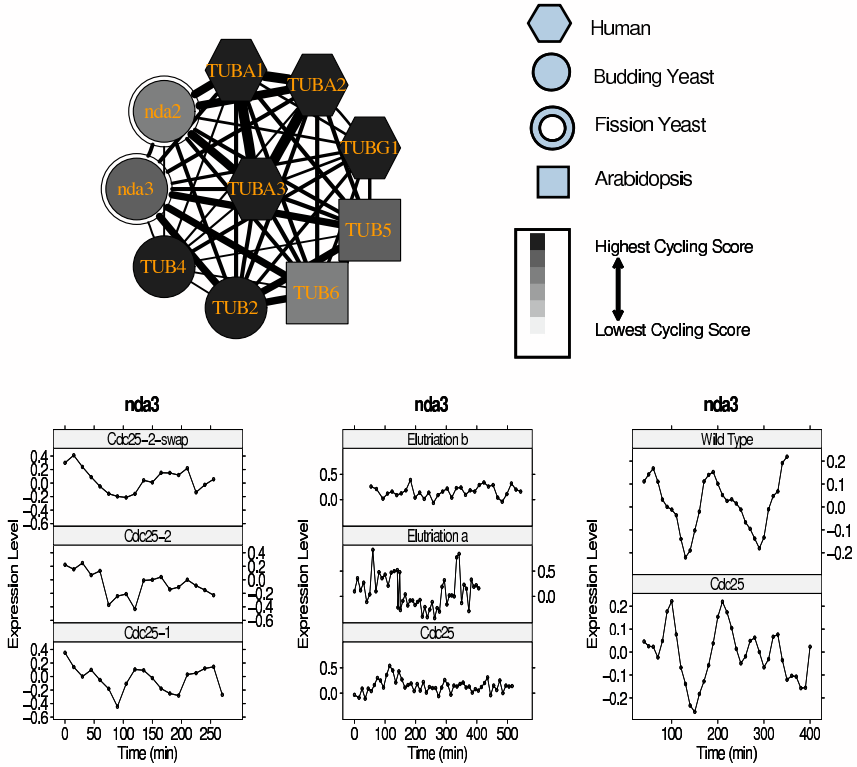


Figure 3.7: Microtubule component clique and expression profiles for fission yeast Nda3 in eight experiments [Rustici et al., 2004, Oliva et al., 2005, Peng et al., 2005]. Nda3, a known cell division gene [Javerzat et al., 1996], obtains a high cycling score but is not one of the 600 top cycling fission genes based on expression analysis. Using our method its score is correctly elevated due to its sequence similarity to high scoring genes.

3.5 Biological Analysis of Conserved Cycling Genes

We applied our algorithm to the expression time series of budding yeast, fission yeast, *Arabidopsis*, and humans, using data from [Spellman et al., 1998, Rustici et al., 2004, Menges et al., 2002, Whitfield et al., 2002]. After obtaining the lists of cycling genes in each species, we divided them into several groups based on

whether a gene is specific to only one species, or is conserved in two, three, or all of the four species.

We compare conserved cycling genes with species-specific cycling genes, as well as the existing lists of cycling genes. In the following discussion, we denote the set of human cycling genes conserved in all four species by CCC4$_{human}$ and those conserved in three species (humans and the two yeasts) by CCC3$_{human}$, etc.

3.5.1 GO Analysis of Conserved Cell Cycle Genes

The CCC3 list gives us our first look at the conserved core of periodically transcribed genes across evolution. Even though CCC3 contains relatively few genes (0.4% to 1.3% of the total number of genes for each species) many of these genes play a role in key processes required for growth. Using the enrichment analysis tool (GenGO) developed in Chapter 2, we identified categories that were enriched in this set. For budding yeast these categories include "mitotic cell cycle" (p-value = $4 * 10^{-17}$), "DNA replication" (p-value = $2 * 10^{-13}$) and "chromatin assembly/disassembly" (p-value = $3 * 10^{-12}$). Similar enrichments were found for human conserved cycling genes and for fission yeast. For example, "mitotic cell cycle" (p-value = $3 * 10^{-15}$), DNA replication (p-value = $1 * 10^{-}14$), and "spindle organization and biogenesis" (p-value = $1 * 10^{-}7$) are enriched in humans, and cell cycle (p-value = $10^{-}9$), "chromatin assembly/disassembly" (p-value = 10^{-9}) are enriched in fission yeast.

3.5.2 Interaction between Cycling Yeast Genes and Key Transcription Factors

In eukaryotic cells, gene expression is regulated by transcription factors, a large class of proteins that are able to bind to DNA. For some species, researchers have found transcription factors that play an important role in regulating periodic gene expression. As a result, cycling genes are more likely to be regulated by these transcription factors. We used a dataset for protein-DNA binding [Harbison et al., 2004] to compare our budding yeast results with the original list of Spellman $et\ al$ which was based on score alone. We extracted the binding information (p-value $<$ 0.005) for the nine transcription factors that have been previously shown to play key roles in regulating cell cycle progression [Simon et al., 2001] (Figure 3.8 (a)). We found 2.5% more interactions between these nine TFs and the top 800 genes on our list when compared with the Spellman list (621 vs. 606, note that a gene could be counted multiple times if more than one TF interacts with it). We also tried the binding information with a stricter p-value ($<$ 0.001), where our method

also found slightly more interactions (477 vs. 474). While this improvement is far less dramatic than the results presented for the human data, it still implies that our method can improve cell cycle assignment even for high quality datasets, like the yeast cell cycle expression data [Wichert et al., 2004].

We also compared the DNA-protein interaction between the cycling genes and two human transcription factors known to involved in cell cycle control [Ren et al., 2002] (Figure 3.8 (b)). In both cases, the percentage of genes bound by the transcription factors is significantly higher for the conserved list than for the list in Whitfield et al. [2002], and both are higher than the human-specific list.

3.5.3 Gene Expression in G0 Phase or Developmental Arrest

In contrast to normal dividing cells, cells in G0 phase do not grow or divide. In this phase, genes that are part of the cell cycle machinery are probably expressed at a lower level or not expressed at all. We use the data in [Gasch et al., 2000] to test this idea. As we show in Figure 3.8 (c), after entering G0 phase, the average expression level of conserved cycling genes becomes significantly lower than that of the budding yeast specific genes, while the list in [Spellman et al., 1998] lies somewhere in between. This finding supports the view that the core cell cycle machinery enters a low activity state while species-specific cycling genes participate in other pathways, e.g. metabolic pathways, necessary for maintaining the living state. It is also possible that the latter genes are responsible for reactivating the core cell cycle machinery to enter the mitotic cell cycle again.

For *Arabidopsis*, we test the similar idea using expression data in an *Arabidopsis* mutant whose flowers enter developmental arrest following stage 12 [Nagpal et al., 2005] (Figure 3.8 (d)). The cells in the stem of the mutant are used as the control. It can be seen, following the developmental arrest, the average expression level of conserved cycling genes goes down while it remains the same in normal dividing cells in the stem.

3.5.4 Protein-Protein Interactions Between Cycling Genes

We further show that there are much more protein-protein interactions among conserved cycling genes than average cycling genes, and so they are more likely to work together in a few modules than spreading over a large set of modules. We use large scale protein-protein interaction data sets in [Gavin et al., 2006, Krogan et al., 2006] for budding yeast and data in [Rual et al., 2005] for human cells. To generate an empirical distribution of interaction numbers, we randomly draw subsets

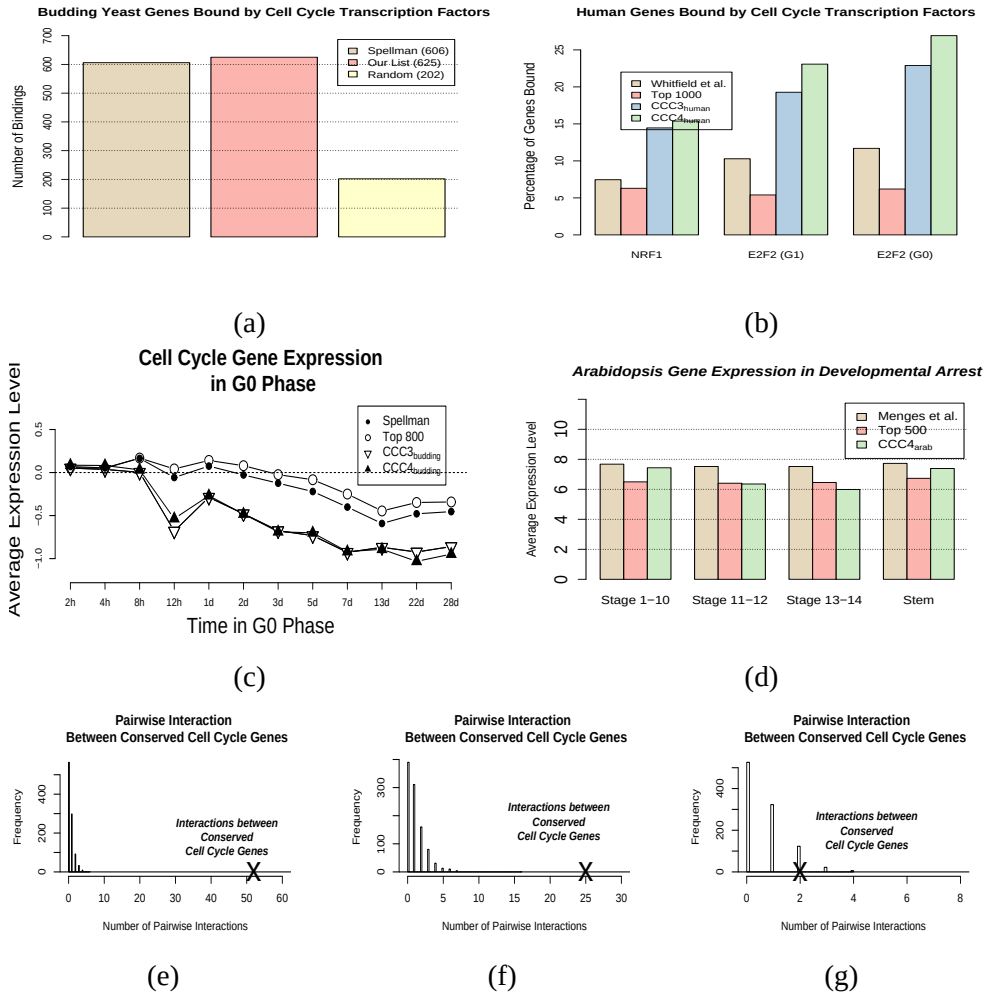


Figure 3.8: (a) shows the number of interactions between budding yeast cycling genes with nine key transcription factors involved in cell cycle control [Harbison et al., 2004]. Our list of cycling genes has slightly higher number of binding than those in [Spellman et al., 1998]. (b) shows the percentage of cycling human genes bound by two cell cycle transcription factors [Ren et al., 2002]. Conserved cycling genes have a significant higher percentage of bindings than that in [Whitfield et al., 2002], which is in turn higher than that in human specific cycling genes. (c) shows the average expression level of cycling budding yeast genes in phase G0 [Gasch et al., 2000]. Conserved cycling genes have significant lower average expression level than those in Spellman et al., while budding yeast specific cycling genes have a higher expression level. (d) Flower cells of Arabidopsis *arf-6 arf-8* mutant show developmental arrest at stage 12, while cells in stem are normal [Nagpal et al., 2005]. We compare the average expression level of conserved cycling genes and those in [Menges et al., 2002]. The conserved genes have a lower expression level during developmental arrest. (e), (f), and (g) show the number of protein protein interactions between conserved cycling genes, comparing with the number of interactions within a random set of cycling genes [Gavin et al., 2006, Krogan et al., 2006, Rual et al., 2005]

from the list of all cycling genes, and count the number of interactions within each subset. It can be seen, for budding yeast there are significantly more interactions between the conserved set (Figure 3.8 (e) and (f)). While the interactions among conserved human genes are not as significantly enriched (Figure 3.8 (g)), we note that there are much fewer interactions in the human data set and there might be more unknown interactions.

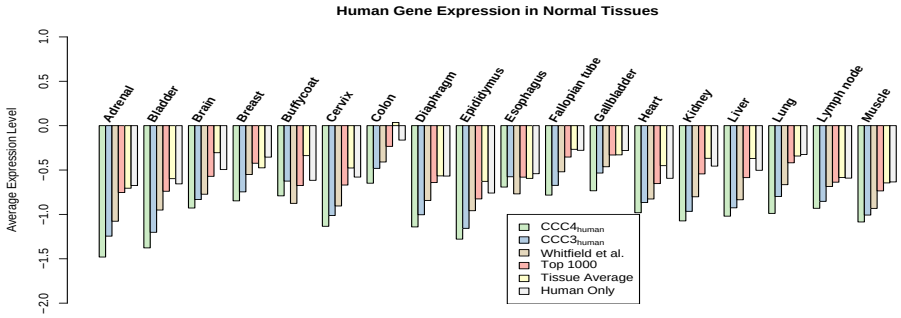
3.5.5 Gene Expression in Human Normal Tissues and Cancer Cell Lines

We compared the average gene expression level in human normal tissues, where most cells have stopped growing. We used the data set in Shyamsundar et al. [2005], and tested the significance of differences by t-test. For all tissues in the data set, we find that the expression of conserved cycling genes is lower than that of human-specific cycling genes, and genes in Whitfield et al. [2002] lies somewhere in between. In 22 out of 36 tissues the difference between the conserved set and genes in [Whitfield et al., 2002] is significant (p-value ≤ 0.05), and the difference between the latter and human-specific genes is significant in almost all tissues (33 out of 36, p-value ≤ 0.05) (Figure 3.9 (a), (b)).

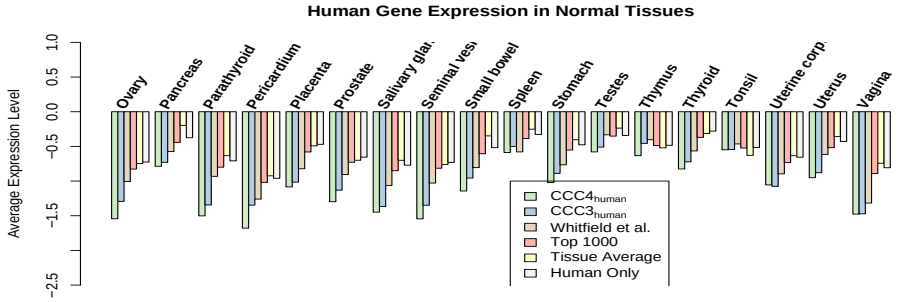
In contrast to cells in normal tissues, cancer cells usually divide aggressively and cell cycle regulated genes are expected to be expressed in higher levels. We used data for two colon cancer cell lines from Provenzani et al. [2006] and found conserved genes are indeed expressed in higher levels than those in Whitfield et al. [2002]. We have found further support when comparing expression levels of an asynchronous cell population where cells are dividing, and that of human cells in G0 phase, where cells have stopped growing. (Figure 3.9 (d)). The expression level of the conserved set is high in the former and dips in the latter population, which makes it a better indicator of the cell cycle state of the population.

3.5.6 Percentage of Conserved Cycling Genes

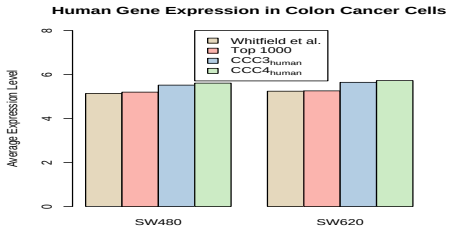
Figure 3.10 presents the number of conserved genes for the different evolutionary distances represented in our datasets. About 21% of the budding and fission yeast cycling genes reside in cliques containing genes from these two species (CCC2). When adding human genes, roughly 10% of cycling yeast genes and 8% of cycling human genes are included in such cliques (CCC3). Finally, between 5% and 7% of cycling genes in all four species are conserved in sequence and expression (CCC4). We note that although our original sequence similarity criterion was



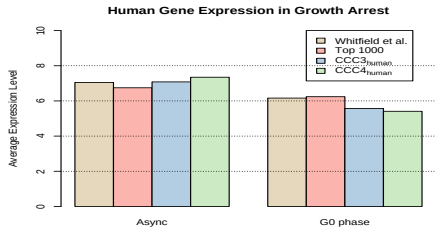
(a)



(b)



(c)



(d)

Figure 3.9: (a) and (b) show the average expression levels of cycling genes in normal human tissues [Shyamsundar et al., 2005]. In all cases, conserved cycling genes have significant lower expression levels than human-specific cycling genes. (c) shows the average expression level in two colon cancer cell lines [Provenzani et al., 2006]. Conserved cycling genes have a higher average expression level than those in [Whitfield et al., 2002]. (d) shows the average expression levels in asynchronous cell population and the G0 phase [Cam et al., 2004].

based on BLAST e-values, following the clique analysis the resulting sets are in very good agreement with curated homology databases Penkett et al. [2006]. For example, 82% of budding yeast genes in CCC2 have a curated fission yeast homolog in CCC2. Similarly, 82% of fission yeast genes in CCC2 have a curated budding yeast homolog in CCC2.

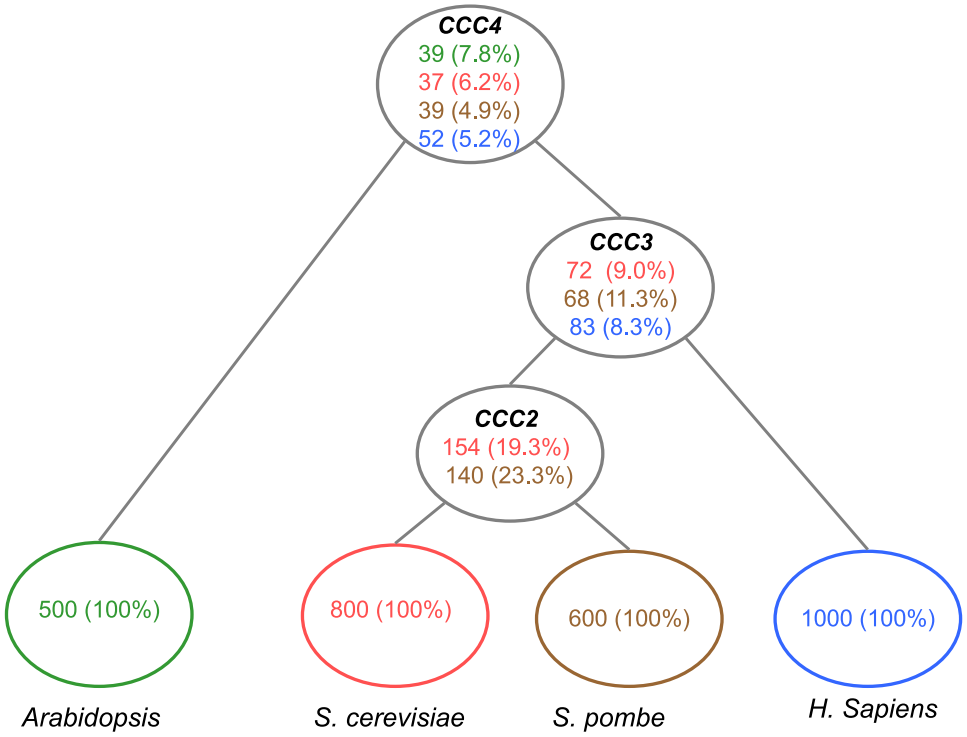


Figure 3.10: Conservation of cycling genes: percentage of conserved cycling genes in the four species.

3.5.7 Motif Analysis for Budding and Fission Yeast Genes

To further validate our findings of a large overlap between the cycling genes in the two yeast species, we turned to motif analysis. Several transcription factors are conserved between budding and fission yeast [Bähler, 2005]. A possible explanation for expression conservation (or lack thereof) is in the conservation (or lack of conservation) of a binding motif for these cycling genes.

We started by looking at genes bound by the budding yeast factor Swi6, which regulates transcription at the G1/S transition [Breeden and Nasmyth, 1987]. We ex-

tracted three lists for this factor. The first, denoted BY6, contained cycling budding yeast genes in CCC2 determined to be bound by Swi6 [Harbison et al., 2004]. The second list, denoted FY6C, contained fission yeast genes that both were in CCC2 and had homologs in BY6. These genes were determined to be cycling and conserved by our method. The third list (FY6NC) contained non-cycling fission yeast genes with cycling budding yeast homologs bound by Swi6. This latter list serves as a negative control because it contains genes that have lost their cycling status between the two species. Four motif finders were run on each dataset; SOMBRERO [Mahony et al., 2005b,a], BioProspector [Liu et al., 2001], Consensus [Hertz et al., 1990], and AlignACE [Roth et al., 1998] (see Materials and methods, below, for details). All four motif finding algorithms were able to identify the Swi6 motif in BY6 and FY6C, indicating that this motif is conserved between the two species, at least for some of the conserved cycling genes. In sharp contrast, none of these motif finders was able to identify the Swi6 motif in the upstream regions of genes in FY6NC.

We have extended the motif analysis discussed above to study ten additional transcription factors that were determined to play a key role in regulating cycling genes in budding yeast [Pramila et al., 2006, Simon et al., 2001]. For each of these factors we extracted all cycling budding yeast genes determined to be bound by this factor [Harbison et al., 2004] and their fission yeast homologs. As we did for Swi6, we further divided the fission yeast genes into two sets; the first contains fission yeast genes in CCC2 and the second (a negative control list) contains non-cycling fission yeast homologs of cycling budding yeast genes. Next, we ran the four motif finders on each dataset.

The results are presented in Table3.3. Here we report on the number of motif finders that identified the correct motif for each factor and on the percentage of genes in the set that contained this motif. Similar to the results obtained for Swi6, the other two G1/S factors, namely Swi4 and Mbp1, exhibit the optimal motif conservation pattern; the expected motifs are found in both the fission yeast cell cycle genes and the positive control of conserved budding yeast cell cycle genes, but are not found in the negative control set of non-cycling fission yeast genes. For G2/M, the Fkh2 sets display similar, although less significant, pattern (two of four motif finders identified the correct motif for the cycling set). However, Fkh1 and Fkh2 motifs also appear, although less strongly, in the negative control sets. In total, FKH-like motifs are present in eight of the 11 negative control datasets. The M/G1 phase analysis is complicated by small dataset size. This may result from the lack of conservation between the two species for this phase [Bähler, 2005]. As a result, motif match for this set is either weak (Swi5) or nonexistent (Mcm1 and Yox1).

Budding yeast phase	Transcription factor	Fission yeast cell cycle genes	Negative control (fission yeast non-cell-cycle genes)	Positive control (conserved budding yeast cell-cycle genes)	Extended positive control (all budding yeast CC genes)
G1/S	Swi4	4	0	4	4
	Swi6	4	0	4	4
	Mbp1	4	0	4	4
G2/M	Fkh1	0	2	1	3
	Fkh2	2	2	1	2
	Ndd1	0	0	4	4
M/G1	Mcm1[a]	0	0	3	4
	Ace2	4[b]	0	0[b]	4
	Swi5	~ 2[b]	0	~ 2[b]	1
	Yox1	0[b]	0[b]	3[b]	3
	Yhp1	0[b]	0[b]	1[b]	~ 1[b]

Table 3.3: Motif analysis of the conserved cycling genes in budding and fission yeast. For each set and each factor we list the number of motif finders (up to four) that identified the correct motif. Each motif finder often recovers multiple correct motifs, and each motif is associated with a list of predicted instances in promoter regions. We report the percentage of promoters that contain instances predicted by at least one-third of the correct motifs. The first and third columns are the CCC2 genes in budding and fission yeast, respectively. The second column is non-cycling fission yeast genes with homologous cycling budding yeast genes. See Additional data file 3 for further details. [a] Mcm1 regulates genes in G2/M and M/G1. [b] These datasets contain ten genes or fewer. $\sim$, weak matches to the known motif.

3.5.8 Essentiality of Conserved Cycling Genes

Finally, we show that conserved cycling genes are more likely to be essential genes, without which the cell is unable to survive or to proceed through the cell cycle normally. We carry out the analysis on both budding yeast and human cells.

Percentage of essential budding yeast genes. For budding yeast, we use the knockout data from the *Saccharomyces* Genome Deletion Project consortium [Winzeler et al., 1999]. We compared several sets, including the set of all cycling genes, the set of conserved cycling genes, and the set of cycling genes with homologs (regardless of cycling or not) in other species. The last set is chosen to show how much information we can gain by incorporating microarray expression data. It can be seen that the set of cycling genes conserved in four species ($CCC4$) has the

highest percentage of essential genes (41.2%), much higher than any other sets in the comparison (Figure 3.11).

Percentage of essential human genes. For humans, we base our analysis on large-scale RNAi knock-down experiments [Mukherji et al., 2006]. In their study, each of the $24,373$ predicted human genes were knocked down, covering $> 95\%$ of all protein-coding genes in the human genome. Analysis of the resulting cells shows that depletion of $1,152$ genes strongly affects the normal progression of the cell cycle. We use this list of essential genes, and carry out the same analysis as we did in the budding yeast. As we can see in Figure 3.11, the sets of conserved cycling genes again have the highest percentage of essential genes (15.7% for $CCC3$ and 17.3% for $CCC4$). In contrast, the full set of cycling genes has similar percentage of essential genes to a random set of genes of the same size. Combining expression data and sequence data in a naive way only slightly increases the percentage of essential genes to 9.8%.

Together we show that, by combining sequence data and microarray expression data, we are able to identify a more coherent set of cycling genes.

3.6 Summary

By combining information from sequence and expression, we were able to identify a large set of genes as conserved in both sequence and cycling status between four different species: budding yeast, fission yeast, humans, and Arabidopsis.

A number of previous studies comparing cycling gene lists derived independently for each species concluded that only a small number of genes are conserved between these species. For example, Rustici et al. [2004] concluded that only 5% to 10% of cycling budding yeast genes have a cycling homolog in fission yeast. Jensen et al. [2006] identified only five orthologous groups to be conserved between the four species (about 1% of the cycling genes). However, due to experimental noise and difference in computing cyclic scores, direct comparison of lists cycling genes across species will significantly underestimate the number of conserved cycling genes.

Our results are strongly supported by additional analyses. We show that cycling genes conserved in multiple species have much stronger cell cycle characteristics than the full list for each single species. There are also extensive interactions within the set of conserved cycling genes, and almost half of the CCC4 yeast genes are essential. These observations and GO analysis indicates that these genes compose a crucial part of the cell cycle system. Together, these findings support our claim that we have derived a core conserved set of cycling genes.

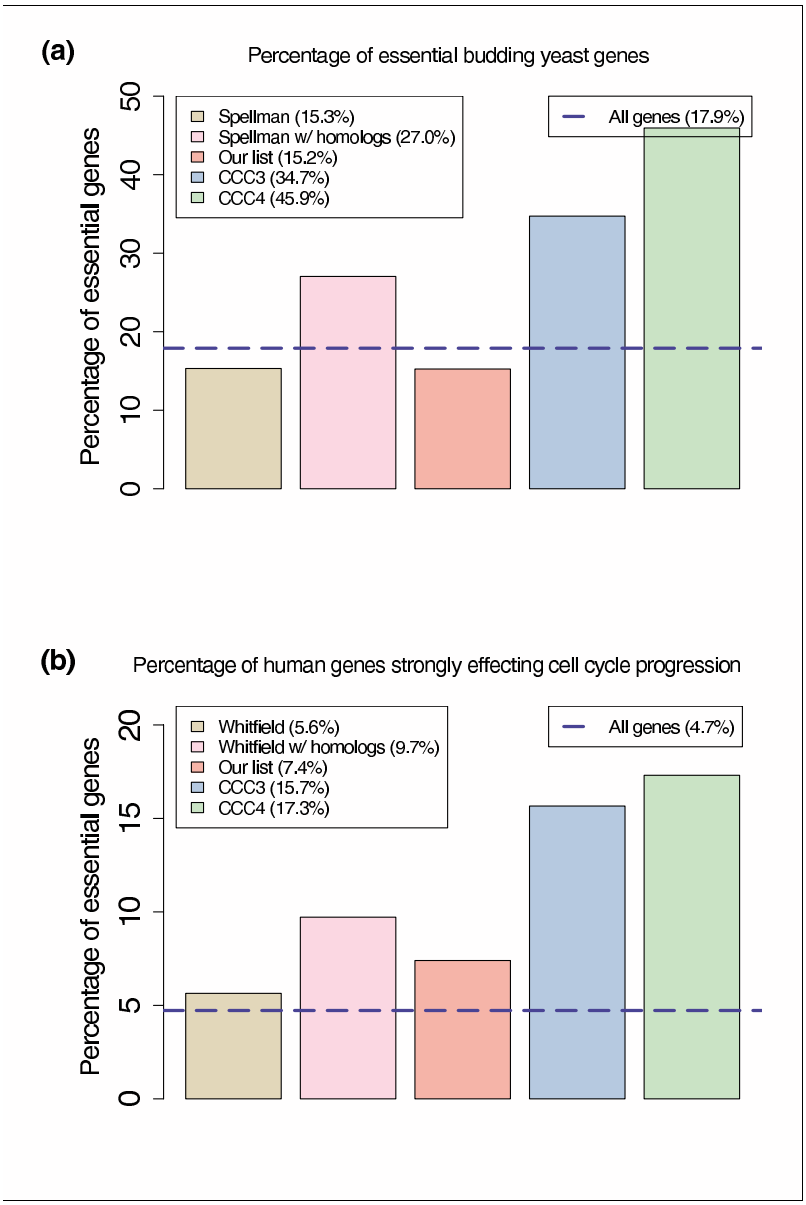


Figure 3.11: The importance of the core cycling genes. (a) Percentage of essential genes in different sets of budding yeast genes [Winzeler et al., 1999]. Although 18% of budding yeast genes are essential, only 15% of cycling genes are essential. Our analysis resolves this apparent contradiction by showing that the conserved cycling genes lists contain a much higher percentage of essential genes (35% and 46% for CCC3 and CCC4). Sequence alone cannot account for this high percentage (27%), indicating the importance of the combined analysis. (b) Similar analysis for the human lists using data from RNA interference knockdown experiments [Mukherji et al., 2006].

Chapter 4

Comparative Study of Gene Expression Regulation in Immune Response

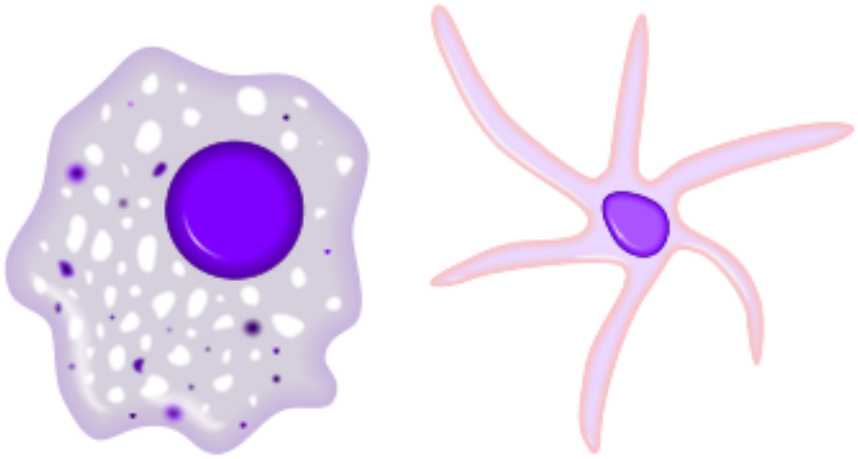
4.1 Overview

4.1.1 The Immune System

Functions and components of the immune system. Most multicellular organisms rely on their immune system to defend against the infection from a multitude of pathogens. In addition, the immune system is also responsible for removing dead cells, tumor cells, or cells infected by pathogens. There are two components of the immune system, namely the innate immune system and the adaptive immune system. The innate immune system is believed to be evolutionarily older and it exists in organisms from plants to humans. In contrast, the adaptive immune system only exists in vertebrates.

The immune system comprises of many types of cells, including macrophages, dendritic cells, neutrophils, natural killer cells, B-cells, T-cells, and they play different roles in the immune response. To understand how these cells collaborate to fend off pathogens of great diversity, we first need to know how they react differently to infections.

Gene expression in immune response depends on the types of the host cell and bacteria. After encountering pathogens, some of the host genes may be differentially expressed. Such changes in expression may have different patterns over a time course. Some genes may be induced, and some others may be repressed in response to the infection. The response pattern of a gene depends on the host



(a) (b)

Figure 4.1: (a) A macrophage and (b) a dendritic cell. (source: Wikipedia)

cell type in which the gene is expressed. This cell-type specific gene expression pattern enables different immune cells to carry out different functions in the immune response . The host immune response also varies greatly depending on the type of pathogens that trigger the response. The host cells may activate very different pathways when challenged by Gram-negative and Gram-positive bacteria. As another example, it has been known that some bacteria will induce very different host response if one of the bacterial genes is switched off [McCaffrey et al., 2004]. Other complicating factors include the susceptibility of the host and whether host cells have been exposed to interferon.

4.1.2 Application of Microarrays in Immunology

There have been many studies using microarrays to compare immune gene expression programs under different conditions. For example, Huang et al. [2001] compared the gene expressions in human dendritic cells infected by various pathogens and derived both a common set and pathogen-specific sets of differentially expressed genes. Boldrick et al. [2002] carried out similar studies on human macrophages, and they also studied the effect of different doses. Chaussabel et al. [2003] identified a set of commonly expressed genes in both human macrophages and dendritic cells, as well as genes uniquely expressed in one the of two cell types. In addition, there are studies comparing the effect of hosts factors and bacterial virulence on the gene expression profiles [Hoffmann et al., 2004, van Erp et al., 2006].

In all these studies, genes are ranked according to a score, e.g. fold-change, and a set of differentially expressed genes is selected using an arbitrary threshold, e.g. ranging from 2-fold change to 3-fold change. Genes in this set may be further clustered into groups, and a few genes are selected for further study. While this approach has already generated many interesting results, it may have missed genes that play an important role in immune response if they just fall below the threshold. This problem is especially important because microarray data are usually very noisy.

4.1.3 Comparative Study of the Immune System

Conservation of the immune system. There is considerable conservation of the immune system at the genomic level between different species, especially genes related to the innate immune response. For example, toll-like receptors, a major class of pattern recognition proteins, are found to be highly conserved across all species [Aderem and Ulevitch, 2000]. It is interesting to find out how much of the immune system is conserved during evolution by comparative study across species. At the same time, this conservation provides us with correlation information between species, which can be used to better interpret noisy experimental results.

Identifying Immune Response Genes by Combining Data from Multiple Species. Microarray expression experiments that study immune response to bacteria infection can be divided along several lines. Here we focus on three such divisions: Cell type, bacteria type and host species.

Innate immunity is the result of the collective responses of different immune cells, which are differentiated from multipotential hematopoietic stem cells [Keller and Snodgrass, 1990]. To understand the roles of and possible interplays between different types of immune cells, it is important to identify both the common responses of different immune cells, as well as responses unique to a certain cell type. Identification of genes differentially expressed in macrophages but not in dendritic cells, and vice versa, may highlight their specific functions and help us understand mechanisms leading to their different immune response roles. In addition to the different cells, specific bacteria types are known to trigger very different innate immune responses [Nau et al., 2002]. Specifically, response to Gram-positive and Gram-negative bacteria is activated by different membrane receptors that recognize molecules associated with these bacteria. Finally, many of the key components in the innate immune system are highly conserved [Hoffmann et al., 1999]. For example, the structure of Toll-like receptors (TLRs), a class of membrane receptors that recognizes molecules associated with bacteria, is highly conserved from Drosophila to mammals. It is less known though to what extent the immune re-

sponse program is conserved and what other genes play a role in this conserved response.

While each of these subsets of experiments (macrophages vs. dendritic, human vs. mouse etc.) can be analyzed separately using ranking methods and then compared to each other, due to noise in gene expression data methods that rely on a score cutoff become much less reliable for genes closer to the threshold [Lu et al., 2007]. Thus, analyzing responses to different pathogens and then comparing the lists derived for each experiment may not identify a comprehensive list of immune response genes. Similarly, while comparing the expression changes triggered by similar bacteria in human and mouse may lead to the identification of conserved immune response patterns, direct comparison of these profiles across experiments is sensitive to noise and orthology assignments, leading to unreliable results and underestimation of conservation [Lu et al., 2007].

It is therefore desirable to combine microarray gene expression datasets from different studies to overcome noise in the datasets and jointly infer genes that are involved in immune response. In Chapter 3 we have combined expression datasets from four species to identify conserved cell cycle genes. The underlying idea is that pairs of orthologous genes are more likely than random pairs of genes to be involved in the same cellular system. Thus, if one of the genes in the pair has a high microarray expression score while the other has a medium score, we can use the high scoring gene to elevate our belief in its ortholog, and vice versa. Our method in Chapter 3 used discrete Markov random fields to construct a homology graph between genes in different species. Next, we developed a belief propagation algorithm to propagate information across species allowing orthologous genes to be analyzed concurrently.

Here we extend this method in several ways so that it can be applied to analyzing immune response data. Unlike the cell cycle, which we assumed worked in a similar way in all cell types of a specific species, here we are interested in both common responses and distinguishing responses for each dividing factor. This requires a different analysis of the posterior values assigned to nodes in the graph. In addition, for the immune response analysis, genes are represented multiple times in the graph (once for each cell and bacteria type) leading to a new graph topology. We are also interested in multiple labels for immune response (up, down, not changing) compared to the binary labels we used for cell cycle analysis (cycling or not). Finally, we use a Gaussian random field instead of a discrete Markov random field. Instead of simply connecting genes with high sequence similarity, the edges in the graph are determined in a novel way that enables us to better utilize the information contained in sequence homology, leading to improved prediction performance.

In the following sections, we will introduce our model for integration of inho-

mogeneous immune response datasets.

4.2 The Model

We formulate the problem of identifying immune response genes using probabilistic similarity network models. In particular, we use Gaussian random fields (GRFs) to model the assignment of gene labels. Gaussian random fields are a special type of Markov random fields. In a GRF, every node follows a normal distribution, and all nodes jointly follow a multivariate normal distribution.

There are two types of nodes in our graphical model (Figure 4.2). The first type is a gene node; it represents the status of a gene in a certain cell type, from a certain host species, in response to a certain type of pathogen. Here we consider two cell types (macrophages and dendritic cells), two host species (humans and mice), and two pathogen types (Gram-negative and Gram-positive bacteria). The number of gene statuses can be either two (involved in immune response or not), or three (suppressed, induced, or unchanged during immune response). For simplicity, we will describe our model using two gene classes, but will present the results based on both two and three classes in the Results section. Corresponding to each gene node is also a score node, representing the observation of expression of the corresponding gene. Together, the GRF jointly models the statuses of all genes in all cell types, all species, and under both types of infection conditions.

The edges in the GRF represent the conditional dependencies between statuses of genes. We put an edge between two gene nodes when they are *a priori* more likely to have the same status than otherwise. Specifically, there are two cases where we add an edge. In the first case, for each gene node in the graph, we connect it with another gene node if the two genes share high sequence similarity, and the experiments related to both nodes are on the same cell type and bacteria type. The assumption is that genes with similar sequence are more likely to have similar functions in the same type of cells and under the infection of the same type of bacteria. The edge potential function, defined on the edges, introduces a penalty when two genes with high sequence similarity are assigned different statuses. In the second case, we connect a gene node with another gene node if the two nodes represent the same gene in the same type of cell (or infected by the same type of bacteria). Here we assume the genes are likely to function similarly in the same type of cells, or under the same type of infection. Again, the potential function penalizes the situation where a gene is assigned different status under different conditions. The amount of penalty depends on the strength or weight attached to the edge. Different edges may have different weights. The joint probability is defined as the product of the node potential functions and edge potential functions,

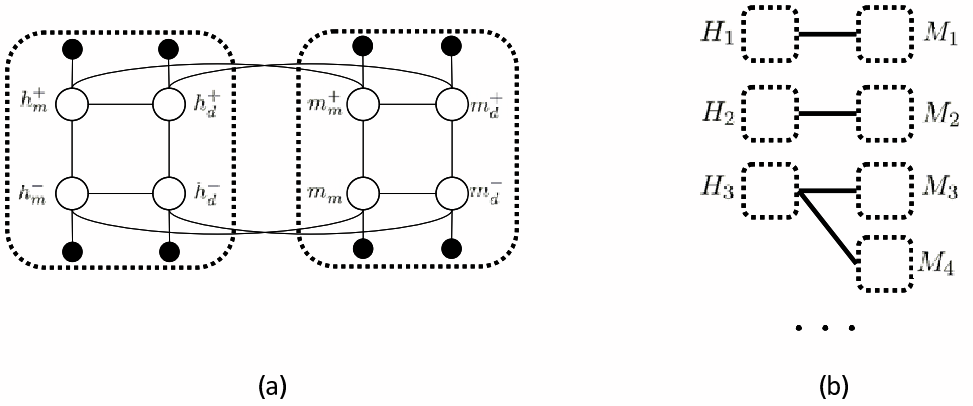


Figure 4.2: Diagram of the Gaussian random field (GRF) model. (a) A subgraph in the GRF containing homologous human and mouse genes. The white node h_m^+ represents the (latent) status of the human gene h in macrophages under infection of Gram-positive bacteria. h_m^- represents the genes status in macrophages under infection of Gram-negative bacteria. h_d^+ and h_d^- represent the statuses of the same genes in dendritic cells under the infection of Gram-positive or Gram-negative bacteria. m_m^+, m_m^-, m_d^+, and m_d^- are similarly defined for the homologous mouse gene m. Two white nodes are connected by an edge if they represent the same gene in two experiments, either on the same cell type or under the infection of the same type of bacteria. We also connect two white nodes if they represent homologous genes in the same cell type and under the infection of the same type of bacteria. The black nodes represent the observation from the expression data in a certain cell type and under the infection of the appropriate bacteria. They are connected with the white nodes representing the corresponding genes under the same condition. (b) A high level diagram of the GRF model. Each dotted box represents a subgraph of four nodes related to the same gene as those shown in (a), and each edge represents four edges connecting the nodes of homologous genes in the two dotted boxes, in the same way as shown in (a).

divided by a normalization function. We can infer the status of individual genes by estimating the joint maximum a posteriori (MAP) assignment of all nodes.

4.2.1 Computing Weight Matrix

When assigning the edge weights, we employ a similar approach to the one in Chapter 3, where we use a Markov random field to jointly model gene statuses in multiple species. In that model, the edges in the graph are weighted by BLAST [Altschul et al., 1990] scores between pairs of genes. Given two genes connected

in the graph, the edge weight (BLAST bit score) represents the sequence similarity between the two genes, which in turn captures the *a priori* dependency between their statuses. However, in a Markov random field model, an edge represents the dependency between the two nodes conditional on the statuses of all other nodes [Bishop, 2006]. In contrast, sequence similarity is computed for a pair of genes regardless of other genes. In other words, what a BLAST score captures is the marginal dependency between the two genes' statuses.

We address this discrepancy based on a connection between edge weights and the covariance matrix of Gaussian random fields. The edge weights of a GRF can be organized into a (symmetric) matrix, where each row (and each column) corresponds to a node, and each element in the matrix is the weight on the edge connecting the corresponding nodes. This weight matrix is the same as the inverse of the covariance matrix of the GRF [Zhu, 2005].

Using this observation, we can build a similarity matrix based on BLAST scores, and use its inverse as the weight matrix on the GRF. Each row (and each column) in the similarity matrix corresponds to a gene. If the BLAST bit score between two genes is above a cutoff, we set the corresponding elements in the similarity matrix to that score. Otherwise, it's set to zero. We use a stringent cutoff so that we are fairly confident of the functional conservation when we add a non-zero element.

Because the similarity matrix contains scores for all genes in two species, the computational cost to invert it is very high. Instead, we compute an approximate inverse. We first convert the whole matrix into a diagonal block matrix by Markov clustering algorithm [Enright et al., 2002], then compute the approximate inverse by inverting each block independently. The matrix inversion is done by Sparse Approximate Inverse Preconditioner [Deshpande et al.].

Finally, we assign edge weights based on this inverse matrix. Note that each gene is represented by four nodes in the graph, because it is present in different experiments on two cell types and two pathogen types. For edges connecting gene nodes in the same cell type and pathogen type, we set the weight according to the inverse similarity matrix. For edges connecting nodes that are identical except for cell type, we use a single edge weight, a hyper-parameter. For edges connecting nodes that are identical except for pathogen type, we use yet another hyper-parameter.

4.2.2 Expression Score Distribution

The gene expression score is a numeric summary computed from the gene's microarray time series. We assume that for each gene population (in the case of two gene classes: involved in immune response or not; in the case of three gene classes:

induced, suppressed, or unchanged during immune response) the scores follow a Gaussian distribution with its own mean and variance. Due to the simplicity of the model and noise in the microarray experiments, the Gaussian distributions are highly overlapped, which makes them hard to separate by expression score alone [Lu et al., 2006].

4.2.3 Node Potential Function

The node potential functions capture information from gene expression data. For each gene i, let C_i denote its (hidden) status, S_i denote its expression score, y_i denote the random variable in the GRF associated with the gene. C_i can be a binary variable if we consider two gene classes (involved in immune response or not), or a ternary variable if we consider three gene classes (induced, suppressed, or unchanged). S_i and y_i are both real variables. Because each y_i follows a (different) normal distribution, we need to have a way to link a gene's probability of belonging to each class with the corresponding normal distribution. This is achieved by the probit link function. Take two gene classes for example. Let p_i be the probability of gene i being involved in immune response conditional on its expression score S_i,

$$p_i = \Pr(C_i = 1|S_i) = \frac{\Pr(S_i|C_i = 1)}{\Pr(S_i|C_i = 1)\Pr(C_i = 1) + \Pr(S_i|C_i = 0)\Pr(C_i = 0)}$$

The node potential function is defined as

$$\psi(y_i) = \phi(y_i|\mu = \Phi^{-1}(p_i), \sigma^2 = 1) \tag{4.1}$$

where $\phi(y_i|\mu, \sigma^2)$ is the probability density function for the normal distribution with mean μ and variance σ^2, and $\Phi^{-1}(x)$ is the probit function, i.e. the inverse cumulative distribution function for the standard normal distribution. In other words, the information from a gene's expression score is encoded by a normal distribution of y_i such that $p_i = \Pr(y_i > 0)$.

In the case of three gene classes ($C_i \in \{-1, 0, +1\}$), we can use the following formulas to link the probabilities of C_i and y_i:

$$\Pr(C_i = 1|S_i) = Pr(y_i > 1) \tag{4.2}$$
$$\Pr(C_i = -1|S_i) = Pr(y_i \leq -1) \tag{4.3}$$
$$\Pr(C_i = 0|S_i) = \Pr(-1 < y_i \leq 1) \tag{4.4}$$

It can be proved that given any (non-zero) probability mass function on C_i, we can find a normal distribution $N(\mu, \sigma^2)$ such that these formulas are satisfied when $y_i \sim N(\mu, \sigma^2)$.

4.2.4 Edge Potential Function

The edge potential functions capture the conditional dependencies between pairs of gene nodes. The assumptions here are that (1) genes with higher sequence similarity are more likely than otherwise to have the same or similar functions; and (2) a given gene is more likely than otherwise to have the same function across cell types and across pathogens.

First we will define the edge potential functions for edges connecting genes in the same cell type and under infection of the same type of bacteria. In this case, the edge potential function depends on the weight matrix we introduced in Section 4.2.1. Note that although all elements in the BLAST score matrix are non-negative (sequence similarities are non-negative), its inverse matrix may have negative elements. As a consequence, edge weights can be either positive or negative. A positive edge weight means the statuses of the two gene nodes are positively correlated, conditional on the status of all other gene nodes. A negative edge weight means they are negatively correlated, conditional on all other gene nodes.

The following edge potential function captures this dependency (λ_0 is a positive hyperparameter):

$$\psi_0(y_i, y_j) = \begin{cases} \exp\{-\lambda_0 |w_{ij}| (y_i - y_j)^2\} & \text{if } w_{ij} \geq 0 \\ \exp\{-\lambda_0 |w_{ij}| (y_i + y_j)^2\} & \text{if } w_{ij} < 0 \end{cases}$$

When the edge weight w_{ij} is positive, the edge potential function places a penalty if y_i and y_j are different. The larger the difference, the higher the penalty. Likewise, when w_{ij} is negative, the edge potential function introduces a penalty based on how close y_i and y_j are to each other. The penalty becomes higher when the y_i and y_j are closer.

For edges connecting the same gene in the same cell type but under infection of different type of bacteria, the edge potential function is defined as

$$\psi_1(y_i, y_j) = \exp\{-\lambda_1 (y_i - y_j)^2\}$$

where λ_1 is a positive hyperparameter. Similarly for edges connecting the same gene under the infection of the same type of bacteria but in different cell types, the edge potential is defined as

$$\psi_2(y_i, y_j) = \exp\{-\lambda_2(y_i - y_j)^2\}$$

where λ_2 is a positive hyperparameter. Together, the joint likelihood function is defined as

$$L = \frac{1}{Z}\prod \psi(y_i) \prod \psi_0(y_i, y_j) \prod \psi_1(y_i, y_j) \prod \psi_2(y_i, y_j) \tag{4.5}$$

4.3 Learning the Model Parameters

In this section we will present our algorithm based on two gene classes. The algorithm can be extended to three gene classes by using different node potential functions (See discussion in Section 4.2.3). We need to learn the parameters of the expression score distributions for each combination of cell types, host species, and pathogen types. In each case, there are four parameters $(\mu_0, \sigma_0^2, \mu_1, \sigma_1^2)$, i.e. the means and variances of the two different Gaussian distributions, one corresponding to the scores of immune response genes, the other corresponding to the scores of the remaining genes.

We learn these parameters in an iterative manner, by an EM-style algorithm. We start from an initial guess of the parameters. Based on these parameters, we infer "soft" posterior assignments of labels to the genes using a version of the belief propagation algorithm on the GRF. The posterior assignments are in turn used to update the score distribution parameters. We repeat the belief propagation algorithm based on the new parameters to infer updated assignments of labels. This procedure goes on iteratively until the parameters and the assignments do not change anymore.

4.3.1 Iterative Step 1: Inference by Belief Propagation

Given the model parameters, we want to compute the posterior marginal distribution for each latent variable y_i, from which we can derive for each gene node the posterior probability of being involved in immune response. It is hard to compute the posteriors directly because the computational complexity of the normalization function in the joint likelihood function scales exponentially. However, due to the dependency structure in the GRF, we can adapt the standard Belief Propagation algorithm [Yedidia et al., 2003] for GRF, and use it to compute all the posteriors efficiently. Unlike MRFs defined on discrete variables, variables in GRFs are continuous and follow normal distributions. The current estimation of the marginal

posterior ("belief") of every latent variable y_i in the GRF is a normal distribution. Similarly, the "messages" passed between nodes are also normal distributions.

The Belief Propagation algorithm consists of the following two steps: "message passing", where every node in the GRF passes its current belief to all its neighbors, and "belief update", where every node updates its belief based on all incoming messages. The algorithm starts from a random guess of the beliefs and messages, and then repeats these two steps until the beliefs converge.

1. Message passing. In this step, every node y_i computes a message for each of its neighbors y_j, sending y_i's belief of y_j's distribution. The message is based on the potential functions, which represent local information (node potential) and pairwise constraints (edge potential), as well as incoming messages from all y_i's neighbors except y_j.

$$m_{ij}(y_j) \leftarrow \int_{y_i} \psi(y_i, y_j)\psi(y_i) \prod_{k \in N(i) \setminus j} m_{ki}(y_i)$$

2. Belief update. Once node y_i has received messages from all its neighbors, it updates the current belief incorporating all these messages and the local information from the node potential. The update rule is as follows

$$b_i(y_i) \leftarrow \frac{1}{v_i}\psi(y_i) \prod_{k \in N(i)} m_{ki}(y_i)$$

where v_i is a normalization constant to make $b_i(y_i)$ a proper distribution.

Because all the messages and beliefs are normal distributions, they can be represented by the corresponding means and variances. More importantly, in this case the message update rule and belief update rule can be formulated into rules updating the means and variances directly, thus avoiding computationally expensive integration operations. The exact update rules are given in the appendix.

4.3.2 Iterative Step 2: Updating the Score Distribution

The posterior computed in step 1 is based on the current (the g'th iteration) estimation of parameters, collectively denoted by $\Theta^{(g)}$. The goal now is to determine the parameters that maximize the expected log-likelihood of the complete data over the observed expression scores given the parameters $\Theta^{(g)} = (\mu_0^{(g)}, \sigma_0^{(g)}, \mu_1^{(g)}, \sigma_1^{(g)})$.

To update the parameters of the score distributions, we first compute the posterior probability of a gene being involved in immune response, based on the posterior of y_i. This is the same as applying the reverse probit function:

$$\Pr(C_i = 1|\Theta^{(g)}) = \int_0^{+\infty} b_i(y_i)dy_i$$

For simplicity, we use the following notations

$$p_i^{(g)} = \Pr(C_i = 1|\Theta^{(g)}) \qquad q_i^{(g)} = \Pr(C_i = 0|\Theta^{(g)})$$

The updated distribution parameters for a Gaussian mixture are computed by standard rules

$$\begin{aligned}
\mu_0^{(g+1)} &= \frac{\sum_i q_i^{(g)} S_i}{\sum_i q_i^{(g)}} \\
\mu_1^{(g+1)} &= \frac{\sum_i p_i^{(g)} S_i}{\sum_i p_i^{(g)}} \\
\sigma_0^{(g+1)} &= \sqrt{\frac{\sum_i q_i^{(g)} (S_i - \mu_0^{(g+1)})^2}{\sum_i q_i^{(g)}}} \\
\sigma_1^{(g+1)} &= \sqrt{\frac{\sum_i p_i^{(g)} (S_i - \mu_1^{(g+1)})^2}{\sum_i p_i^{(g)}}}
\end{aligned}$$

4.4 Results

4.4.1 Immune Response Data

Immune response data. Immune response microarray experiments were retrieved from supporting websites of [Detweiler et al., 2001, Chaussabel et al., 2003, Huang et al., 2001, Lang et al., 2002, Hoffmann et al., 2004, van Erp et al., 2006, McCaffrey et al., 2004, Draper et al., 2006, Granucci et al., 2001], totaling 21 data sets. The data sets include experiments on macrophages and dendritic cells in humans and mice. For each cell type we have included experiments using Gram-positive and Gram-negative bacteria, except for mouse dendritic cells, for which we only found Gram-negative bacteria datasets. Human and mouse orthologs were downloaded from Mouse Genome Database [Eppig et al., 2005]. Tables 4.1 and 4.2 summarize the datasets used in this paper.

Host/Cell Type	Gram- Datasets	Gram+ Datasets
Human Macrophages	4	2
Human Dendritic Cells	3	2
Mouse Macrophages	3	6
Mouse Dendritic Cells	1	0

Table 4.1: Summary of immune response datasets used.

Host/Cell Type	Gram- Datasets	Gram+ Datasets
Human Macrophages	*Samonella enterica* subspecies *typhimurium*	*Mycobacterium tuberculosis*
Human Dendritic Cells	*Escherichia coli*	*Mycobacterium tuberculosis*
Mouse Macrophages	Lipopolysaccharide	*Listeria monocytogenes*, Group B *streptococcus*
Mouse Dendritic Cells	*Escherichia coli*	

Table 4.2: Summary of infectious agents used.

4.4.2 Computing Expression Scores

Computing expression scores. For each gene in each experiment, an expression score is computed from the gene expression time series data. The score is based on the slope of the time series to capture both the change in expression levels and the time between infection and response. Specifically, we first determine the sign of a gene's score (s_i) by comparing the absolute values of the highest and the lowest expression levels. The score is positive if the former is higher, or negative if the latter is higher. Denote the time point that corresponds to the highest expression level (in the former case) or to the lowest expression level (in the latter case) as t_i. The score is computed as follows: $S_i = s_i * \text{expression}(t_i)/t_i$.

Due to different protocols being used and experimental noise, agreement between different datasets, even if done using the same type of cells and bacteria, may sometimes be low . For example, the overlap between lists of fission yeast cell cycle genes identified in three studies [Rustici et al., 2004, Oliva et al., 2005, Peng et al., 2005] is on 30%. Nevertheless, since each dataset contains new observation of the same underlying biological process, combining them may better capture the biological truth. Here we want to combine scores from different experiments on the same host cell type and bacteria type. For example, there are five datasets where human macrophages were infected by Gram negative bacteria, and we would like to combine the five scores for each gene into one.

To test for the consistency between the datasets to be combined, we define the

Host/Cell Type	Gram	Consistency	p-value
Human Macrophages	$+$	0.577	< 0.001
Human Macrophages	$-$	0.490	< 0.001
Human Dendritic Cells	$+$	0.610	< 0.001
Human Dendritic Cells	$-$	0.546	< 0.001
Mouse Macrophages	$+$	0.466	< 0.001
Mouse Macrophages	$-$	0.553	< 0.001

Table 4.3: Consistency between immune response datasets.

following measure

$$\text{Consistency} = \frac{\text{\# of genes ranked in top 1000 in at least } \max\{|D|/2, 2\}\text{datasets}}{1000}$$

for each cell/bacteria type, where $|D|$ is the number of datasets. We compare it to the consistency of randomized data, and compute an empirical p-value (Table 4.3). In each case, the consistency is significant with a p-value < 0.001.

4.4.3 Recovering Known Human Immune Response Genes

Recovering known human immune response genes. To evaluate the performance of our model, we retrieved 642 human innate immune response genes from a database [Kelley et al., 2005], and used them as the labeled data. We learned the model parameters by three-fold cross validation using the labeled data. We compared the performance of GRF, MRF, and the baseline model where genes are ranked by their expression score alone. We use the fraction of known immune response genes recovered by a model as the performance measure. Because the set of immune response genes we used does not have labels indicating the cell types or infection conditions, we treat a gene as "positive" regardless of the cell type and bacteria type. For GRF and MRF models, the genes were ranked by their highest posterior probability (in any of the cell or bacteria types). For the baseline model, the genes are ranked by their expression scores. As we show in Figure 4.3, both GRF and MRF models outperform the baseline model. These models are able to infer a better gene's posterior probability by transferring information between the same gene across cell types or from homologous genes across species. At the threshold of top 10% genes, MRF is able to recover 28% of known immune response genes, compared with 26% by the baseline model. Encouragingly, GRF leads to the biggest improvement in performance. Of the top 10% high scoring genes based on the posterior computed by GRF, 35% are known immune response genes, a 34.6% increase compared to the baseline (score only) model.

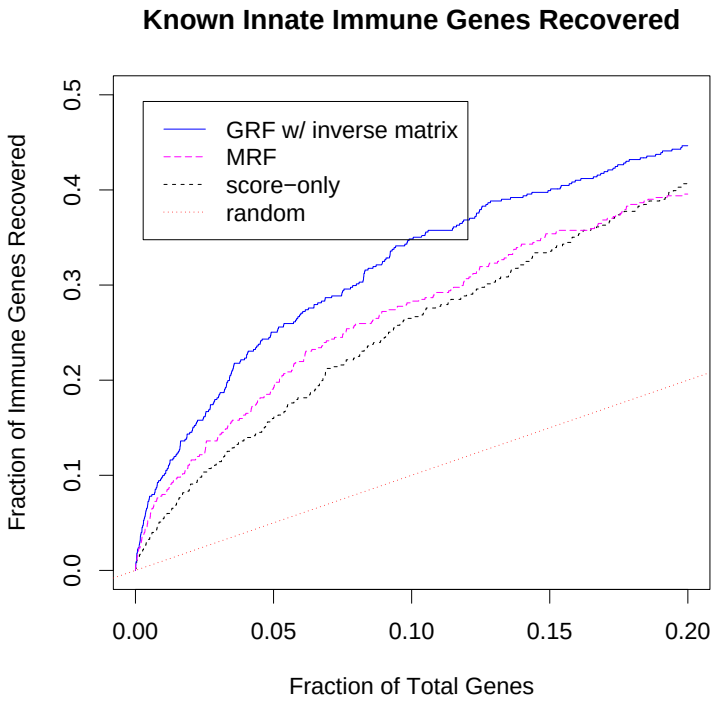


Figure 4.3: Performance comparison of the Gaussian random field (GRF) with improved weights, the Markov random field (MRF), and the baseline model where genes are ranked by their expression scores. Using MRF we were able to recover 18% known immune genes in the top 5% of ranked genes. This is a 28% improvement compared with the baseline model (which recovers 14% of the immune genes). The GRF model is able to recover 25% known immune genes at the same threshold, a 79% improvement over the baselyne method and a 38% improvement over the MRF model.

4.4.4 Identification of Common Response Genes

Identification of common response genes by combined analysis. Based on the learned posterior probabilities, we ranked the genes for each cell type in each species, for both Gram-positive and Gram-negative infections. We identified 57 ortholog pairs that are assigned high posterior in all cell types and infection types. These genes are commonly induced by all bacteria in both macrophages and dendritic cells across the two species. We first compared our list with a separate list of

genes commonly induced in human macrophages by various bacteria. This latter list was derived from expression experiments that were not included in our analysis [Nau et al., 2002]. The results confirmed the lists we identified. The overlap between the two lists was highly significant with a p-value = 1.70×10^{-25} (p-value computed using hypergeometric distribution).

We also compared our list with top 500 genes induced by *Mycobacterium tuberculosis* in mouse bronchoalveolar lavage (BAL) cells. Usually BAL cells include a large portion of macrophages and some dendritic cells. Again, we saw is a significant overlap between our list and the top induced genes in BAL cells (p-value = 1.50×10^{-7}).

To reveal the functions of the common response genes we carried out GO enrichment analysis using STEM [Ernst and Bar-Joseph, 2006]. The enriched GO categories include many common categories involved in immune responses, including "immune response" (p-value=3.9×10^{-8}), "inflammatory response" (p-value= 2.5×10^{-7}), "cell-cell signaling" (p-value=1.1×10^{-6}), "defense response" (p-value=1.5×10^{-6}), and "response to stress" (p-value= 2.4×10^{-5}).

Many of the classic players of innate immune activation and inflammation are recovered. For example, TNF is a proinflammatory cytokine and stimulates the acute phase reaction [Lukacs et al., 1995]. IL1 is an important mediator of inflammatory response and involved in cell proliferation, differentiation, and apoptosis [Mizutani et al., 1991, Bratt and Palmblad, 1997]. The list also includes chemokines that recruit and activate leukocytes (CCL3, CCL4, CCL5, CXCL1) [Wolpe et al., 1988] or attracts T-cells (CXCL9) [Valbuena et al., 2003]. Also important to the regulation of inflammation response is IL10, a well-known anti-inflammatory molecule [Lammers et al., 2003]. In addition, ETS2, NFkB, and JUNB are all very important transcription factors for inflammation. [Sun and Andersson, 2002].

To identify the pathways involved in common immune response, we searched for networks enriched by common response genes using Ingenuity Pathway Analysis (Ingenuity®Systems, www.ingenuity.com). One of such networks is shown in Figure 4.4.

4.4.5 Immune Response Conserved in Specific Cell Types

Immune response conserved in specific cell types. In addition to genes commonly induced across all dividing factors, we also identified genes that are differentially expressed between the two cell types. We identified 127 genes that are highly induced in dendritic cells in both bacteria types across human and mouse, but are not induced in macrophages. Many of the genes are known to be associated with functions of dendritic cells, especially the antigen processing and presentation. For

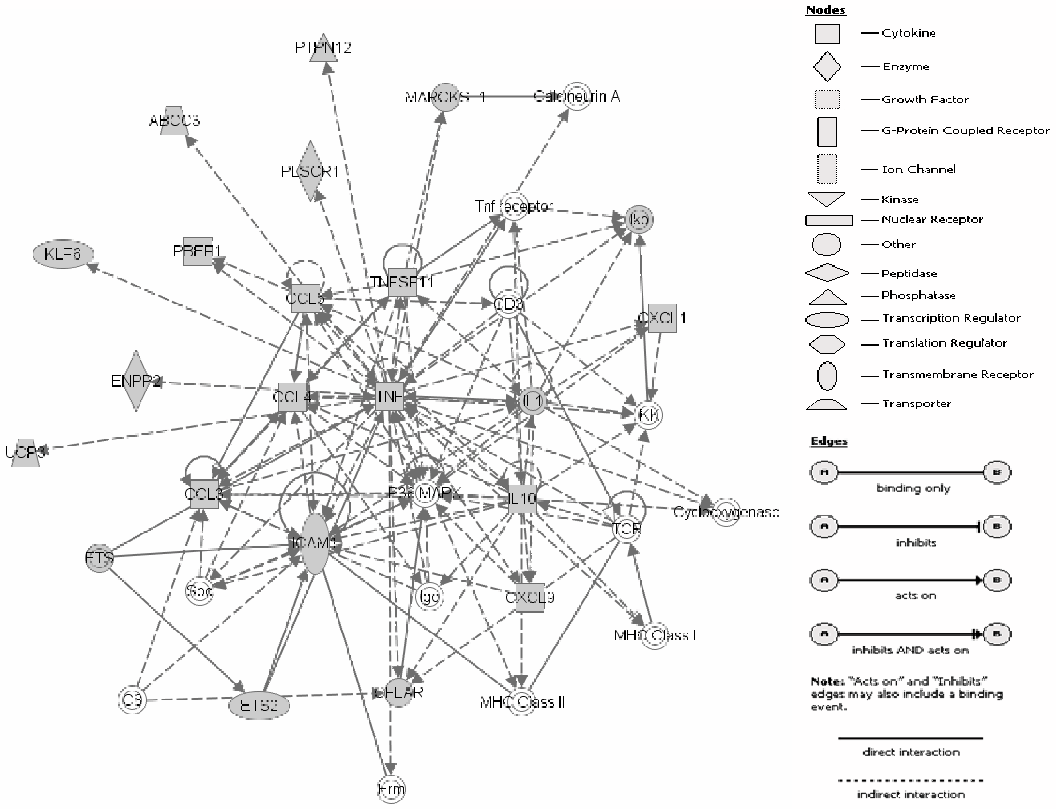


Figure 4.4: One of the networks of genes commonly induced in both dendritic cells and macrophages when infected by bacteria, in both human and mouse. The network was constructed by Ingenuity Pathway Analysis (Ingenuity®Systems, www.ingenuity.com). The gray-colored nodes are genes inferred to be expressed at high levels in all cell types, regardless of the bacteria type or species. White-colored nodes are genes interacting with commonly induced genes. Note the large fraction of the pathway recovered by our method. Many known immune response genes are present in this network. IL1 is an important mediator of inflammatory response and involved in cell proliferation, differentiation, and apoptosis [Mizutani et al., 1991, Bratt and Palmblad, 1997]. ETS2 is an important transcription factor for inflammation. CCL3, CCL4, and CCL5 are chemokines that recruit and activate leukocytes [Wolpe et al., 1988]. The profiles for one of these genes, CCL5, are shown in Figure 4.6

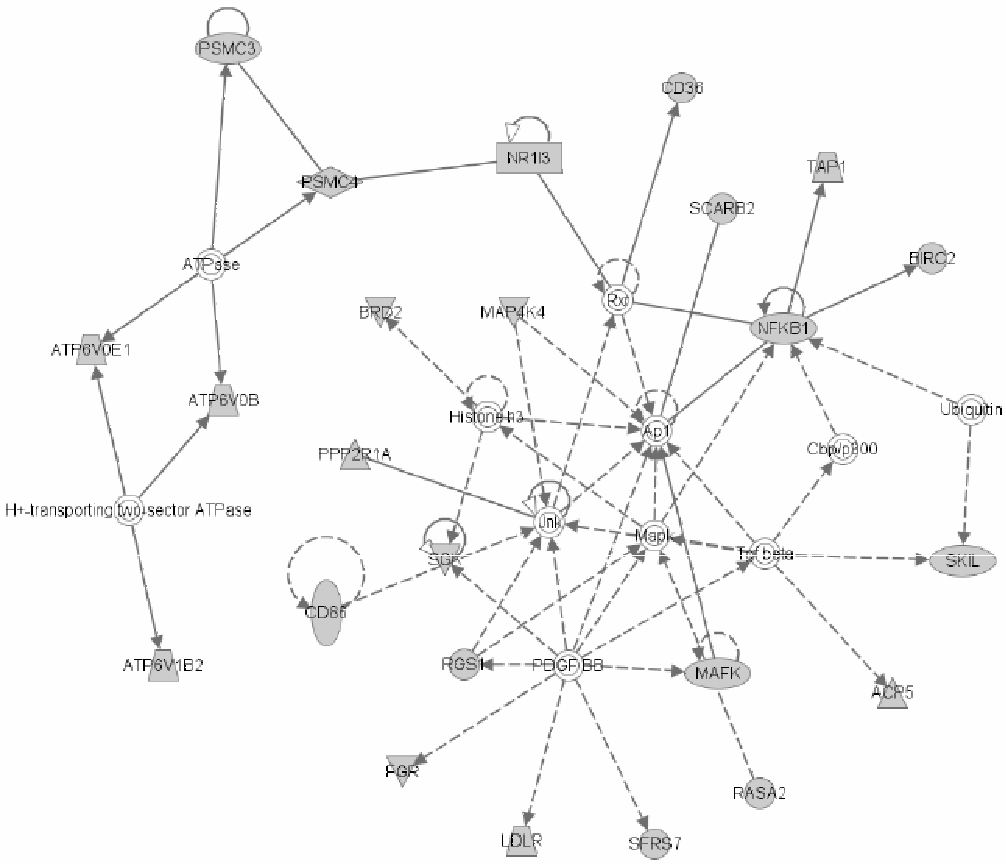


Figure 4.5: One of the networks of genes strongly induced in dendritic cells but less so, unchanged, or suppressed in macrophages. (The legend is the same as in Figure4.4). The network was constructed by Ingenuity Pathway Analysis (Ingenuity®Systems, www.ingenuity.com). The gray-colored nodes are genes inferred to be expressed at high levels in both dendritic cells and macro-phages, regardless of the bacteria type or species. White-colored nodes are genes interacting with commonly induced genes. Many known immune response genes are present in this network. CD86 is an essential co-stimulatory molecule that delivers this second signal and is also a marker of dendritic cell maturation. TAP is involved in the transportation of peptides generated by the proteosome from the cytosol to endoplasmic reticulum, which is an important step in MHC class I antigen presentation, a major function of dendritic cells. The profiles of CD86 are shown in Figure 4.7

example, components of the proteosome are prominently represented in the genes determined to be induced in dendritic cells. The proteosome is a multi-protein complex responsible for cleaving cytosolic proteins and is a necessary first step in MHC class I antigen presentation, a major function of dendritic cells. Peptides generated by the proteosome are then transported from the cytosol to endoplasmic reticulum by TAP, also represented in the gene list, where they are loaded on to MHC I molecules. Once the peptide-MHC I complex is displayed on the DC surface, the canonical class I pathway of antigen presentation is complete. Antigen presentation by DC is also accomplished through the class II pathway and the DC-specific gene list includes HLA-DRA, a human MHC II (class II) surface molecule. In addition to peptide-MHC complexes, T cell activation during antigen presentation requires a second signal. CD86 is an essential co-stimulatory molecule that delivers this second signal and is also a marker of dendritic cell maturation; CD86 is represented in the gene list. Also in the gene list enriched for expression in dendritic cells are TNFSF9 and TNFSF4. These molecules are cytokines that play a role in antigen presentation between dendritic cells and T lymphocytes. CD93 is represented in the DC results. This molecule is involved in the phagocytosis of apoptotic bodies. It is believed that phagocytosis of apoptotic bodies by dendritic cells has important effects on tolerance.

We searched pathways enriched by these genes, and one of the enriched networks in shown below in Figure 4.5.

We have also identified 157 genes that are more likely to be induced in macrophages than in dendritic cells. Among these genes, IFNGR1 is important for macrophages to detect interferon-gamma (also known as type II interferon), a key activating cytokine of macrophages. HMGB1 is believed to be involved in inflammation and sepsis. It is a chromatin structural protein that is released from some cells as a cytokine and is associated with fatal outcome from inflammation in sepsis. Another interesting gene is ADAM12, which is from a family of proteinases that are likely involved in tissue remodeling/wound healing by macrophages.

4.5 Summary

By combining expression experiments across species, cell types and bacteria type we were able to obtain a core set of innate immune response genes. The set we identified contained many of the known key players in this response and also included novel predictions. We have also identified the unique signature of macrophages and dendritic cells leading to insights regarding the set of processes activated in each of these cells type as part of the response.

While our method assumes that homologous genes share similar functions, it

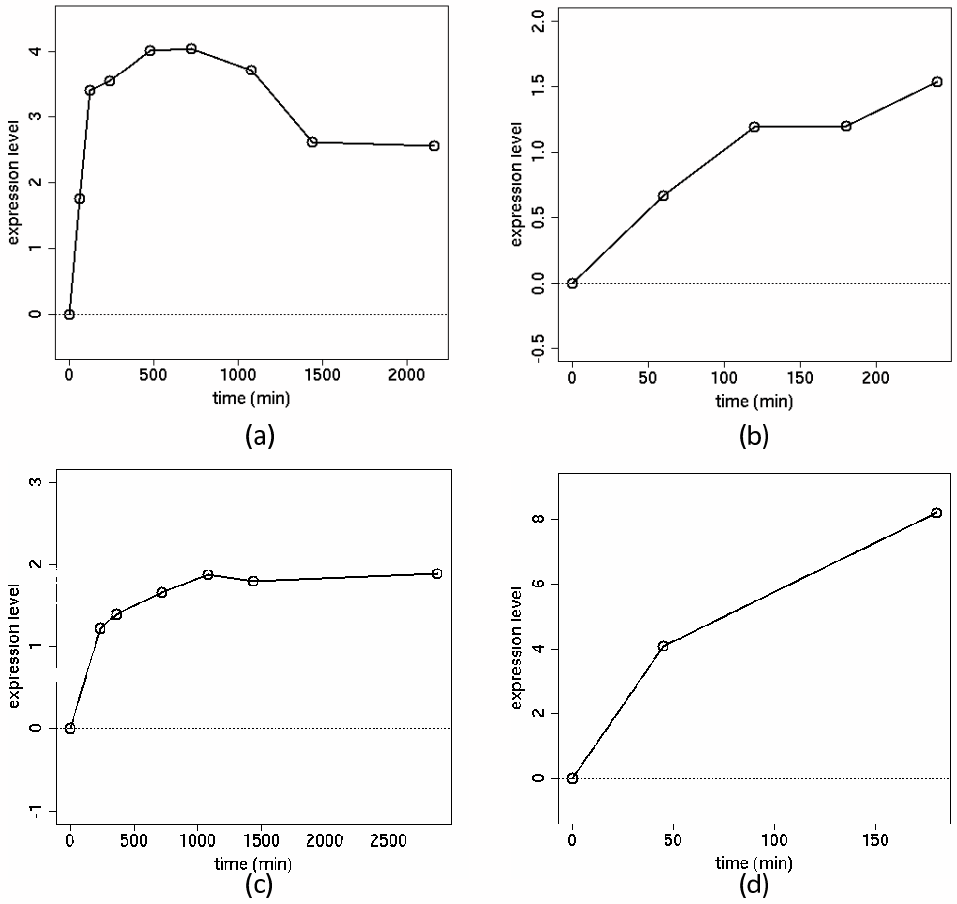


Figure 4.6: Expression profiles of CCL5 which was identified by our method as a common immune response gene. (a) and (b) are expression profiles for human CCL5 in dendritic cells and macrophages during immune response. (c) and (d) expression profiles for mouse CCL5 in dendritic cells and macrophages. The expression of both genes are strongly induced following infection.

is still sensitive to the observed expression profiles. Thus, if two homologs display different expression patterns they would be assigned to different cell or bacteria types. Still, the reliance on homology is a very useful feature for most genes. As we have shown, using this assumption we can drastically improve the ability of our method to recover the correct set of genes.

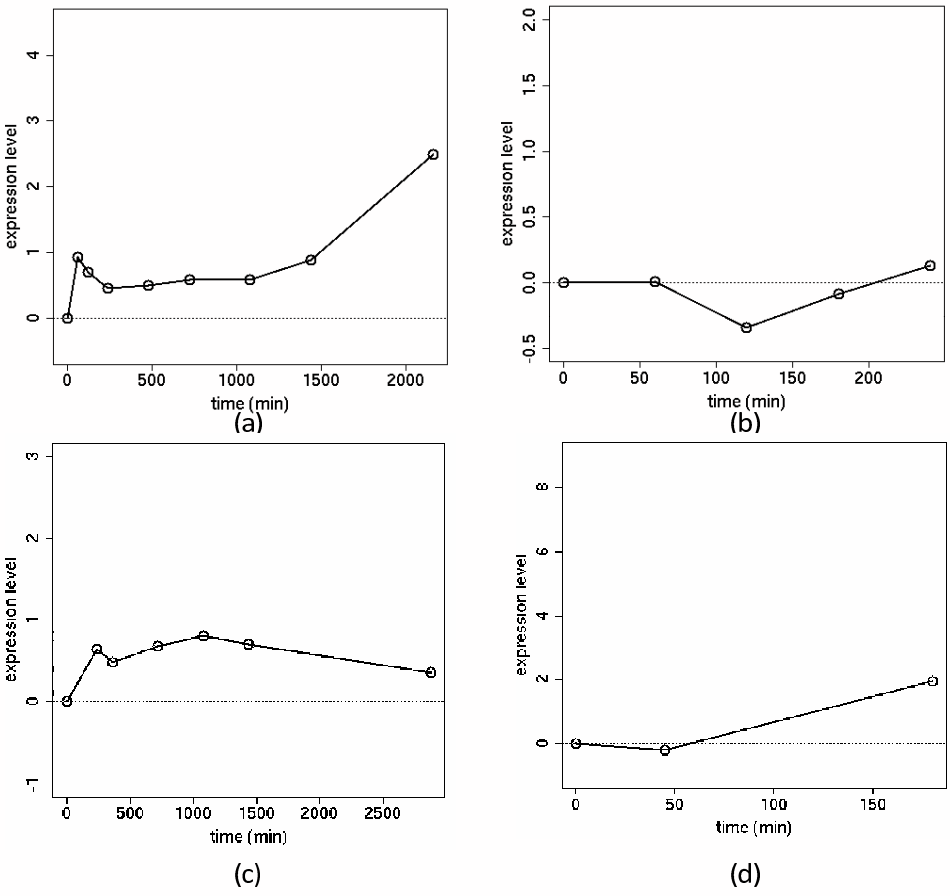


Figure 4.7: Expression profiles of CD86, a gene identified to be activated only in dendritic cells. (a) and (b) are expression profiles for human CD86 in dendritic cells and macrophages during immune response. (c) and (d) are expression profiles for mouse CD86 in dendritic cells and macrophages. For both species, the expression of the gene is induced after infection in dendritic cells, but unchanged in macrophages.

Chapter 5

Conclusions and Future Work

In previous chapters, I have described the development of a generative model for functional analysis of gene sets and two probabilistic models for combined analysis of multi-species microarray data, and the application of the models to the cell cycle and innate immune response gene expression programs. The results are summarized in Section 5.1. In Section 5.2 I discuss some of the open problems and extensions for future work.

5.1 Conclusions

With the growing amount of high-throughput biological data, modern biology is becoming more of a data-driven science. It is important to develop and apply computational methods that can take full advantage of the available data. A crucial issue is to integrate data of different types and from different sources for better analysis of biological systems.

5.1.1 Generative Model for Functional Analysis of Gene Sets

In this thesis, I have first presented an algorithm to incorporate the information in Gene Ontology [Ashburner et al., 2000], including both annotations and the hierarchical structure of the ontology, for functional analysis of gene sets. While many tools have already been developed for the same task, our method, GenGO, has the distinctive feature that it takes into account the full dependency structure encoded in GO, and has shown dramatic performance improvements in some cases.

The method is based on a generative probabilistic model, where I assume that the biological processes in a cell can have one of the two states, 'active' or 'inactive', and genes are activated by their associated biological processes. The algo-

rithm then looks for a small set of active biological processes that can best explain the set of observed genes. I compare GenGO with three other existing methods on simulated gene sets, using both annotations on budding yeast and humans. The performance of GenGO is close to perfect when the noise level is low, and still much better than all other methods even with high level of noise. Finally, I apply GenGO to the analysis of real data from a number of different experiments and species, and shows that it is able to avoid much redundancy and accurately balance the set of GO categories it returns, including both high level and specific categories.

5.1.2 Random Field Models for Analysis of Cross-Species Data

For cross species analysis, I have presented algorithms combining sequence data and gene expression data from multiple species and cell types to study the underlying gene expression program. As I have shown in different parts of the thesis, this approach has led to better identification of the genes participating in the expression programs.

For the cell cycle, I propose a Markov random field model that jointly models the cell cycle status of genes in multiple species. According to this model, a gene is more likely to be a cycling gene if its cyclic (expression) score is higher. Also, when two genes are similar in the sequence and one of them has a borderline cyclic score, it is more likely to be a cycling gene if the other gene is a cycling gene. The algorithm looks for the assignment of the cell-cycle statuses that maximizes the joint likelihood of all cyclic scores and sequence similarity. I compare the performance of our method with the method that uses only expression data, and show that our method is able to recover more known cycling genes. For the innate immune response, I propose a Gaussian random field model, which models the response status of genes in two cell types from two species, infected by a number of different bacteria. I show that the Gaussian random field model performs better than the Markov random field model, as well as the method that uses only expression data, in recovering known immune genes.

I have also presented methods to delineate the core components that are conserved across species/cell types, as well as those belong to specific species/cell types. For the cell cycle program, I am able to find a core set of conserved cycling genes. I analyze the conserved cycling genes using a number of complementary high-throughput datasets, and show that these genes have much stronger characteristic of cell cycle regulation than the full list of cycling genes. I also compare the core set with the full set of cycling genes and show it has much higher percentage of essential genes. For the immune response expression program, I identify sets of genes with conserved response. For example, CCL5 is shown to be induced in both dendritic cells and macrophages in both mouse and human. I also identify

sets of genes that are specifically induced in one cell type, but not in the other, and the differential induction is conserved between human and mouse. One example is CD86, which is induced in dendritic cells in both human and mouse, but not in macrophages.

While this thesis presents a number of computational tools for data integration, there is still much work left in the area. In the following section I will talk about some of the future directions.

5.2 Future Work

5.2.1 Biological Validation of Conserved Immune Response Genes

In Chapter 4 I have identified a few sets of genes that are conserved in the immune response between two cell types and between human and mouse. To further validate our results, we are planning to do more host-pathogen experiments. One idea is to pick some bacterium previously not used in our study (Table 4.2), and study the host immune response induced by it. It would be a good indicator of success if the set of conserved response genes are induced by this new bacterium.

5.2.2 Extensions of GenGO

In Chapter 2, I propose a generative model, GenGO, for functional analysis of gene sets. In the model, I assume that genes are in one of two states, either active or inactive. In other words, the state of a gene is assumed to be discrete. However, in many cases the activity of a gene may be better modeled by a continuous variable. For example, when profiling gene expression by microarrays, in addition to observing which genes are up- or down-regulated, one also observes the magnitude of the expression change, which is a continuous number. It would be a waste of information if the measurement is discretized into just two classes.

Here I describe a possible way to extend the model to support continuous states. The idea is to model a gene's activity state by a continuous variable on $[0, 1]$, and assume an active biological process can "generate" the activity state of its associated genes, following a beta distribution. For genes not associated with any active biological processes, their state follows another beta distribution. The beta distribution is a continuous distribution on $[0, 1]$ with two parameters. Its probability density function can be either unimodal or bimodal, providing great modeling flexibility. It is possible to determine the set of active biological processes by maximizing the likelihood of observed activity of all genes (normalized to within $[0, 1]$) over all possible sets of GO categories.

Another possible extension is to regularize the objective function using structural information of the GO hierarchy. In our current model, the likelihood function is penalized by the number of active GO categories. An interpretation of this penalty term is it corresponds to the prior probability of a set of GO categories being active. However, this formulation ignores the relation between GO categories. There are several possible ways to incorporate the structural information. For example, we can penalize the objective function by both the number of active categories, as well as the inverse distance between these categories. Intuitively, the latter means we prefer GO categories to be more spread out in the GO hierarchy. Another possible source of prior information is the size of (i.e. the number of genes in) an GO category. For example, we may want to penalize GO categories whose size is either too big or too small, reflecting our belief that these categories are less likely to be active.

5.2.3 Extensions of Random Field Models

In Chapter 4, I propose to use Gaussian random field models with the inverse weight matrix, and show it achieves better performance than Markov random field models in predicting immune response genes. The Markov random field model is based on the original weight matrix. It would be interesting to compare our results to Markov random field models with the inverse weight matrix, and see how much of the improvement is due to the inverse weight matrix.

One direction for future work is to find better ways to learn the graph structure of Gaussian random field models from data. Currently, I learn the structure by using the SPAI algorithm [Grote and Huckle, 1997] to compute a sparse approximate inverse of the weight matrix, which essentially tries to look for X that minimize the Robina's norm

$$\|WX - I\|_F$$

where W is the weight matrix and I is the identity matrix. The algorithm starts from some given sparse matrix, e.g. a diagonal matrix, then searches for a matrix that augments the sparsity structure as well as decreases the objective function. Many other ways have been proposed to learn a sparse graph. For example, one can choose to maximize the L_1-penalized log-likelihood (e.g. [Banerjee and El Ghaoui, 2008, Friedman et al., 2008]). It would be interesting to compare the different methods for learning the graph structure, and their impact on the prediction accuracy.

Another direction is to extend the random field models to handle multiple classes. In the study of cell cycle and immune response expression programs, the model assigns genes into two (cycling or not) or three classes (up-regulated,

down-regulated, and unchanged). While this approach has done a reasonably good job, in some cases we may need to handle more classes. For example, some immune response genes may be induced immediately after the host cell's exposure to the pathogen, while others may be induced in later stages. In order to characterize various dynamical responses of a gene expression program, we need to classify the genes based on when their expression profiles change. One possible way to handle multiple classes of expression patterns is outlined as follows. First we define k modal expression pattern profiles, e.g. "early up", "late up", "up down", "down up", etc. For each gene expression time series, we compute its distance to these profiles. Now we define a random field where each node is a (latent) $k-$dim Gaussian with mean equal to a gene's distance vector, and the edges are derived from homology. By approximating this random field by a Gaussian random field, we may be able to perform efficiently inference and determine the class membership for each gene.

5.2.4 Cross-Species Study of Biological Networks and Beyond

The major tool I use in this thesis is probabilistic graphical models. Probabilistic graphical models have several advantages in integrating information. First, they have the ability to represent complicated dependency structure that can't be captured by independent pair-wise relationship. Second, by taking into account the information encoded by the graph, graphical models enable learning from both labeled and unlabeled data. I plan to extend and apply the framework presented in this thesis to other areas in computational biology.

In this thesis I focused on identification of conserved or species-specific gene sets. However, genes usually work together to carry biological functions. The interaction and regulation of genes are better represented as networks. There are already some studies on cross-species analysis of protein-protein interaction networks [Sharan et al., 2005]. It is interesting to extend our framework into cross species analysis of regulation networks, to identify conserved and species-specific regulatory modules.

Higher order organisms such as humans have more than one type of cells. Although the different types of cells have the exactly the same genome, their may look and behave vastly different. One of the reasons for this difference is because the cells don't have the same epigenetic modifications. The information in epigenetic modifications controls the accessibility to the promoter of a gene, and thus regulates the gene's expression patterns. One challenge in the study of epigenetics is how to integrate data from different types of cells to infer the underlying epigenetic regulation code. I believe by developing new computational tools for data integration, we will be able to better understand the epigenetic programs.

The accelerated accumulation of expression and sequence data provides great opportunities for cross-species study of biological systems, but also poses new computational challenges. Large datasets often impose higher computational cost and thus it is crucial to develop fast learning and inference algorithms. Some new efficient methods have been proposed in other areas, such as using approximation to speed up computation [Potetz 2007], using asynchronous message passing for faster convergence of belief propagation [Elidan et al.], and using convex formulations [Wainwright 2005]. I am interested in developing and applying efficient algorithms to solve large-scale problems. Due to complicated interactions between biological properties, some problems may have heterogeneous correlation structure. It is interesting to develop models that can capture multiple (latent) classes of relationship, and learn the classes from data.

In sum, I believe our computational framework will play an important role in the cross-species study of biological systems, and I look forward to seeing more applications of the framework to open problems.

Bibliography

M. Adams, S. Celniker, R. Holt, C. Evans, J. Gocayne, P. Amanatides, S. Scherer, P. Li, R. Hoskins, R. Galle, et al. The Genome Sequence of Drosophila melanogaster. *Science*, 287(5461):2185–2195, 2000.

A. Aderem and R. Ulevitch. Toll-like receptors in the induction of the innate immune response. *Nature*, 406(6797):782–7, 2000.

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

S. Altschul et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, 1997.

S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic local alignment search tool. *J. Mol. Biol*, 215(3):403–410, 1990.

Arabidopsis Genome Initiative. Analysis of the genome sequence of the flowering plant Arabidopsis thaliana. *Nature*, 408(6814):796–815, 2000.

M. Ashburner, C. Ball, J. Blake, D. Botstein, H. Butler, J. Cherry, A. Davis, K. Dolinski, S. Dwight, J. Eppig, et al. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

J. Bähler. Cell-Cycle Control of Gene Expression in Budding and Fission Yeast. *Annu. Rev. Genet*, 39:69–94, 2005.

O. Banerjee and L. El Ghaoui. Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data. *The Journal of Machine Learning Research*, 9:485–516, 2008.

Z. Bar-Joseph, S. Farkash, D. Gifford, I. Simon, and R. Rosenfeld. Deconvolving cell cycle expression data with complementary information. *Bioinformatics*, 20 Suppl 1:I23–I30, 2004.

Z. Bar-Joseph, G. Gerber, T. Lee, N. Rinaldi, J. Yoo, F. Robert, D. Gordon, E. Fraenkel, T. Jaakkola, R. Young, et al. Computational discovery of gene modules and regulatory networks. *Nature Biotechnology*, 21(11):1337–1342, 2003.

S. Bergmann, J. Ihmels, and N. Barkai. Similarities and differences in genome-wide expression data of six organisms. *PLoS Biol.*, 2(1):e9, 2004.

B. Beutler. Inferences, questions and possibilities in Toll-like receptor signalling. *Nature*, 430:257–263, 2004.

C. Bishop, editor. *Pattern Recognition and Machine Learning*. Springer, 2006.

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. *Proc. 18th International Conf. on Machine Learning*, pages 19–26, 2001.

J. C. Boldrick, A. A. Alizadeh, M. Diehn, S. Dudoit, C. L. Liu, C. E. Belcher, D. Botstein, L. M. Staudt, P. O. Brown, and D. A. Relman. Stereotyped and specific gene expression programs in human innate immune responses to bacteria. *PNAS*, 99(2):972–977, 2002.

P. Bork, T. Dandekar, Y. Diaz-Lazcoz, F. Eisenhaber, M. Huynen, and Y. Yuan. Predicting function: from genes to genomes and back. *J. Mol. Biol*, 283(4):707–725, 1998.

Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, pages 1124–1137, 2004.

Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–655, 1998.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

J. Bratt and J. Palmblad. Cytokine-induced neutrophil-mediated injury of human endothelial cells. *The Journal of Immunology*, 159(2):912–918, 1997.

L. Breeden and K. Nasmyth. Cell cycle control of the yeast HO gene: cis-and trans-acting regulators. *Cell*, 48(3):389–97, 1987.

P. Brown and D. Botstein. Exploring the new world of the genome with DNA microarrays. *Nature Genetics*, 21:33–37, 1999.

H. Cam, E. Balciunaite, A. Blais, A. Spektor, R. Scarpulla, R. Young, Y. Kluger, and B. Dynlacht. A common set of gene regulatory networks links metabolism and growth inhibition. *Mol Cell*, 16(3):399–411, 2004.

D. Chaussabel, R. T. Semnani, M. A. McDowell, D. Sacks, A. Sher, and T. B. Nutman. Unique gene expression profiles of human macrophages and dendritic cells to phylogenetically distinct parasites. *Blood*, 102(2):672–681, 2003.

S. Dalton and L. Whitbread. Cell cycle-regulated nuclear import and export of Cdc47, a protein essential for initiation of DNA replication in budding yeast. *Proceedings of the National Academy of Sciences of the United States of America*, 92(7):2514, 1995.

J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. *ICML*, 2006.

U. de Lichtenberg, L. J. Jensen, A. Fausboll, T. S. Jensen, P. Bork, and S. Brunak. Comparison of computational methods for the identification of cell cycle-regulated genes. *Bioinformatics*, 21(7):1164–1171, 2005.

C. Debouck and P. Goodfellow. DNA microarrays in drug discovery and development. *Nature Genetics*, 21:48–50, 1999.

V. Deshpande, M. Grote, P. Messmer, and W. Sawyer. Parallel implementation of a sparse approximate inverse preconditioner.

C. S. Detweiler, D. B. Cunanan, and S. Falkow. Host microarray analysis reveals a role for the Salmonella response regulator phoP in human macrophage cell death. *PNAS*, 98 (10):5850–5855, 2001.

D. W. Draper, H. N. Bethea, and Y.-W. He. Toll-like receptor 2-dependent and -independent activation of macrophages by group b streptococci. *Immunology Letters*, 2006. URL `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&li%st_uids=16242782&dopt=Citation`.

I. Dubchak, M. Brudno, G. Loots, L. Pachter, C. Mayor, E. Rubin, and K. Frazer. Active Conservation of Noncoding Sequences Revealed by Three-Way Species Comparisons, 2000.

L. Duret, F. Dorkeld, and C. Gautier. Strong conservation of non-coding sequences during vertebrates evolution: potential involvement in post-transcriptional regulation of gene expression. *Nucleic Acids Res*, 21(10):2315–2322, 1993.

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

A. Enright, S. Van Dongen, and C. Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Research*, 30(7):1575–1584, 2002.

J. Eppig, C. Bult, J. Kadin, J. Richardson, and J. Blake. The Mouse Genome Database (MGD): from genes to mice–a community resource for mouse biology. *Nucleic Acids Research*, 33(Database Issue):D471, 2005.

J. Ernst and Z. Bar-Joseph. STEM: a tool for the analysis of short time series gene expression data. *BMC Bioinformatics*, 7(1):191, 2006.

B. Fischer, J. Grossmann, V. Roth, W. Gruissem, S. Baginsky, and J. Buhmann. Semi-supervised LC/MS alignment for differential proteomics. *Bioinformatics*, 22(14), 2006.

L. Ford and D. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399–404, 1956.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432, 2008.

A. Gasch, P. Spellman, C. Kao, O. Carmel-Harel, M. Eisen, G. Storz, D. Botstein, and P. Brown. Genomic Expression Programs in the Response of Yeast Cells to Environmental Changes. *Molecular Biology of the Cell*, 11(12):4241–4257, 2000.

A. Gavin, P. Aloy, P. Grandi, R. Krause, M. Boesche, M. Marzioch, C. Rau, L. Jensen, S. Bastuck, B. Dümpelfeld, et al. Proteome survey reveals modularity of the yeast cell machinery. *Nature*, 440:631–636, 2006.

A. Goffeau, B. Barrell, H. Bussey, R. Davis, B. Dujon, H. Feldmann, F. Galibert, J. Hoheisel, C. Jacq, M. Johnston, et al. Life with 6000 Genes. *Science*, 274(5287):546–567, 1996.

F. Granucci, C. Vizzardelli, N. Pavelka, S. Feau, M. Persico, E. Virzi, M. Rescigno, G. Moro, and P. Ricciardi-Castagnoli. Inducible il-2 production by dendritic cells revealed by global gene expression analysis. *Nature Immunology*, 2001.

H. Grassmé, V. Jendrossek, and E. Gulbins. Molecular mechanisms of bacteria induced apoptosis. *Apoptosis*, 6(6):441–445, 2001.

S. Grossmann, S. Bauer, P. Robinson, and M. Vingron. An improved statistic for detecting over-represented gene ontology annotations in gene sets. *RECOMB*, pages 85–98, 2006.

M. Grote and T. Huckle. Parallel Preconditioning with Sparse Approximate Inverses. *SIAM JOURNAL ON SCIENTIFIC COMPUTING*, 18:838–853, 1997.

C. Harbison, D. Gordon, T. Lee, N. Rinaldi, K. Macisaac, T. Danford, N. Hannett, J. Tagne, D. Reynolds, J. Yoo, et al. Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431:99–104, 2004.

D. Hardie, D. Carling, and M. Carlson. THE AMP-ACTIVATED/SNF 1 PROTEIN KINASE SUBFAMILY: Metabolic Sensors of the Eukaryotic Cell? *Annual Review of Biochemistry*, 67(1):821–855, 1998.

R. Hardison. Comparative Genomics. *PLoS Biol*, 1:2, 2003.

G. Hertz, G. Hartzell, and G. Stormo. Identification of consensus patterns in unaligned DNA sequences known to be functionally related. *Bioinformatics*, 6(2):81–92, 1990.

J. Hoffmann, F. Kafatos, C. Janeway, and R. Ezekowitz. Phylogenetic perspectives in innate immunity. *Science*, 284(5418):1313–8, 1999.

R. Hoffmann, K. van Erp, K. Trulzsch, and J. Heesemann. Transcriptional responses of murine macrophages to infection with yersinia enterocolitica. *Cellular Microbiology*, 6 (4):377–390, 2004.

L. Holm and C. Sander. Mapping the protein universe. *Science*, 273(5275):595–603, 1996.

Q. Huang, D. Liu, P. Majewski, L. C. Schulte, J. M. Korn, R. A. Young, E. S. Lander, and N. Hacohen. The Plasticity of Dendritic Cell Responses to Pathogens and Their Components. *Science*, 294(5543):870–875, 2001.

A. Ihler, J. Fisher, and A. Willsky. Loopy Belief Propagation: Convergence and Effects of Message Errors. *JOURNAL OF MACHINE LEARNING RESEARCH*, 6(1):905, 2006.

J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai. Revealing modular organization in the yeast transcriptional network. *Nat Genet*, 31(4):370–377, 2002.

J. Javerzat, G. Cranston, and R. Allshire. Fission yeast genes which disrupt mitotic chromosome segregation when overexpressed. *Nucleic Acids Res*, 24(23):4676–4683, 1996.

L. J. Jensen, T. S. Jensen, U. de Lichtenberg, S. Brunak, and P. Bork. Co-evolution of transcriptional and post-translational cell-cycle regulation. *Nature*, 2006.

G. Jones. Cellular signaling in macrophage migration and chemotaxis. *Journal of Leukocyte Biology*, 68(5):593, 2000.

G. Keller and R. Snodgrass. Life span of multipotential hematopoietic stem cells in vivo. *Journal of Experimental Medicine*, 171(5):1407–1418, 1990.

J. Kelley, B. de Bono, and J. Trowsdale. IRIS: a database surveying known human immune system genes. *Genomics*, 85(4):503–11, 2005.

M. Kellis, N. Patterson, M. Endrizzi, B. Birren, and E. S. Lander. Sequencing and comparison of yeast species to identify genes and regulatory elements. *Nature*, 423:241–254, May 2003. ISSN 0028-0836. 10.1038/nature01644.

N. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. Tikuisis, et al. Global landscape of protein complexes in the yeast Saccharomyces cerevisiae. *Nature*, 440:637–643, 2006.

K. Lammers, P. Brigidi, B. Vitali, P. Gionchetti, F. Rizzello, E. Caramelli, D. Matteuzzi, and M. Campieri. Immunomodulatory effects of probiotic bacteria DNA: IL-1 and IL-10 response in human peripheral blood mononuclear cells. *FEMS Immunology and Medical Microbiology*, 38(2):165–172, 2003.

R. Lang, D. Patel, J. J. Morris, R. L. Rutschman, and P. J. Murray. Shaping Gene Expression in Activated and Resting Primary Macrophages by IL-10. *J Immunol*, 169(5): 2253–2263, 2002.

S. Leem, C. Chung, Y. Sunwoo, and H. Araki. Meiotic role of SWI6 in Saccharomyces cerevisiae. *Nucleic Acids Research*, 26(13):3154–3158.

B. Lemaitre, E. Nicolas, L. Michaut, J. Reichhart, and J. Hoffmann. The Dorsoventral Regulatory Gene Cassette spätzle/Toll/cactus Controls the Potent Antifungal Response in Drosophila Adults. *Developmental Cell*, 5(3):441–450, 2003.

X. Liu, D. Brutlag, and J. Liu. BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pac. Symp. Biocomput*, 6:127–138, 2001.

D. Lockhart, H. Dong, M. Byrne, M. Follettie, M. Gallo, M. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Norton, et al. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 14:1675–1680, 1996.

X. Lu, W. Zhang, Z. Qin, K. Kwast, , and J. Liu. Statistical resynchronization and bayesian detection of periodically expressed genes. *Nucl. Acids. Res.*, 32:447–455, 2004.

Y. Lu, S. Mahony, P. Benos, R. Rosenfeld, I. Simon, L. Breeden, and Z. Bar-Joseph. Combined analysis reveals a core set of cycling genes. *Genome Biology*, 8(7):R146, 2007.

Y. Lu, R. Rosenfeld, and Z. Bar-Joseph. Identifying cycling genes by combining sequence homology and expression data. *Bioinformatics*, 22(14):e314–322, 2006.

N. Lukacs, R. Strieter, S. Chensue, M. Widmer, and S. Kunkel. TNF-alpha mediates recruitment of neutrophils and eosinophils during airway inflammation. *The Journal of Immunology*, 154(10):5411–5417, 1995.

S. Mahamud. Comparing belief propagation and graph cuts for novelty detection. *Proc. Conf. Comp. Vision Pattern Rec*, 2006.

S. Mahony, A. Golden, T. Smith, and P. Benos. Improved detection of DNA motifs using a self-organized clustering of familial binding profiles. *Bioinformatics*, 21(1):283–291, 2005a.

S. Mahony, D. Hendrix, A. Golden, T. Smith, and D. Rokhsar. Transcription factor binding site identification using the self-organizing map. *Bioinformatics*, 21(9):1807–1814, 2005b.

R. L. McCaffrey, P. Fawcett, M. O'Riordan, K.-D. Lee, E. A. Havell, P. O. Brown, and D. A. Portnoy. From the Cover: A specific gene expression program triggered by Gram-positive bacteria in the cytosol. *PNAS*, 101(31):11386–11391, 2004.

M. Menges, L. Hennig, W. Gruissem, and J. Murray. Cell Cycle-regulated Gene Expression in Arabidopsis. *Journal of Biological Chemistry*, 277(44):41987–42002, 2002.

H. 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(1):31, 2002.

H. Mizutani, N. Schechter, G. Lazarus, R. Black, and T. Kupper. Rapid and specific conversion of precursor interleukin 1 beta (IL-1 beta) to an active IL-1 species by human mast cell chymase. *Journal of Experimental Medicine*, 174(4):821–825, 1991.

J. Mooij and H. Kappen. Sufficient Conditions for Convergence of the Sum–Product Algorithm. *Information Theory, IEEE Transactions on*, 53(12):4422–4437, 2007.

M. Mukherji, R. Bell, L. Supekova, Y. Wang, A. P. Orth, S. Batalov, L. Miraglia, D. Huesken, J. Lange, C. Martin, S. Sahasrabudhe, M. Reinhardt, F. Natt, J. Hall, C. Mickanin, M. Labow, S. K. Chanda, C. Y. Cho, and P. G. Schultz. Genome-wide functional analysis of human cell-cycle regulators. *PNAS*, 103(40):14819–14824, 2006.

P. Nagpal, C. M. Ellis, H. Weber, S. E. Ploense, L. S. Barkawi, T. J. Guilfoyle, G. Hagen, J. M. Alonso, J. D. Cohen, E. E. Farmer, J. R. Ecker, and J. W. Reed. Auxin response factors ARF6 and ARF8 promote jasmonic acid production and flower maturation. *Development*, 132(18):4107–4118, 2005.

K. Nasmyth and L. Dirick. The role of SWI4 and SWI6 in the activity of G1 cyclins in yeast. *Cell*, 66(5):995–1013, 1991.

K. Natarajan, M. Meyer, B. Jackson, D. Slade, C. Roberts, A. Hinnebusch, and M. Marton. Transcriptional Profiling Shows that Gcn4p Is a Master Regulator of Gene Expression during Amino Acid Starvation in Yeast. *Molecular and Cellular Biology*, 21(13):4347, 2001.

G. Nau, J. Richmond, A. Schlesinger, E. Jennings, E. Lander, and R. Young. Human macrophage activation programs induced by bacterial pathogens. *Proceedings of the National Academy of Sciences*, page 22649799, 2002.

W. Navarre and A. Zychlinsky. Pathogen-induced apoptosis of macrophages: a common end for different pathogenic strategies. *Cellular Microbiology*, 2(4):265–273, 2000.

P. Nurse. Universal control mechanism regulating onset of M-phase. *Nature*, 344:503–508, 1990.

A. Oliva, A. Rosebrock, F. Ferrezuelo, S. Pyne, H. Chen, S. Skiena, B. Futcher, and J. Leatherwood. The Cell Cycle–Regulated Genes of Schizosaccharomyces pombe. *PLoS Biol*, 3(7):e225, 2005.

S. Park, K. Ewalt, and S. Kim. Functional expansion of aminoacyl-tRNA synthetases and their interacting factors: new perspectives on housekeepers. *Trends in Biochemical Sciences*, 30(10):569–574, 2005.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

J. Pedraza and A. van Oudenaarden. Noise Propagation in Gene Networks, 2005.

X. Peng, R. Karuturi, L. Miller, K. Lin, Y. Jia, P. Kondu, L. Wang, L. Wong, E. Liu, M. Balasubramanian, et al. Identification of Cell Cycle-regulated Genes in Fission Yeast D in Box. *Mol Biol Cell*, 16(3):1026–1042, 2005.

C. Penkett, J. Morris, V. Wood, and J. Bahler. YOGY: a web-based, integrated database to retrieve protein orthologs and associated Gene Ontology terms. *Nucleic Acids Research*, 34(Web Server issue):W330, 2006.

T. Pramila, W. Wu, S. Miles, W. S. Noble, and L. L. Breeden. The Forkhead transcription factor Hcm1 regulates chromosome segregation genes and fills the S-phase gap in the transcriptional circuitry of the cell cycle. *Genes Dev.*, 20(16):2266–2278, 2006.

A. Provenzani, R. Fronza, F. Loreni, A. Pascale, M. Amadio, and A. Quattrone. Global alterations in mRNA polysomal recruitment in a cell model of colorectal cancer progression to metastasis. *Carcinogenesis*, 2006.

C. Rao, D. Wolf, and A. Arkin. Control, exploitation and tolerance of intracellular noise. *Nature*, 420(6912):231–237, 2002.

B. Ren, H. Cam, Y. Takahashi, T. Volkert, J. Terragni, R. Young, and B. Dynlacht. E2F integrates cell cycle progression with DNA repair, replication, and G2/M checkpoints. *Genes & Development*, 16(2):245–256, 2002.

M. Rojas. Differential induction of apoptosis by virulent Mycobacterium tuberculosis in resistant and susceptible murine macrophages: role of nitric oxide and mycobacterial products. *The Journal of Immunology*, 159(3):1352–1361, 1997.

F. Roth, J. Hughes, P. Estep, and G. Church. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nature Biotechnology*, 16:939–945, 1998.

J. Rual, K. Venkatesan, T. Hao, T. Hirozane-Kishikawa, A. Dricot, N. Li, G. Berriz, F. Gibbons, M. Dreze, N. Ayivi-Guedehoussou, et al. Towards a proteome-scale map of the human protein- protein interaction network. *Nature*, 437:1173–1178, 2005.

G. Rustici, J. Mata, K. Kivinen, P. Lió, C. Penkett, G. Burns, J. Hayles, A. Brazma, P. Nurse, and J. Bähler. Periodic gene expression program of the fission yeast cell cycle. *Nature Genetics*, 36:809–817, 2004.

M. Schena, D. Shalon, R. Davis, and P. Brown. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270(5235):467–470, 1995.

J. Schuchhardt, D. Beule, A. Malik, E. Wolski, H. Eickhoff, H. Lehrach, and H. Herzel. Normalization strategies for cDNA microarrays. *Nucleic Acids Research*, 28(10):E47–e47, 2000.

R. Sharan, S. Suthram, R. Kelley, T. Kuhn, S. McCuine, P. Uetz, T. Sittler, R. Karp, and T. Ideker. From the Cover: Conserved patterns of protein interaction in multiple species. *Proceedings of the National Academy of Sciences*, 102(6):1974, 2005.

R. Shyamsundar, Y. Kim, J. Higgins, K. Montgomery, M. Jorden, A. Sethuraman, M. van de Rijn, D. Botstein, P. Brown, and J. Pollack. A DNA microarray survey of gene expression in normal human tissues. *Genome Biology*, 2005(6):R22, 2005.

I. Simon, J. Barnett, N. Hannett, C. Harbison, N. Rinaldi, T. Volkert, J. Wyrick, J. Zeitlinger, D. Gifford, T. Jaakkola, and R. Young. Serial regulation of transcriptional regulators in the yeast cell cycle. *Cell*, 106:697–708, 2001.

P. Spellman, G. Sherlock, M. Zhang, V. Iyer, K. Anders, M. Eisen, P. Brown, D. Botstein, and B. Futcher. Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization. *Molecular Biology of the Cell*, 9:3273–3297, 1998.

L. Stein, Z. Bao, D. Blasiar, T. Blumenthal, M. Brent, N. Chen, A. Chinwalla, L. Clarke, C. Clee, A. Coghlan, et al. The genome sequence of Caenorhabditis briggsae: A platform for comparative genomics. *PLoS Biol*, 1(2):166–192, 2003.

J. M. Stuart, E. Segal, D. Koller, and S. K. Kim. A gene-coexpression network for global discovery of conserved genetic modules. *Science*, 302(5643):249–55, 2003.

Z. Sun and R. Andersson. NF-[kappa] B Activation and Inhibition: A Review. *Shock*, 18 (2):99, 2002.

M. Tappen and W. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 900–906, 2003.

S. Tatikonda and M. Jordan. Loopy belief propagation and Gibbs measures. *Uncertainty in Artificial Intelligence*, 18:493–500, 2002.

D. Thomas and Y. Surdin-Kerjan. Metabolism of sulfur amino acids in Saccharomyces cerevisiae. *Microbiol Mol Biol Rev*, 61(4):503–532, 1997.

A. Ureta-Vidal, L. Ettwiller, and E. Birney. Comparative genomics: genome-wide analysis in metazoan eukaryotes. *Nature Reviews Genetics*, 4(4):251–262, 2003.

G. Valbuena, W. Bradford, and D. Walker. Expression Analysis of the T-Cell-Targeting Chemokines CXCL9 and CXCL10 in Mice and Humans with Endothelial Infections Caused by Rickettsiae of the Spotted Fever Group, 2003.

K. van Erp, K. Dach, I. Koch, J. Heesemann, and R. Hoffmann. Role of strain differences on host resistance and the transcriptional response of macrophages to infection with Yersinia enterocolitica. *Physiol. Genomics*, 25(1):75–84, 2006.

J. Venter, M. Adams, E. Myers, P. Li, R. Mural, G. Sutton, H. Smith, M. Yandell, C. Evans, R. Holt, et al. The Sequence of the Human Genome. *Science*, 291(5507):1304–1351, 2001.

R. Waterston, K. Lindblad-Toh, E. Birney, J. Rogers, J. Abril, P. Agarwal, R. Agarwala, R. Ainscough, M. Alexandersson, P. An, et al. Initial sequencing and comparative analysis of the mouse genome. *Nature*, 420(6915):520–62, 2002.

M. Whitfield, G. Sherlock, A. Saldanha, J. Murray, C. Ball, K. Alexander, J. Matese, C. Perou, M. Hurt, P. Brown, et al. Identification of Genes Periodically Expressed in the Human Cell Cycle and Their Expression in Tumors. *Molecular Biology of the Cell*, 13: 1977–2000, 2002.

S. Wichert, K. Fokianos, and K. Strimmer. Identifying periodically expressed transcripts in microarray time series data. *Bioinformatics*, 20:5–20, 2004.

C. Wilson, J. Kreychman, and M. Gerstein. Assessing annotation transfer for genomics: quantifying the relations between protein sequence, structure and function through traditional and probabilistic scores. *J. Mol. Biol*, 297:233–249, 2000.

E. Winzeler, D. Shoemaker, A. Astromoff, H. Liang, K. Anderson, B. Andre, R. Bangham, R. Benito, J. Boeke, H. Bussey, et al. Functional Characterization of the S. cerevisiae Genome by Gene Deletion and Parallel Analysis. *Science*, 285(5429):901–906, 1999.

L. Wodicka, H. Dong, M. Mittmann, M. Ho, and D. Lockhart. Genome-wide expression monitoring in Saccharomyces cerevisiae. *Nature Biotechnology*, 15:1359–1367, 1997.

S. Wolpe, G. Davatelis, B. Sherry, B. Beutler, D. Hesse, H. Nguyen, L. Moldawer, C. Nathan, S. Lowry, and A. Cerami. Macrophages secrete a novel heparin-binding protein with inflammatory and neutrophil chemokinetic properties. *Journal of Experimental Medicine*, 167(2):570–581, 1988.

X. Xie, J. Lu, E. J. Kulbokas, T. R. Golub, V. Mootha, K. Lindblad-Toh, E. S. Lander, and M. Kellis. Systematic discovery of regulatory motifs in human promoters and 3[prime] utrs by comparison of several mammals. *Nature*, 434:338–345, March 2005. ISSN 0028-0836. 10.1038/nature03441.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring Artificial Intelligence in the New Millennium*, (Chap. 8): 239–236, January 2003.

X. Zhou, M. Kao, and W. Wong. From the Cover: Transitive functional annotation by shortest-path analysis of gene expression data. *Proceedings of the National Academy of Sciences*, 99(20):12783, 2002.

X. Zhu. Semi-supervised learning with graphs (doctoral thesis). Technical report, CMU, 2005.