Computational Discovery of Gene Modules, Regulatory Networks and Expression Programs

by

Georg Kurt Gerber

B.A., Mathematics, UC Berkeley (1991)
M.P.H., Infectious Diseases, UC Berkeley (1994)
S.M., Electrical Engineering and Computer Science, MIT (2003)

Submitted to the Harvard-MIT Division of Health Sciences and Technology
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Medical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2007

Author . . . .
Harvard-MIT Division of Health Sciences and Technology
July 1, 2007

Certified by . . . .
David K. Gifford
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . .
Martha L. Gray
Edward Hood Taplin Professor of Medical and Electrical Engineering
Co-Director, Harvard-MIT Division of Health Sciences and Technology

Computational Discovery of Gene Modules, Regulatory Networks and Expression Programs

by
Georg Kurt Gerber

Submitted to the Harvard-MIT Division of Health Sciences and Technology
on July 1, 2007, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Medical Engineering

Abstract

High-throughput molecular data are revolutionizing biology by providing massive amounts of information about gene expression and regulation. Such information is applicable both to furthering our understanding of fundamental biology and to developing new diagnostic and treatment approaches for diseases. However, novel mathematical methods are needed for extracting biological knowledge from high-dimensional, complex and noisy data sources. In this thesis, I develop and apply three novel computational approaches for this task. The common theme of these approaches is that they seek to discover meaningful groups of genes, which confer robustness to noise and compress complex information into interpretable models. I first present the GRAM algorithm, which fuses information from genome-wide expression and *in vivo* transcription factor-DNA binding data to discover regulatory networks of gene modules. I use the GRAM algorithm to discover regulatory networks in *Saccharomyces cerevisiae*, including rich media, rapamycin, and cell-cycle module networks. I use functional annotation databases, independent biological experiments and DNA-motif information to validate the discovered networks, and to show that they yield new biological insights. Second, I present GeneProgram, a framework based on Hierarchical Dirichlet Processes, which uses large compendia of mammalian expression data to simultaneously organize genes into overlapping programs and tissues into groups to produce maps of expression programs. I demonstrate that GeneProgram outperforms several popular analysis methods, and using mouse and human expression data, show that it automatically constructs a comprehensive, body-wide map of inter-species expression programs. Finally, I present an extension of GeneProgram that models temporal dynamics. I apply the algorithm to a compendium of short time-series gene expression experiments in which human cells were exposed to various infectious agents. I show that discovered expression programs exhibit temporal pattern usage differences corresponding to classes of host cells and infectious agents, and describe several programs that implicate surprising signaling pathways and receptor types in human responses to infection.

Thesis Supervisor: David K. Gifford
Title: Professor of Electrical Engineering and Computer Science

Biographical information

Personal

Born: December 31, 1970 in Los Angeles, CA.

Education

B.A., Mathematics, *summa cum laude*, UC Berkeley, 1991.
M.P.H., Infectious Diseases, UC Berkeley, 1994.
S.M., Electrical Engineering and Computer Science, MIT, 2003.

Professional experience

- Graduate Research Assistant, MIT Computer Science and Artificial Intelligence Laboratory (Supervisor: Prof. David Gifford), 2000–present.
- Instructor, Northeastern University Bioinformatics Essentials Graduate Certificate Program, 2002.
- Chief Technology Officer, IS?TV, Inc., 2000.
- Senior Vice President, L-Squared Entertainment, Inc., 1995–1999.
- President and Chief Executive Officer, Infinite Interactions, Inc., 1993–1995.
- Graduate Research Assistant, UC Berkeley Department of Computer Science (Supervisor: Prof. Brian Barsky), 1993–1994.
- Research Associate/Laboratory Automations Specialist, Metabolex, Inc., 1992.
- Undergraduate Teaching Assistant, UC Berkeley Department of Mathematics, 1991.
- Undergraduate Research Assistant, UC Berkeley Department of Biophysics (Supervisor: Prof. Hans Bremermann), 1990.
- President, Speak Easy Communications System, 1984–1989.

Academic awards and honors

- National Defense Science and Engineering Graduate (NDSEG) Fellowship, 2002–2005.
- Sigma Xi honor society of science and engineering, Associate Member, elected 2003.
- National Science Foundation (NSF) Fellowship (awarded), 2002.

- University of California Regents' Graduate Fellowship, 1992–1994.
- UC Berkeley Highest Distinction in General Scholarship in Mathematics, 1991.
- Phi Beta Kappa honor society, elected 1990.
- University of California Regents' Undergraduate Scholarship, 1989–1991.
- UC Berkeley Dean's List, every semester attended, 1989–1991.
- J&M Seitz Scholarship, 1989.

Acknowledgments

"It was cold consolation to think that I, who looked upon it with my eyes and fondled it with my ten flesh-and-bone fingers, was no less monstrous than the book. I felt it was a nightmare thing, an obscene thing, and that it defiled and corrupted reality. I considered fire, but I feared that the burning of an infinite book might be similarly infinite, and suffocate the planet in smoke."

—Jorge Luis Borges, The Book of Sand

IN SCIENTIFIC papers, we boil our work down into figures, tables and equations. At the end of most papers, there is a tiny section, usually one to two sentences, called "acknowledgements." It's unfortunate that section is so short, because it often contains the keys to the real story behind the science—a story about people. In the seven years I've spent at MIT and Harvard, I've come to realize that science ultimately springs from the incredibly rich and complex interactions we have with colleagues, friends, family and society at large. So, in this section I will try to reveal a little of the real story behind this thesis.

However, I must say upfront that the origins of the most mathematical portions of this work remain more mysterious to me. On those occasions, when traces of infinity shook loose from equations, I credit primarily twilight, the lizards, and the way that light in the mid-summer late in the afternoon sometimes spreads across the bare walls of old houses.

First, I must acknowledge the most direct collaborators on this work. Ziv Bar-Joseph co-developed the GRAM algorithm with me when he was a fellow graduate student at MIT. Ziv has been more than just a fantastic intellectual collaborator; he has been a true friend. Robin Dowell, a post-doc in the Gifford lab, was also a great collaborator. She worked with me on various aspects of GeneProgram, particularly the cross-species analysis. Robin is not only a talented researcher, but she's also a gifted conversationalist. Over the last two years, I don't know how many hours we've spent talking about everything from molecular evolution to car engines.

Next, I must acknowledge the members of my thesis committee, who have guided my work for almost seven years. Professor David Gifford, my thesis supervisor, introduced me to the field of computational functional genomics and has been instrumental in encouraging my research by providing ideas, resources, and guidance while also fostering the critical collaborative relationships outside of our immediate group. I think that part of the reason that Professor Gifford and I have worked well together is that we share an entrepreneurial spirit. He is always starting new projects, and investigating novel technologies, research ideas and collaborations. Moreover, he's not afraid to step outside the boundaries of traditional academic disciplines, and he's been particularly encouraging of my pursuit of a medical degree simultaneously with my Ph.D.

Professor Tommi Jaakkola introduced me to many topics in machine learning and statistics, and suggested a number of the core computational ideas in my research. He's one of the smartest people I've ever encountered, and his deep knowledge of his field is truly inspirational. Also, Professor Jaakkola is one of the nicest and most encouraging faculty members I've ever met. Despite his busy schedule, he always made time to talk and always treated me as a collaborator. Even when research wasn't going well, Professor Jaakkola never failed to remind me of the true purpose of the academic enterprise—teaching, learning and developing new ideas.

Professor Richard Young has provided much inspiration for the biological parts of this thesis. His passion for biology and unlocking the secrets of genetic regulation is infectious. Professor Young has also been critical in fostering true collaborations between biologists in his lab and computer scientists in my group.

I must also acknowledge a number of graduate students and post-docs in Professor Gifford, Jaakkola, and Young's labs. John Barnett, Tim Danford, Tong Lee, Chris Reeder, Kenzie MacIsaac, Reina Riemann, Alex Rolfe, Alan Qi and Nicola Rinaldi have let me bounce ideas off them, have provided critical comments on manuscripts and presentations, and have been great all-around friends!

Finally, I must acknowledge my family. My brother Karl was my constant companion throughout childhood and it was with him that I set out one September, long ago, for Los Angeles Valley College, where I think the whole idea about getting a Ph.D. got started. My sister Margot supposedly first suggested the Valley College plan, so she also shares some part in the saga. Then, of course, there are my parents. My father Barry first introduced me to computers, possibly as early as 1974. Some of my first memories are of my father, brother and me playing Wumpus together on an old terminal. Undoubtedly, it was those experiences that sparked my interest in computers and programming. As far as I can tell, most of what my father knows he's taught himself, and I must credit him for passing on his drive for learning independently, which has served me well as a researcher. My other earliest memories are of my mother reading to me about dinosaurs, reptiles and pre-historic cultures. To her, it is the world of people and other living things that matters most, and I think she always knew I would end up studying biology and medicine someday. She has supported every step of my academic odyssey from the baked bungalows of Valley College, to the misty hills of Berkeley, to the Infinite Corridor of MIT, to the stone quad of Harvard Medical School.

Finally, I must acknowledge my wife, Anne-Marie. I fell in love with this smart, somewhat offbeat, pretty, brown-haired, blue-eyed girl more than a decade ago. Since then, she has shared my long, strange journey from Hollywood executive to computational biology researcher and medical student. Although this path has not always been an easy one to follow, we have traveled it together, with hands joined, and only in that way has it been possible to make it out of the labyrinth.

Previously published work and collaborations

The section describing DNA microarray technologies in Chapter 1 is an updated version of material that originally appeared in my Masters' thesis [73]. Figure 1-1 is a modified version of a figure that I created for [170].

The work in Chapter 2 (the GRAM algorithm) was done in collaboration with Ziv Bar-Joseph and Tong Lee. Ziv Bar-Joseph and I developed the GRAM algorithm jointly. Tong Lee directly performed or supervised the new biological experiments described in the Chapter 2. The text and figures in Chapter 2 come from two previously published papers [118, 21] and material that also appeared in Ziv Bar-Joseph's Ph.D. thesis [16].

Robin Dowell collaborated on various aspects of the GeneProgram work presented in Chapter 3. In particular, she was instrumental in creating the mouse-human gene mapping, processing the expression data, and interpreting the discovered gene programs.

A publication based on portions of the GeneProgram and GeneProgram++ work (Chapters 3 and 4) is scheduled to appear in [76].

A note on notation

An effort has been made to use consistent notation throughout this thesis. Scalars will be denoted with italic Roman or Greek lower-case letters such as x and y or μ. All vectors will be column vectors and denoted with bold italic Roman or Greek lower-case letters such as $\boldsymbol{x}$ and $\boldsymbol{y}$ or $\boldsymbol{\mu}$. Matrices will be denoted with bold italic Roman or Greek upper-case letters such as $\boldsymbol{A}$ and $\boldsymbol{B}$ or $\boldsymbol{\Gamma}$. Sets will be denoted with upper-case Roman italic letters such as C and T. Random variables will be denoted by fonts without serifs such as x and y for random scalars and $\mathbf{\mathsf{x}}$ and $\mathbf{\mathsf{y}}$ for random vectors. Unfortunately, lower-case Greek letters are also used for random variables but the sans serif style is not available. In these cases, the intention should be clear from the context.

To autumn and all that November may bring,
and to my wife, Anne-Marie, who faces that
wondrous mystery with me every day.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

"These ambiguities, redundancies, and deficiencies recall those attributed by Dr. Franz Kuhn to a certain Chinese encyclopedia entitled *Celestial Emporium of Benevolent Knowledge.* On these remote pages it is written that animals are divided into (a) those that belong to the Emperor, (b) embalmed ones, (c) those that are trained, (d) suckling pigs, (e) mermaids, (f) fabulous ones, (g) stray dogs, (h) those that are included in this classification, (i) those that tremble as if they were mad, (j) innumerable ones, (k) those drawn with a very fine camel's brush, (l) others, (m) those that have just broken a flower vase, (n) those that resemble flies at a distance."

—*Jorge Luis Borges, Other Inquisitions*

THIS THESIS is about developing and applying new computer algorithms to discover meaningful groups of genes from large collections of high-throughput biological data. The problem is challenging, because the data are complex, high-dimensional and noisy. We have developed several algorithms that address these issues and will describe them in this thesis. However, before delving into computational details, it's worth considering the more philosophical question: why try to put genes into groups in the first place?

Categorizing and grouping things—particularly living organisms and their parts—is an ancient human pursuit[1]. However, as illustrated by the classification of animals

[1]The use of computers for grouping biological entities also has a relatively long history. The first such work was published in 1957, by medical researcher and microbiologist Peter Sneath, in a paper titled "The application of computers to taxonomy" [201]. In that work, Dr. Sneath described a hierarchical clustering method and computer program, which he applied to categorizing members of the bacterial genus *Chromobacterium* that cause a rare tropical infection.

in the *Celestial Emporium of Benevolent Knowledge*, not all groupings are good ones. The Greek philosopher Aristotle (384-322 BC) developed fairly sophisticated ideas about what constitutes good categories. Chief among his contributions in this field is the notion that a collection of traits, rather than a single characteristic, should be used to differentiate members of a group. In his work, *Historia Animalium*, he wrote:

> "The method then that we must adopt is to attempt to recognize the natural groups, following the indications afforded by the instincts of mankind, which led them for instance to form the class of Birds and the class of Fishes, each of which groups combines a multitude of differentiae, and is not defined by a single one as in dichotomy [78]."

Of course, Aristotle didn't quite figure everything out, and subsequent generations of philosophers and scientists have expanded on his ideas. Three important modern criteria for evaluating the inherent meaningfulness of groups are based on evolutionary, mathematical and cognitive perspectives.

From the evolutionary perspective, a system—from an individual cell to entire ecologies—can be divided into *modules*, which are relatively independent "building-blocks" comprised of functionally related components [37]. The inherent decomposability of a system into modules is best justified from the point of view of evolutionary fitness. Nobel Prize winner and complexity theorist Herbert Simon summarized the argument for modularity eloquently:

> "But if the effectiveness of design of each organ depends on the design of the organs with which it interacts, then there is no guarantee that improvements of one organ will not worsen the performance of others ... Suppose, instead, that the effectiveness of each organ depends very little on the design of others, provided that the inputs each requires are supplied by the others. Then, up to a scale factor, the design of each organ can be improved independently of what is happening in the others; and it is easy to show that fitness will rise much more rapidly than when there is mutual dependence of design [37]."

From the mathematical perspective, good groupings of data provide *compression* of information without loss of important details. Mathematical techniques for compressing information enable many modern[2] technologies including cell-phones and digital video [47]. The basic idea behind many compression techniques is to assign data items to groups, and then to replace individual items with statistics derived from the groups. In compression algorithms, one must generally trade-off the competing objectives of making the representation of the compressed data as small as possible with retaining as much information as possible [212]. From the practical standpoint, a compressed data set can serve as a useful summary of the original data. Further, if there is noise in data, compression can result in a smoother, more representative view of the information.

[2]Well, these are modern as of 2007!

From the cognitive perspective, good groupings of data boost the human brain's limited ability to understand the natural world. To study the significance of mental categories, the cognitive psychologist Eleanor Rosch did field experiments with the Dugum Dani people of Papua New Guinea in the 1970's [180]. The Dani people lacked words describing color hues other than dark and light. However, when Rosch showed them colored chips of various hues, they were able to sort them into natural "perceptual categories." Rosch later expanded her experiments to study perception of other categories, such as shape and semantics, and in different populations, including American children [181, 182]. Overall, Rosch's experiments and those of others in her field suggest that the human tendency to categorize things is critical for our understanding of the world [132]. Rosch wrote that:

> "Categorization occurs ... to reduce the limitless variation ... to manageable proportions ... categories would ... follow the lines of natural correlations of attributes, those that maximize the correlation and thus the predictability of attributes within categories (quoted in [132])."

Later in this thesis, we will revisit the three perspectives discussed above. In particular, we'll describe how the sets of genes discovered using the algorithms developed in this thesis exhibit properties of modularity, represent mathematically principled compressions of data, and help us to understand better both model organisms and human biology.

The work we will present focuses on analysis of data gathered using a major new technique in experimental biology, DNA microarrays, which allow researchers to measure thousands of functional characteristics of a biological system simultaneously. In the next section, we will provide an overview of DNA microarray technologies and discuss some of their limitations, in order to provide context for the work presented in this thesis.

1.1 DNA Microarrays

1.1.1 A very brief overview of molecular genetics

In this subsection, we will discuss the bare essentials of molecular genetics for those unfamiliar with the subject. For more information, please consult a recent text such as [130].

Essential molecules for life: DNA, RNA and proteins

Much of cellular behavior arises from the interactions between three types of molecules: DNA (deoxyribonucleic acid), RNA (ribonucleic acid), and proteins.

DNA may be thought of as the primary long-term information storage molecule in the cell. Often likened to the "blue-print" for the cell, DNA consists of a long backbone of alternating sugar and phosphate residues. Attached to each sugar residue is one of four nitrogen-containing bases: adenine (A), cytosine (C), guanine (G) and thymine (T). It is the linear ordering or *sequence* of these bases in the DNA molecule that encodes information. Non-covalent bonds between hydrogen atoms in adenine/thymine and cytosine/guanine allow these bases to form stable pairs. The bases that form such pairs are said to be *complementary*. It is this base-pairing that allows DNA to exist as a double-stranded molecule in most normal physiological situations. The two strands encode the same information but use complementary bases.

While RNA also serves as an information carrier in the cell, these molecules act in a more transitory manner than do DNA molecules. RNA is quite similar to DNA chemically, differing only in the type of sugar used (ribose instead of deoxyribose) and the use of the base uracil (U) instead of thymine. Unlike DNA, RNA molecules primarily exist in a single-stranded form, which is less stable and subject to cellular degradation. A primary function of cellular RNA is to act as the *messenger* molecules that carry information copied from DNA. This process of copying DNA to RNA is called *transcription*, and is carried out by pieces of cellular "machinery" called RNA polymerases. The single-stranded copies are called mRNA (messenger RNA) and serve to amplify the original DNA "signal." In keeping with the "blue-print" analogy, the RNA molecules are like working prints that have been photocopied from the original DNA-based plans.

Many mRNA copies are produced from the DNA, and the information encoded in these copies is then *translated* into proteins by elaborate pieces of cellular "machinery" called ribosomes. If mRNA copies are not made from a gene, then no protein will be produced and the gene is said to be silent or inactive. When the gene is transcribed into RNA copies, the mRNA is said to be *expressed*.

Proteins are the primary "work-horses" of the cell, serving in a wide variety of roles including formation of physical structures, communication with other cells, and catalysis of metabolic reactions. In keeping with the "blue-print" analogy, proteins are the wood, walls, windows, doors, and even hammers and cranes used for constructing the building! Proteins consist of chains of amino acids, and the sequence of these amino acids and subsequent molecular modifications enable the protein to fold into a complex three-dimensional structure. While only four different nucleic acids are used to encode information at the DNA or mRNA level, proteins use twenty different amino acids. A triplet genetic code is used, in which three nucleotides, called a *codon*, specify a single amino acid. Because there are four types of nucleotides used in DNA, there are $4^3 = 64$ possible codons. This allows for a degenerate genetic code, in which a single amino acid can be specified by any of several codons.

Genes and Regulation

Because proteins play such a fundamental role in the life of a cell, biologists are extremely interested in identifying those regions of DNA called *genes* that ultimately lead to functional proteins. The *genome* for an organism is its complete hereditary information encoded in DNA, including both genes and regions of DNA that do not code for proteins. A variety of computational and experimental techniques have been used to identify genes [203]. Surprisingly, in more advanced organisms, much DNA does not code for functional proteins. The explanations for this are complex and still not fully understood [176, 203]. For these and other reasons, discovering genes is by no means a straight-forward or error-free process. Further, even the definition of a gene is rather controversial and has been subject to revision in recent years [203]. Fortunately, there are organizations such as the National Center for Biotechnology Information that seek to compile databases of the latest consensus genetic data. Currently, information is publicly available for many organisms, including bacteria, yeast, worms, plants, rodents, and humans [147].

Genetic regulation—the control over when and where genes are active in the cell—is critical in all organisms. Because the ultimate products of genes are proteins that carry out an extremely diverse set of cellular tasks, lack of control of genetic activity can have disastrous consequences. For instance, in animals, many genes code for proteins involved in cellular growth and division. If these genes are inappropriately regulated, cellular growth may go unchecked, leading to malignant tumor formation. Because genetic regulation is so critical, its study comprises a large part of the field of molecular genetics.

The process of genetic regulation is very complex, and is not yet completely understood for even simple organisms such as bacterial viruses [167, 13]. On the most basic level, genetic regulation is carried out by transcription factors (specialized proteins) that act either to inhibit or activate transcription of genes. Many transcription factors bind to regions of DNA near the genes they regulate. In reality, the process of regulation is much more complicated, especially in more advanced organisms such as humans. Genetic regulation is known to involve combinatorial interactions among transcriptional factors, binding of non-coding RNA molecules, alterations of the three-dimensional chromatin structure of DNA, control of the rates of transcription and translation, and many other mechanisms [159, 137, 145, 65, 176, 167, 165].

1.1.2 The "new era" of genomics

Classical molecular genetics tended to focus on the understanding of individual genes, which often involved years of painstaking experimental work for each gene [187]. It has only been in the last several years, with the advent of new automated technologies, that biologists have begun to take a more holistic approach. This approach is sometimes called *systems biology*, or more specifically *genomics*, when the goal is to understand organisms at the level of their genomes. This field is generally divided into two sub-disciplines: *structural* and *functional* genomics. Structural genomics is primarily concerned with elucidating the sequences of genes and regulatory elements. The definition of functional genomics is a bit less clear, but an article on the topic by Philip Hieter and Mark Boguski [88] provides a useful viewpoint:

> "...functional genomics refers to the development and application of global (genome-wide or system-wide) experimental approaches to assess gene function by making use of the information and reagents provided by structural genomics. It is characterized by high throughput or large-scale experimental methodologies combined with statistical and computational

analysis of the results . . . Computational biology will perform a critical and expanding role in this area: whereas structural genomics has been characterized by data management, functional genomics will be characterized by mining the data sets for particularly valuable information."

One of the most valuable tools in functional genomics is the DNA microarray, which allows researchers to measure the expression levels of thousands of mRNAs simultaneously. Although there are a variety of DNA microarray technologies in use, all operate on the basic principle that complementary strands of nucleic acids can "recognize" or hybridize with each other with high specificity through base-pairing.

DNA microarrays consist of a large number of different DNA molecules, called "probes," which are immobilized on a solid surface in an ordered matrix. The positions of the DNA molecules on the array are predetermined, and the different arrayed pieces of DNA usually correspond to portions of genes or inter-genic regions of interest. For genome-wide expression analysis, messenger RNA from a group of cells is worked up to produce a sample of nucleic acid "targets," some of which will hybridize with the DNA probes on the microarray [187]. Various methods, usually involving either fluorescence or radioactivity, can be used to detect the degree of hybridization that occurs and thus quantify the expression levels of mRNAs present. The two most popular DNA microarray technologies differ principally in the probes or arrayed material used: either short sequences of DNA (oligonucleotides) [126] or complementary DNA (cDNA) [56, 43].

1.1.3 Oligonucleotide microarrays

Oligonucleotide arrays use as probes short DNA sequences, typically of 10-60 nucleotides, which are synthesized directly on microarray slides. The three major technologies for production of oligonucleotide microarrays involve photolithography, miniature mirrors, or ink-jet printing.

One of the main producers of oligonucleotide arrays is the company Affymetrix, Inc., which uses a photolithographic process borrowed from semiconductor manufacturing [3]. Affymetrix's microarray manufacturing process begins with a quartz wafer that is coated with silane molecules, forming a uniform matrix. Nucleotides are then attached to this matrix via linker molecules. A photolithographic mask determines what area of the wafer will receive nucleotides. The mask has small windows, and when ultraviolet light is shined through these openings, the exposed linkers become chemically "de-protected," allowing coupling to added nucleotides. The added nucleotides also have chemical groups at a particular position that are removed when exposed to ultraviolet light. By using a sequence of different masks, and adding nucleotides in the appropriate order, a large number of oligonucleotides of chosen sequences may be synthesized directly on the quartz wafer. Current Affymetrix GeneChips© have over 1.3 million spatially distinct "features" (oligonucleotides of different sequences) on a single array. Each oligonucleotide probe is typically 25 bases long (a 25mer). For expression analysis applications, approximately twenty probes are carefully chosen to represent a given gene transcript. Computational and empirical methods are used to choose the probes, in order to maximize sensitivity and specificity. For each probe selected, a *mismatch* probe is constructed that differs from the probe by one nucleotide near the center of the sequence. These mismatch probes serve as controls for non-specific hybridization.

NimbleGen Systems, Inc. employs an alternative manufacturing method also based on photochemistry, but that uses miniature mirrors instead of photolithographic masks to focus light [154]. The mirrors are computer-controlled to produce high-resolution ultraviolet light patterns on a solid substrate, creating "virtual masks." Current Nimblegen arrays contain up to 2.1 million unique probe features on a single array, with probe sizes ranging from 24-70 nucleotides.

Agilent Technologies, Inc., produces oligonucleotide microarrays using an ink-jet printing method [5]. Their manufacturing process uses standard phosphoramidite chemistry to synthesize oligonucleotides base-by-base. The four different nucleotides are sprayed onto the microarray as needed in precise, minute quantities using ink-jet nozzles. Oligonucleotides of 60 bases in length (60mers) are typically used for these arrays. Current Agilent arrays contain up to 244,000 probes on a single array.

The sample preparation, hybridization, and scanning methods for all the oligonucleotide microarrays described above are similar. For example, for expression analysis using Affymetrix microarrays, the sample or target is typically prepared by collecting mRNA from a population of cells and then using a viral enzyme, reverse transcriptase, to produce a DNA copy of the RNA. The DNA is then transcribed back into RNA *in vitro* using a source of ribonucleotides with attached biotin molecules. Biotin is a small molecule that binds very tightly to the protein streptavidin, which is used later in the microarray assay. The purpose of these steps is to amplify the amount of starting mRNA and to label the material. The labelled RNA is then hybridized to the microarray, and a fluorescent molecule attached to strepavadin is added. After several more staining and washing steps, the result is that areas on the chip will fluoresce proportionately to the concentration of hybridized RNA present [4]. A laser is used to excite the fluorescing molecules, and an optical scanner captures the emissions. Image processing algorithms are then used to correct for background noise and other optical effects [3]. The final output is a set of numbers that are related to the level of mRNA present for each gene.

1.1.4 Printed cDNA microarrays

In contrast to oligonucleotide arrays, cDNA arrays typically use much longer pieces of DNA (often a few hundred nucleotides in length [56, 43]). The DNA attached to these arrays is typically obtained from cDNA *libraries*. These libraries are usually constructed by making cDNA copies (with reverse transcriptase) of mRNA from particular cell types [56] (e.g., cells from different tissue sources). Once a set of cDNA probes has been selected, the typical procedure is to amplify the cDNA using the polymerase chain reaction (PCR) and then "print" it onto a glass microscope slide. Printing is done using a variety of technologies, including a robotic arm that spots the DNA onto the slide using a pen-like apparatus [43, 187], and the Agilent ink-jet technology [5]. The cDNA array technique is very flexible, because DNA from almost any source may be used. Specialized arrays have been constructed with cDNAs from human lymphocytes and other tissues, fruit flies, yeast, bacteria, and many other sources [7, 220, 202, 63].

As with the procedure for expression analysis using oligonucleotide arrays described above, the target material to be hybridized to a cDNA array consists of a sample of mRNA prepared from a population of cells. However, in the case of cDNA arrays, labelling is done during the reverse transcription to cDNA step. The *in vitro* transcription step is skipped, and the cDNA is hybridized directly to the microarray.

These is another important difference in target preparation that is done to compensate for the somewhat inexact printing process. Because inconsistent amounts of cDNA can be deposited at a given spot, direct comparisons across different arrays may be inaccurate. Thus, a method of competitive hybridization is used in which two separate samples labelled with different fluorescent dyes are simultaneously hybridized to the array.

The remaining laser excitation and scanning processes for cDNA arrays are also similar to those used for oligonucleotide arrays. One difference is that two lasers of different wavelengths are used, so that the intensity of fluorescence of the two samples can be measured separately. A ratio is then reported, giving the fold difference in expression between the two samples. Thus, with this technique expression levels are always reported as ratios relative to one sample.

1.1.5 Comparison of DNA microarray technologies

All the microarray technologies described above have been applied to a variety of biological research problems [56, 43, 3, 154, 5]. In the past, cDNA microarrays were the dominant technology. Knowledge of a gene's DNA sequence was not required, longer probes were presumed to lead to less cross-hybridization, and they were cheaper to produce. However, with the sequencing of many biologically important organisms, the development of oligonucleotide arrays using longer probes or mismatch pairs, and cost reductions in oligonucleotide array manufacture, these advantages have become less important. Thus, recent experimental studies now most commonly employ commercially manufactured oligonucleotide arrays.

Each microarray manufacturer claims various technological advantages. Agilent argues that their technology is accurate, very flexible for producing custom arrays, and the most cost effective. Affymetrix claims to make the best quality products, because of the superiority of their manufacturing process and more extensive experience producing microarrays. NimbleGen produces the highest density microarrays, and argues that their "virtual mask" technology allows them to make very high quality custom arrays more cost effectively than can Affymetrix. At this time, it is unclear which, if any, of these technologies has significant practical advantages over the others.

1.1.6 Microarrays for measuring DNA-protein interactions

Although DNA microarrays were originally used for measuring the expression levels of genes, they have been more recently applied to localizing DNA-protein interactions genome-wide. In this *location analysis* application, a chemical such as formaldehyde is first used to cross-link proteins to DNA in cells. Extracted DNA is then broken into randomly sized pieces using sonication. An antibody to a selected protein is then used to immunoprecipitate DNA fragments bound to the protein. The resulting precipitated nucleic acids are purified, PCR amplified and fluorescently labeled to provide the material for hybridization to a microarray for detection. Figure 1-1 shows a cartoon of the experimental protocol.

This method of chromatin immunoprecipitation, followed by DNA microarray hybridization (termed ChIP-chip), has emerged as a powerful tool for studying *in vivo* genome-wide protein-DNA interactions including transcription factor binding [174, 125, 103, 199, 118, 90, 217, 122, 218, 82, 40, 177, 164], DNA replication and recombination [222, 77], and nucleosome occupancy and histone modification state [28, 153, 179, 144, 113, 27, 223]. Such information has been used to discover transcription factor DNA binding motifs, to predict gene expression, and to construct large-scale regulatory network models [82, 134, 121, 85, 21, 113, 27, 131].

A major difference between microarrays used for ChIP-chip experiments and those used for measuring gene expression levels is that the former typically need to array a much larger portion of the genome. The coding regions of genes comprise a relatively small part of the total DNA of eukaryotic organisms [203]. Because proteins may interact with DNA anywhere throughout the genome, microarrays used for ChIP-chip analysis should in principle cover the entire genome. Cost and technology limitations drove initial ChIP-chip array designs. Early studies employed printed cDNA arrays, with only a single 500-2,000 base-pair (bp) probe representing each intergenic region [118, 82, 113]. Improvements in array technology and reduced costs have allowed for smaller probe lengths and more complete genome coverage. For instance, a recent printed array design using Agilent technology covers the non-repeat portions of the yeast or human genomes with oligonucleotide probes every few hundred bases, yielding approximately 42,000 array features for yeast and 5 million for humans [164, 117]. Another recent array design uses Affymetrix microarrays with probes representing every 20 to 35 base-pairs of sequence across human chromosomes 21 and 22 [40, 27].

1.1.7 Limitations of DNA microarrays

As with any technology, DNA microarrays have their limitations. The biggest issue is that microarray data is extremely noisy. Indeed, some recent studies have found microarray data to be only moderately reproducible at best [100, 173]. Noise is introduced during the many steps of array manufacture, sample preparation, hybridization, and detection [84]. For instance, pipette error, temperature fluctuations, and reagent quality can introduce variation in mRNA amplification and the efficiency of fluorescent tagging. The variability of all these factors—from chip-to-chip, laboratory-to-laboratory, and even experimenter-to-experimenter—makes it challenging to compare results or to quantify precisely the detection limits of microarrays.

High levels of noise are especially problematic for microarrays, because they measure the expression levels of thousands of genes simultaneously. Thus, it becomes extremely likely by chance alone that at least some genes will have very high or

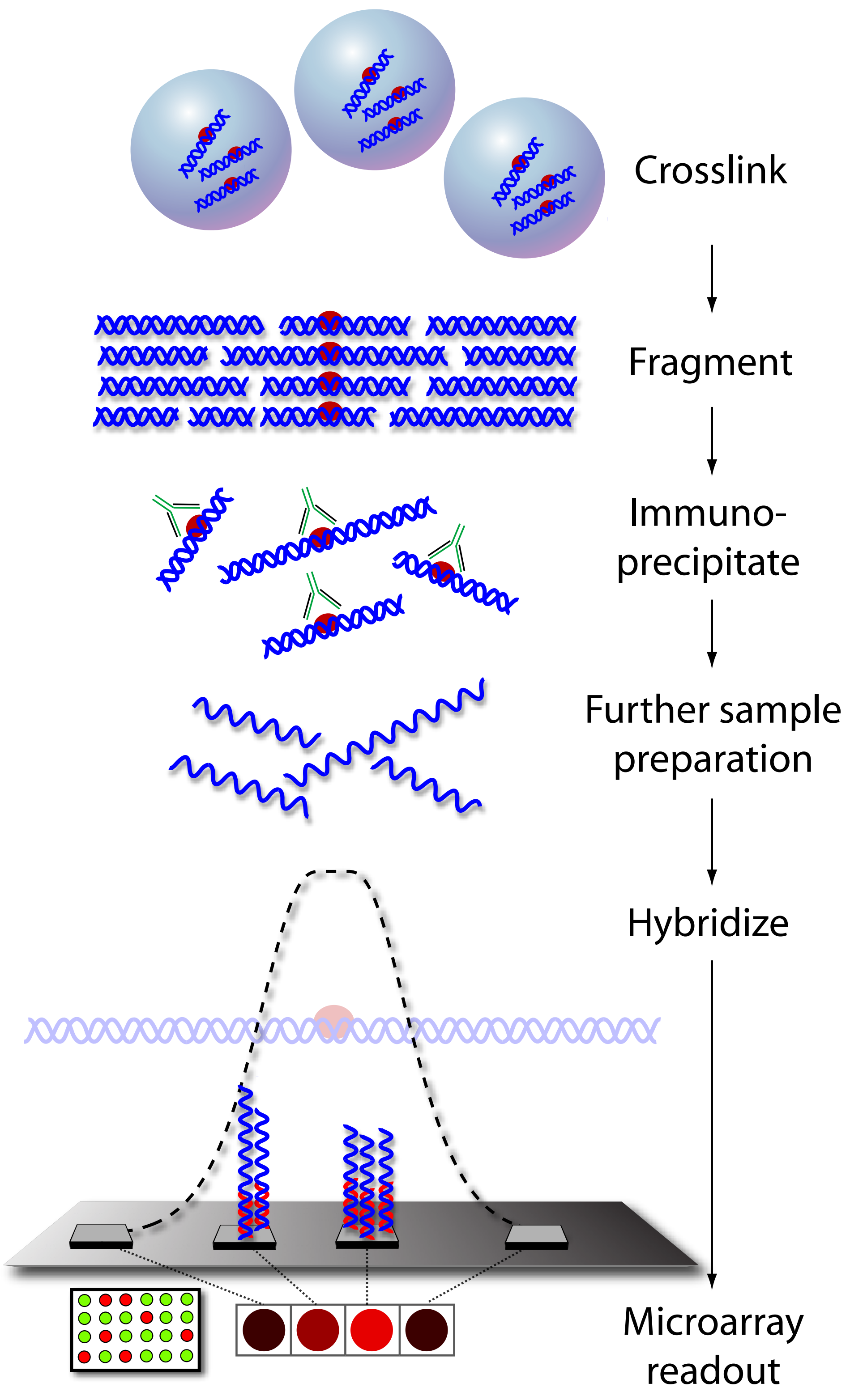

Crosslink
Fragment
Immuno-
precipitate
Further sample
preparation
Hybridize
Microarray
readout

Figure 1-1: The ChIP-chip experimental protocol is used to measure genome-wide protein-DNA binding. Formaldehyde or another chemical is used to cross-link proteins to DNA in cells. Extracted DNA is then broken into randomly sized pieces, typically using sonication. An antibody to a selected protein is then used to immunoprecipitate DNA fragments bound to the selected protein. The resulting precipitated nucleic acids are purified, PCR amplified and fluorescently labeled to provide the material for hybridization to a microarray for detection.

low expression values. In a recent commentary in *The Lancet*, Dr. John Ioannidis described several studies that suffered from this problem:

> "The promise of microarrays has been of apocalyptic dimensions. As put forth by one of their inventors, 'all human illness can be studied by microarray analysis, and the ultimate goal of this work is to develop effective treatments or cures for every human disease by 2050 [184].' All diseases are to be redefined, all human suffering reduced to gene-expression profiles ... Yet ... on close scrutiny, in five of the seven largest studies on cancer prognosis, this technology performs no better than flipping a coin ... Well, I think there is no free lunch in good research. Microarrays need evidence and this cannot be obtained from a couple of small studies, no matter how high-tech ... we should aim for many independent studies with a total of several thousand patients, a hundred-fold more than the current standard [101]."

As suggested in the above commentary, larger sample sizes are an important component for the improvement of the quality of research studies using microarrays. A complementary approach is to develop better computational and experimental methods for reducing the effective noise level of microarrays (see for example the references [97, 214, 53, 108, 54]). The work in this thesis presents several computational methods that achieve this objective, by exploiting the idea that more statistical power can be derived from groups than from single genes.

1.2 Categorization of the work in this thesis

The work described in this thesis is cross-disciplinary, involving development of new computer algorithms and their application to biological data. Multi-disciplinary work is inherently difficult to categorize.

From the computational perspective, the author developed new algorithms, extended existing ones, and wrote computer programs to implement them. That work drew on techniques from the computer science and mathematical subdisciplines of software engineering, machine learning, computational geometry, statistics, functional analysis and numerical analysis.

From the biological perspective, the author analyzed and interpreted experimental data using knowledge from the biological subdisciplines of molecular genetics, functional genomics, cell biology, anatomy, physiology, immunology and microbiology. However, this work was clearly not experimental biology, because the author did not perform experiments in a biological laboratory.

Computational biology perhaps provides the most succinct categorization of the cross-disciplinary work described in this thesis. The National Institutes of Health Biomedical Information Science and Technology Initiative Consortium's working definition of computational biology is:

> "The development and application of data-analytical and theoretical methods, mathematical modeling and computational simulation techniques to the study of biological, behavioral, and social systems [148]."

1.3 Thesis roadmap

In this thesis, we describe three novel computational approaches—the GRAM (Chapter 2), GeneProgram (Chapter 3) and GeneProgram+++ (Chapter 4) algorithms—and show that each method finds biologically meaningful sets of genes in large compendia of DNA microarray data. In this section, for each of the three approaches, we provide a brief overview of the problems addressed by the method and then outline the organization of each chapter, describing the method and its applications to biological data. Chapter 5 provides a summary of the contributions made in this thesis, and discusses directions for future work.

1.3.1 The GRAM algorithm

Problem overview

Understanding of regulatory interactions and molecular mechanisms governing genetic networks is of fundamental importance to basic biology, and is also relevant to improved diagnosis and treatment of human diseases. A variety of new high-throughput data sources have recently become available, and these hold the promise of revolutionizing molecular biology by providing a large-scale view of the regulation of genes in the cell. Fundamental goals at this scale involve discovering patterns of combinatorial regulation and how the activity of genes involved in related biological processes is coordinated and interconnected. However, each high-throughput data source measures only a particular aspect of cellular activity and suffers from limitations in accuracy. Thus, an important goal is to integrate information from multiple data sources, so that each type of data can compensate for the limitations of the others. A further goal is to develop automated methods that can aid in deducing abstractions that can conceptually reduce genetic network complexity without significant loss of explanatory power.

Initial work on constructing genome-wide regulatory networks relied exclusively on expression data (see Section 2.1 for details). However, these approaches assume that expression levels of regulated genes depend on expression levels of regulators. This assumption is often not biologically realistic, because the expression levels of many regulators do not reflect their physiologic activity due to factors such as post-translational modifications, protein degradation mechanisms, and cellular sequestration of regulators [143].

Large scale, genome-wide location analysis for DNA-binding regulators offers a second means for identifying regulatory relationships [118]. Location analysis identifies physical interactions between regulators and DNA regions, providing strong *direct* evidence for genetic regulation. Although useful, binding information is also limited, as the presence of the regulator at a promoter region indicates binding but not function; the regulator may act positively, negatively or not at all. In addition, as with all microarray based data sources, location analysis data contains substantial experimental noise. Because expression and location analysis data provide complementary information, our goal was to develop an efficient computational method for integrating these data sources. We expected that such an algorithm could provide assignments of groups of genes to regulators that would be both more accurate and more biologically relevant than assignment based solely on either data source alone.

Chapter 2 organization overview

In Chapter 2, we present a novel algorithm, GRAM (Genetic RegulAtory Modules), which fuses information from genome-wide expression and *in vivo* transcription factor-DNA binding data sets to discover regulatory networks of gene modules. A gene module is defined as a set of genes that are both co-expressed and bound by the same set of transcription factors. After some brief introductory material, we discuss prior work on gene module discovery in Section 2.1. Next, we present the algorithm in detail in Section 2.2. In Section 2.3, we use the GRAM algorithm to discover a genome-wide regulatory network using binding information for 106 transcription factors in *Saccharomyces cerevisiae* in rich media conditions and over 500 expression experiments. We also validate the quality of these results by performing analyses using four independent data sources, and use the discovered modules to label transcription factors as activators or repressors and identify patterns of combinatorial regulation. In Section 2.4, we analyze a new genome-wide location analysis data set for regulators in yeast cells treated with rapamycin, and use the GRAM algorithm to provide biological insights into this regulatory network. In Section 2.5, we present a method for using modules to build automatically genetic regulatory sub-networks for specific biological processes, and use this technique to reconstruct accurately key elements of the cell-cycle in yeast. Finally, Section 2.6 concludes the chapter with a discussion of the advantages and limitations of the GRAM algorithm.

1.3.2 The GeneProgram algorithm

Problem overview

The great anatomic and physiologic complexity of the mammalian body arises from the coordinated expression of genes. A fundamental challenge in computational biology is the identification of sets of co-activated genes in a given biological context and the characterization of the functional breadth of such sets. Understanding of the functional generality of gene sets has both practical and theoretical utility. Sets of genes that are very specific to a particular cell type or organ may be useful as diagnostic markers or drug targets. In contrast, sets of genes that are active across diverse cell types can give us insight into unexpected developmental and functional similarities among tissues. While there has been considerable effort in systems biology to understand the structure and organization of co-expressed sets of genes in isolated tissues in the context of pathological processes, such as cancer and infection [107, 128, 189, 215], relatively little attention has been given to this task in the context of normal physiology throughout the entire body [198, 205]. By analyzing gene expression in this latter context, we can gain an understanding of baseline gene expression programs and characterize the specificity of such programs in reference to organism-wide physiological processes.

Analysis of large genome-wide mammalian expression data compendia present several new challenges that do not arise when analyzing data from simpler organisms. First, tissue samples usually represent collections of diverse cell-types mixed together in different proportions. Even if a sample consists of a relatively homogenous cell population, the cells can still behave asynchronously, due to factors such as microenvironments within the tissue that receive different degrees of perfusion. Second, each tissue sample is often from a different individual, so that the compendium represents a patchwork of samples from different genetic and environmental backgrounds. Finally, the number of expression programs and distinct cell populations present in a compendium is effectively unknown *a priori*.

Chapter 3 organization overview

In Chapter 3, we present GeneProgram, a novel unsupervised computational framework that uses expression data to simultaneously organize genes into overlapping programs and tissues into groups to produce maps of inter-species expression programs, which are sorted by generality scores that exploit the automatically learned groupings. Our method addresses each of the above challenges relating to the use of large mammalian expression data compendia by using a probabilistic model that: 1) allocates mRNA to different expression programs that may be shared across tissues, 2) is hierarchical, treating each tissue as a sample from a population of related tissues, and 3) uses Dirichlet Processes, a non-parametric Bayesian method that provides prior distributions over numbers of sets while penalizing model complexity.

We begin in Section 3.1 by providing some introductory material and discussion of prior work relating to discovery of gene sets from expression data compendia. In Section 3.2, we present background material on ordinary and Hierarchical Dirichlet Process mixture models, which are a core component of the GeneProgram probability model. In Section 3.3, we provide a detailed description of the GeneProgram algorithm and probability model. In Section 3.4, we perform synthetic data experiments to explore the kinds of structures GeneProgram and several other well-known unsupervised learning algorithms can recover from noisy data. In Section 3.5, we apply GeneProgram to the Novartis Gene Atlas v2 [205], consisting of expression data for

79 human and 61 mouse tissues. Using this data set, we compare GeneProgram's ability to recover biologically relevant gene sets to that of biclustering methods, and produce a body-wide map of expression programs organized by their functional generality scores. Finally, in Section 3.6, we discuss the significance of our results and comment on possible future research directions.

1.3.3 The GeneProgram++ algorithm

Problem overview

In many microarray gene expression experiments, we are interested in genes' behavior relative to some baseline condition. For instance, we may be interested in the extent of induction or repression of gene expression after cells are exposed to environmental stresses [71], infected with microorganisms [202, 149, 150, 107, 33, 91, 160, 80], or observed throughout development [220]. The simplest such studies may consider only a few experiment-control pairs. However, in more complex studies, researchers may seek to explore complex patterns of change, such as temporal dynamics.

In analyzing patterns of gene expression change, we would like to discover sets of genes that behave coherently. In Chapter 3, we present the GeneProgram algorithm, and show that it has a number of advantages over previous methods in the discovery of biologically relevant sets of genes from large compendia of data. However, a limitation of the GeneProgram algorithm is that it does not explicitly model patterns of gene expression change.

There are at least three ways we could imagine extending the GeneProgram algorithm to model patterns of gene expression change. To simplify the discussion, we will describe examples in terms of induction or repression changes. First, we could introduce an additional parameter for each gene in each expression program, indicating whether it was induced or repressed. In this scenario, expression programs would consist of sets of genes in which each gene is consistently either induced or repressed (but not both) in a subset of tissue samples. The problem with this approach is that expression programs could be difficult to interpret and would not really match our intuition for what constitutes an "atomic" or modular biological process, in which we expect the expression of all relevant genes to coordinately change in the same direction in response to some stimuli. This expectation suggests a second possibility for extending the algorithm, in which we could introduce an additional parameter at the level of the expression program, indicating whether the genes in the program were induced or repressed. With this model, the direction of expression change for genes in a program would be consistent and would be the same for all tissue samples using the program. However, it is easy to imagine compendia of experiments in which an expression program is induced in some tissue samples and repressed in others. This then suggests the third possibility—the one GeneProgram++ uses—in which we introduce a parameter for each tissue sample at the level of the expression program, specifying the direction of expression change for genes in the program. Thus, with this model, the direction of expression change is consistent for all genes in a program for a particular sample.

Chapter 4 organization overview

In Chapter 4, we present GeneProgram++, an extension of our original GeneProgram algorithm that explicitly models general patterns of expression changes, including not only induction or repression, but also temporal dynamics. Patterns of expression change are modeled using the novel concept of program *usage modifiers*. A usage modifier is a variable that is specific to a tissue-expression program pair and describes how a tissue uses the program. For instance, usage modifiers can specify the temporal phase and direction (induction or repression) of expression. Thus, the genes used by a tissue from a program and the manner in which they are expressed (e.g., early induction versus late repression) are chosen probabilistically and influenced by the behavior of similar tissues. Further, usage is by definition consistent across a program for a particular tissue, which facilitates biological interpretation.

After some introductory material, we begin by discussing the GeneProgram++ algorithm in detail in Section 4.1. We first describe how a simplified version of GeneProgram without tissue groups is extended to include program usage modifiers. We then describe the full GeneProgram++ model and Markov Chain Monte Carlo (MCMC) sampling methods for the model. In Section 4.2, we describe some additional technical improvements relating to posterior distribution summarization, and benchmark the performance of the improved algorithm on expression data. In Section 4.3, we apply GeneProgram++ to a compendium of 62 short time-series gene expression experiments in which various human cell types have been exposed to different infectious agents or immune-modulating substances [107], and produce a map of expression programs organized by functional generality scores. We evaluate the biological relevance of the discovered expression programs using biological process categories and pathway databases, as well as genome-wide data profiling binding of human transcription factors. Finally, we provide examples of discovered expression programs involved in key pathways related to the response to infection. We conclude the chapter with Section 4.4, in which we discuss the significance of our results.

CHAPTER 2

Genetic RegulAtory Modules (GRAM)

"This we know: the earth does not belong to man, man belongs to the earth. All things are connected like the blood that unites us all. Man did not weave the web of life, he is merely a strand in it."

—*From the apocryphal Chief Seattle speech*[1]

UNDERSTANDING OF regulatory interactions and molecular mechanisms governing genetic networks is of fundamental importance to basic biology, and is also relevant to improved diagnosis and treatment of human diseases. A variety of new high-throughput data sources have recently become available, and these hold the promise of revolutionizing molecular biology by providing a large-scale view of the regulation of genes in the cell. Fundamental goals at this scale involve discovering patterns of combinatorial regulation and how the activity of genes involved in related biological processes is coordinated and interconnected. However, each high-throughput data source measures only a particular aspect of cellular activity and suffers from limitations in accuracy. Thus, an important goal is to integrate information from multiple data sources, so that each type of data can compensate for the limitations of the others. A further goal is to develop automated methods that can aid in deducing abstractions that can conceptually reduce genetic network complexity without significant loss of explanatory power.

Initial work on constructing genome-wide regulatory networks relied exclusively on expression data (see section 2.1 for details). However, these approaches assume

[1]These words are commonly thought to have been spoken by Chief Seattle, the Native American leader, in 1854. However, they were actually written by screenwriter Ted Perry for a 1972 movie, *Home*, about ecology [10]. However, because the apocryphal speech has been quoted so much, it is possible that history has finally been altered, and Chief Seattle really did say all this.

that expression levels of regulated genes depend on expression levels of regulators. This assumption is often not biologically realistic, because the expression levels of many regulators do not reflect their physiologic activity due to factors such as post-translational modifications, protein degradation mechanisms, and cellular sequestration of regulators [143].

Large scale, genome-wide location analysis for DNA-binding regulators offers a second means for identifying regulatory relationships [118]. Location analysis identifies physical interactions between regulators and DNA regions, providing strong *direct* evidence for genetic regulation. Although useful, binding information is also limited, as the presence of the regulator at a promoter region indicates binding but not function: the regulator may act positively, negatively or not at all. In addition, as with all microarray based data sources, location analysis data contains substantial experimental noise. Because expression and location analysis data provide complementary information, our goal was to develop an efficient computational method for integrating these data sources. We expected that such an algorithm could provide assignments of groups of genes to regulators that would be both more accurate and more biologically relevant than assignment based solely on either data source alone.

In this chapter, we present a novel algorithm, GRAM (Genetic RegulAtory Modules), which fuses information from genome-wide expression and *in vivo* transcription factor-DNA binding data sets to discover regulatory networks of gene modules. A gene module is defined as a set of genes that are both co-expressed and bound by the same set of transcription factors. Unlike previous approaches [58, 162, 99, 26, 191] that have relied primarily on *functional information* from expression data, the GRAM algorithm explicitly links genes to the factors that regulate them by using DNA binding data to incorporate *direct* physical evidence of regulatory interactions.

We use the GRAM algorithm to discover a genome-wide regulatory network using binding information for 106 transcription factors in *Saccharomyces cerevisiae* in rich media conditions and over 500 expression experiments. We validate the quality of these results by performing analyses using four independent data sources. We then use the discovered modules to label transcription factors as activators or repressors and identify patterns of combinatorial regulation. Further, we present a method for using modules to build automatically genetic regulatory sub-networks for specific biological processes, and use this to reconstruct accurately key elements of the cell-cycle in yeast. Finally, we analyze a new genome-wide location analysis data set for regulators in yeast cells treated with rapamycin, and use the GRAM algorithm to provide biological insights in this regulatory network.

2.1 Related work

Computational discovery of genetic regulatory networks has been a very popular research area in recent years. In this section, we will focus on previous work most relevant to our own. For additional prospectives on this topic, the reader is encouraged to consult a recent review such as [169].

Many studies have focused on networks derived from a single data type, such as gene expression or protein interaction data. For instance, Friedman *et al.* [68] and Hartemink *et al.* [84] used limited expression data to learn static Bayesian networks for the regulation of gene expression in *S. cerevisiae.* In a more recent approach, Segal *et al.* constructed a probabilistic model that uses expression data to construct a large-scale map linking regulators to regulated genes [191]. All these studies assume that expression levels of regulated genes depend on expression levels of regulators, which can be biologically unrealistic as discussed in the introduction to this chapter.

Other studies have combined genome-wide binding data with other data types, but, unlike our method, these have used a strict cutoff for binding data, reducing it to a binary relationship. For example, Hartemink *et al.* [85] used binding data to constrain the structure of a learned static Bayesian network. Ideker *et al.* [96] combined protein interaction and binding data to construct a network structure, and then used expression data to identify specific subnetworks in that network.

Several methods have been developed that combine DNA sequence motifs or information from gene function databases with expression data to discover sets of presumably co-regulated genes [99, 162]. As an example, the methods of Ihmels *et al.* and Pilpel *et al.* start with an initial set of genes that are selected using a certain criteria (e.g., DNA binding motif or MIPS functional category) and use expression data to refine the initial set [99, 162]. Although these methods represent an important first step, our method improves upon them in several ways. First, the GRAM algorithm exhaustively and efficiently searches the combinatorial space of subsets of transcriptional factors. This allows us to distinguish modules that were found to be identical in previous work but that are controlled by different transcription factors (see Section 2.6.1). Second, unlike these previous methods, the GRAM algorithm comprehensively combines the two data sources, binding and expression data, by revisiting the binding data after refining an initial gene set. Finally, our method focuses not only on the genes themselves, but also on the relationships between transcription factors and genes. This allows us to further refine our understanding of the cell's regulatory network, by assigning functional annotations to transcriptional factors.

We also note that all the prior approaches discussed above generate static or steady-state networks. In contrast, we present methods to reconstruct regulatory networks using temporal information. This is especially important in the analysis of dynamic processes in the cell, such as replication.

2.2 The GRAM algorithm

In this section we present the GRAM algorithm for discovering gene modules. As described above, modules are sets of genes that are both co-expressed and regulated by the same set of transcription factors (TFs). The computational challenges we face are:

1. *High dimensionality of expression data.* Our algorithm must handle continuous data for thousands of genes measured in hundreds of experiments.

2. *Large number of potential regulators.* Each module is potentially regulated by

any combination of over one hundred transcription factors.

3. *Noisy data.* Both genome-wide binding and expression data are measured experimentally using DNA microarrays, which produce notoriously noisy output.

The GRAM algorithm addresses these challenges by using efficient methods for robust determination of nearby genes in the high-dimensional expression data space and search over combinations of transcriptional regulators. The algorithm begins by performing an exhaustive search over all possible subsets of regulators indicated by the DNA-binding data with a stringent criterion for determining binding. Once a set of genes bound by a common set of transcriptional regulators is found, the algorithm identifies a subset of these genes with highly correlated expression, which serves as a "seed" for a gene module. The algorithm then revisits the binding data, and seeks to add additional genes to the module that are similarly expressed and bound by the same set of transcriptional regulators using a relaxed binding criteria. Note that the GRAM algorithm allows genes to belong to more than one module. In the following subsections we present a formal description of the algorithm.

2.2.1 Formal model description

Figure 2-1 provides pseudo-code for the GRAM algorithm. The inputs consist of matrices of expression data values and binding data p-values. That is, let $\boldsymbol{E}$ denote the matrix of continuous expression data, where the rows represent genes and the columns represent D experiments. We assume that expression data has been mean-centered and normalized by the standard deviation for each gene. Let $\boldsymbol{B}$ denote the matrix of binding p-values, where rows correspond to genes and columns correspond to transcription factors, i.e., b_{it} denotes the binding p-value of TF t for gene i. Below we discuss details of each step of the algorithm.

Initialization with TF binding patterns

The first step of the algorithm is the construction of a series of sets, $\mathbb{F}^1, \ldots, \mathbb{F}^J$, of all possible regulatory TF combinations strictly implied by the binding data, and all subsets of these combinations. Each set $\mathbb{F}^j$ contains all sets of TFs of size j that are implied by the data. To be precise, let $T(i,p)$ denote the set of all transcription factors that bind to a gene i with p-value less than p, i.e., the list of factors t such that $b_{it} < p$. We denote an element of $\mathbb{F}^j$ by F_k, i.e., $F_k \in \mathbb{F}^j$ implies that $|F_k| = j$ and $F_k \subseteq T(i, p_1)$ for some gene i and a strict binding p-value p_1.

The algorithm proceeds by iterating over all elements of each $\mathbb{F}^j$ beginning with the highest numbered set, i.e., with $j = J$. Note that there is a potentially exponential number of combinations of TFs to be considered as regulators for gene modules. However, because we restrict our search to combinations "confidently" implied by the data, the number of subsets to be searched over is much smaller. This feature of the algorithm enabled it to operate on a data set containing genome-wide binding data for over 100 TFs, as discussed in section 2.3.

```
GRAM($\boldsymbol{E}$,$\boldsymbol{B}$,$p_1$,$p_2$)
// $\boldsymbol{E}$ and $\boldsymbol{B}$ are matrices of expression and binding experiments, respectively
// $p_1$ and $p_2$ are strict and relaxed binding thresholds, respectively

Initialization:
        Construct a series of sets, $\mathbb{F}^1, \ldots, \mathbb{F}^J$, of all possible regulatory TF combinations
        strictly implied by the binding data, and all subsets of these combinations.
        $\mathbb{G}^J \leftarrow$ all genes
        // $\mathbb{G}^j$ is the working set of genes in iteration $j$

For all sizes of TF regulatory sets $j = J, \ldots, 1$:
        For all subsets of TFs $F_k \in \mathbb{F}^j$:
            $G \leftarrow G(F_k, p_1)$, a set of genes for a candidate module
            $\boldsymbol{c} \leftarrow$ core expression profile for the set $G(F_k, p_1)$
            Expand $G$ to $M(F_k)$, the final module, by including genes with a relaxed binding threshold
            Output $M(F_k)$, if the module has a sufficient number of genes
        $\mathbb{G}^{j-1} \leftarrow \mathbb{G}^j \setminus \bigcup_k M(F_k)$ such that $|F_k| = j$
```

Figure 2-1: Pseudo-code for the GRAM algorithm for identifying gene modules from genome-wide expression and transcription factor binding data. See the text for details.

Finding core expression profiles

The algorithm seeks to find a *core* expression profile for all genes strictly bound by a set of TFs, i.e., a point in expression space that is significantly close to as many co-bound genes as possible. Note that the core expression profile is a robust estimate in the sense that it is insensitive to co-bound genes with outlying expression measurements.

Let $\mathbb{G}^j$ denote the set of genes being considered by the algorithm for sets of TFs of size j, i.e., $|F_k| = j$. Let $G(F_k, p)$ denote the set of all genes in $\mathbb{G}^j$ to which all the TFs in F_k are bound with a given p-value threshold p, i.e., all the genes $l \in \mathbb{G}^j$ such that $F_k \subseteq T(l, p)$. Then, for every $F_k \in \mathbb{F}^j$, the genes in $G(F_k, p_1)$ serve as candidates for a module regulated by the factors in F_k.

For each candidate set $G(F_k, p_1)$ of n genes (where n is greater than some threshold), the algorithm attempts to find a core expression profile. That is, the algorithm seeks a point $\boldsymbol{c}$ in expression space such that for an expression distance threshold r_n (depending on the number of genes in the set, as described below), the ball centered at $\boldsymbol{c}$ of radius r_n contains as many genes in $G(F_k, p_1)$ as possible.

We define the distance between two points $\boldsymbol{x}$ and $\boldsymbol{y}$ in normalized expression space as:

$$d(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sqrt{\sum_{d=1}^{D} (x_d - y_d)^2}}{D}$$

We then denote by $C(F_k, p_1, \boldsymbol{c})$ the set of genes in $G(F_k, p_1)$ that are all at distance at most r_n from a given point $\boldsymbol{c}$ in expression space. In order to find the optimal point, we would like to solve the following maximization problem:

$$\boldsymbol{c} = \arg\max_{\boldsymbol{c}'} |C(F_k, p_1, \boldsymbol{c}')| \tag{2.1}$$

Unfortunately, the exact solution of this problem is exponential in the dimension D of the expression space [14]. Because the GRAM algorithm is intended for application to large expression data sets, an exact search is therefore not practical. Instead we use an approximation algorithm that has good theoretical guarantees and runs in time $O(n^3)$. Note that since only a relatively small number of genes are bound by a set of transcription factors, n will be fairly small (e.g., typically less than 100).

The approximation algorithm we use involves iterating over all triplets of genes from $G(F_k, p_1)$. For each such triplet, the algorithm computes the center $\boldsymbol{c}'$ for the triplet and the number of genes in $C(F_k, p_1, \boldsymbol{c}')$. The algorithm selects the point c that maximizes $|C(F_k, p_1, \boldsymbol{c}')|$ over all the centers computed from triplets.

In order to present the theoretical guarantees of this method for finding the core expression profile, we need a lemma from Badoiu and Clarkson [14]. Let P be a set of points in a high dimensional space. For $S \subseteq P$, let $B(S)$ denote the smallest ball enclosing all the points in S, let $\boldsymbol{c}_{B(S)}$ denote the center of such a ball, and let $r_{B(S)}$ be its radius. Then, the following lemma holds.

Lemma 1 [Badoiu and Clarkson 2003] *There exists a set $S \subseteq P$ of size $2/\epsilon$ such that the distance between $\boldsymbol{c}_{B(S)}$ and any point $\boldsymbol{p} \in P$ is at most $(1+\epsilon)r_{B(P)}$.*

To see how this applies to our method, set $\epsilon = 2/3$. Then, according to the above lemma, there exist three points such that the ball defined by the center of these points and a radius of $(1+\epsilon)r_n$ contains all the points in the set we are looking for. Because our method searches over all possible triplets of points, the algorithm must encounter the three points guaranteed by the above lemma. In other words, our algorithm finds a solution that is at least as good as a solution that can be found for $r^* = r_n/(1+2/3)$.

The radius r_n determines how close genes' expression values must be to a core profile in order to be considered co-expressed. This distance is expected to vary with the size of the set of genes being considered—the larger the set, the more likely it is that genes will be close to the core profile by chance. We determine the values for r_n by boot-strapping the data, thus avoiding additional unwarranted assumptions about specific probability models for the expression data. The method involves selecting n genes uniformly at random for each sample q. For each sample, we compute the center $\boldsymbol{c}^{(q)}$ for this set, and then determine the distance $r_n^{(q)}$ to the fifth closest gene not in the set of genes. We then select r_n to be the fifth-percentile of the sampled $r_n^{(q)}$ values.

Expanding candidate sets to determine final modules

Now that the algorithm has determined a core expression profile $\boldsymbol{c}$ for a candidate module regulated by factors F_k, it expands the module through a "relaxation" step that includes genes that may have been omitted due to noise in the TF binding measurements. This is accomplished by considering all genes i that are close to the expression core, i.e., $i \in \mathbb{G}^j$ and $d(\boldsymbol{e}_i, \boldsymbol{c}) < r_n$. For a gene i to be included in the module, it must be bound by all factors in F_k with at least a "relaxed" p-value threshold p_2 (determined experimentally as described in Section 2.2.2). That is, for all TFs $f \in F_k$, we must have that $b_{if} < p_2$. The rationale for this is that genes with expression values close to the core profile are likely to be co-regulated (bound by the same factors), and we are thus willing to relax the stringency for determining TF binding.

We then form the module $M(F_k)$, consisting of the original genes in the candidate module and those added in the relaxation step. The module thus formed will now contain similarly expressed and co-bound genes. Note that if a gene i is included in a module controlled by a set of TFs F_k, it is likely to be included in any module controlled by a subset $F_{k'} \subset F_k$. Because we are interested in the complete set of factors controlling a gene, we exclude the gene from inclusion in further modules controlled by fewer TFs. That is, $\mathbb{G}^{j-1} = \mathbb{G}^j \setminus \bigcup_k M(F_k)$ such that $|F_k| = j$. This procedure reduces the number of overlapping modules, without reducing the explanatory power of the final modules. Once all sets F_k containing j TFs factors have been evaluated, the algorithm then proceeds to consider sets with $j - 1$ TFs.

2.2.2 Experimental determination of strict and relaxed binding thresholds

In order to determine appropriate values for the strict and relaxed binding thresholds (p_1 and p_2), we performed independent experiments to test the predictions of the binding data at a number of different confidence levels. We used chromatin-IP with gene-specific PCR analysis for selected regulators to test the results predicted at each of the different p-value thresholds. We then determined how frequently each regulator-gene interaction agreed in both the genome-wide binding and gene-specific PCR experiments. We selected two different regulators, Nrg1 and Stb1, for testing. For each regulator, we selected sets of genes with genome-wide binding p-values closest to one of four thresholds (0.001, 0.005, 0.01, 0.05), and performed chromatin IP and gene-specific PCR. See the supplemental web site for the reference [118] for complete results.

Based on these experiments, we set $p_1 = 0.001$ and $p_2 = 0.01$. For these values we determined that p_1 achieves approximately a 70% true-positive rate and a 5% false-positive rate, while p_2 achieves an over 90% true-positive rate, but an over 50% false-positive rate. Thus, by starting with p_1, genes in candidate modules have a low false-positive binding rate. However, this strict p-value reduces the number of possible true-positives. As we demonstrate below, by complementing binding with expression data, we can use the relaxed threshold (p_2) and thus increase the true-positive rate without increasing the false-positive rate.

2.2.3 Inference of dynamic sub-networks

We extended the GRAM algorithm to infer dynamic sub-networks from static binding data and temporal expression data. The first step of the procedure involves running the GRAM algorithm on the full set of available transcription factor binding data and a large compendium of expression experiments. In the second step, genes in the generated modules are flagged if they are determined to be involved in a given biological process based on an objective criteria. For the cell-cycle, this criteria was cycling behavior in a collection of time-series gene expression experiments [202]. A statistical test based on the hypergeometric distribution is then used to determine which modules contain a significant number of flagged genes. The transcription factors that regulate these significant modules are then collected into a list. We then manually select a set of expression experiments in which the biological process of interest is expected to be particularly active. In the third step, GRAM is run using the selected expression data, the flagged genes, and the list of regulators determined in the previous step, producing a set of modules with genes and factors presumably directly involved in the process of interest.

Finally, in order to uncover the dynamics of the regulatory system, we combine the above procedure with our interpolation and alignment algorithms. Briefly, our interpolation and alignment algorithms use a probabilistic model to fit splines (piecewise polynomials) to gene expression time-series data and then aligns these continuous curves, which may differ in phase or period (see [73, 18, 19]). We use our interpolation algorithm to produce continuous expression profiles for all genes in the sub-network. Next, we select one module as an anchor, and align the rest of the mean continuous expression profiles of all the other modules to the mean continuous expression profile of the anchor module using our continuous alignment algorithm. We restrict the alignment to time-shift (i.e., we do not allow temporal stretching). This results in a temporal ordering of the discovered modules. Note that by using the expression levels of regulatory targets in modules (rather than TF expression levels directly), we can determine the times at which regulators are physiologically active, even if TF expression profiles do not change under the experimental conditions studied. This allows us to correctly assign regulators to different temporal phases of the dynamic system, without directly observing TF protein levels.

2.3 Gene modules in rich media conditions and modes of regulatory control

The GRAM algorithm was applied to genome-wide binding data for 106 transcription factors and over 500 expression experiments (complete details on the data used are available on the supplementary web site for the reference [21]). One-hundred six gene modules were identified, containing 655 distinct genes and regulated by 68 of the transcription factors. Figure 2-2 presents a visualization of these results as a graph with edges between gene modules and regulators. A complete list of modules is available on the supplementary web site for [21].

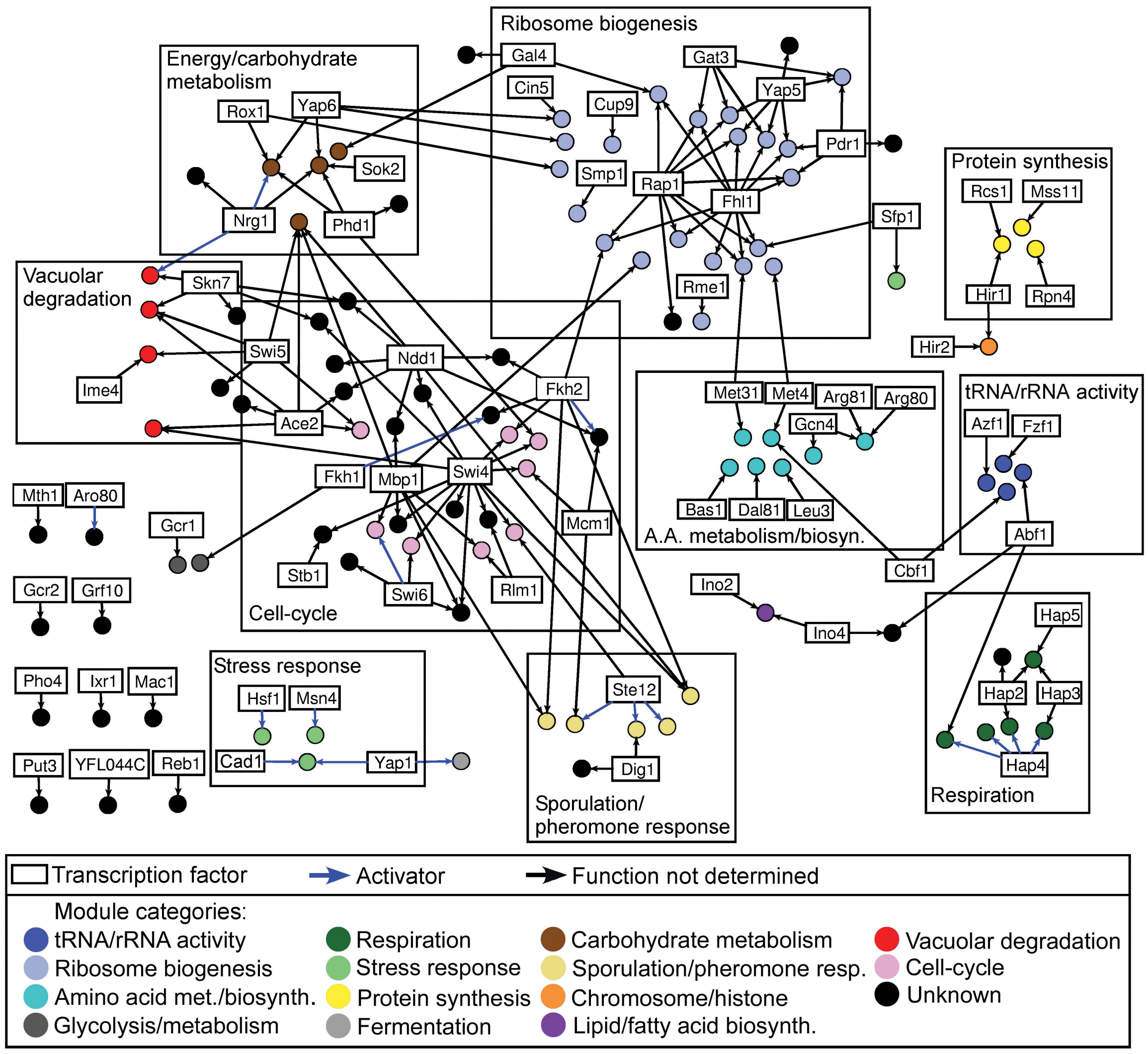


Figure 2-2: Visualization of the transcriptional regulatory network discovered by the GRAM algorithm as a graph with edges between gene modules and regulators shows that there are many groups of connected gene modules/regulators involved in similar biological processes. The network consists of 106 modules containing 655 distinct genes regulated by 68 transcription factors. In most cases in which a gene module is controlled by one or more regulators, there was previous evidence suggesting that these regulators physically or functionally interact (see Table 2.1 for details). The directed arrows point from transcription factors to the gene modules that they regulate. Blue arrows indicate discovered activator regulatory relationships (see Table 2.3 and the text for details). Gene modules are colored according to the MIPS category to which a significant number of genes belong (significance test using the hypergeometric distribution, $p < 0.005$). Modules containing many genes with unknown function or an insignificant number belonging to the same MIPS category are uncolored. When the gene modules discovered by the GRAM algorithm were compared to results generated using location data alone, the GRAM algorithm yielded an almost three-fold increase in modules significantly enriched for genes in the same MIPS category.

2.3.1 Discovery of biologically relevant gene modules

Several results obtained by analysis of the discovered gene modules suggest that the algorithm identifies biologically relevant groupings of genes. First, we found that gene modules generally identify groups of genes that function in a similar biological pathway as defined by the MIPS [140] functional categorization (see Figure 2-2 and the supplementary web site for [21]). For this analysis, we computed the overlap between the genes contained in each module and different MIPS sub-categories. We used the hypergeometric distribution to compute a significance value for this overlap, and used the category with the highest significance level to label modules, provided that the overlap for such a category was significant with a p-value < 0.005.

Second, we found the gene modules to be generally accurate in assigning regulators to sets of genes whose functions are consistent with the regulators' known functional roles. As an example, Gcr1 is a well-characterized regulator of glucose metabolism [89, 15]; 6 of the 7 genes identified in the Gcr1 module are enzymes involved in glycolysis and gluconeogenesis. Additionally, we found that in most cases in which a gene module is controlled by one or more regulators, there was previous evidence suggesting that these regulators physically or functionally interact (see Table 2.1). We used the Saccharomyces Genome Database [166] to find such evidence in the prior literature. Fifty-five modules had combinations of regulators. Of these, we found 26 combinations of regulators for which there is evidence for functional interactions (regulators bind to and regulate common genes) and 15 combinations of regulators for which there is evidence for physical interactions (regulators contact each others). For example, gene modules identify Hap2/3/4/5, Hap4/Abf1, Ino2/Ino4, Hir1/Hir2, Mbp1/Swi6, and Swi4/Swi6 interactions. Taken together, the above results provide evidence that the GRAM algorithm identifies not only biologically related sets of genes, but also relevant factors that are interacting to control the genes.

2.3.2 Integration of binding and expression data improves on either data source alone

While genome-wide location data alone is potentially useful for deriving transcriptional regulatory networks, a key feature of the GRAM algorithm is its ability to compensate for technical limitations in the location data through the integration of expression data. To determine binding events in location data, Lee *et al.* [118] used a statistical model and chose a relatively stringent p-value threshold (0.001) with the intention of reducing false-positives at the expense of true-positives. The GRAM algorithm presents a powerful alternative to using a single p-value threshold to predict binding events, because our method allows the p-value cutoff to be relaxed if there is sufficient supporting evidence from expression data. As an example, consider Hap4, a well-characterized regulator of genes involved in oxidative phosphorylation and respiration [66]. The Hap4 modules contain twenty-eight genes that are involved in respiration and show a high degree of co-regulation over the collected expression data sets (see figure 2-3). Six of these genes (PET9, ATP16, KGD2, QCR6, SDH1,

Module regulators	**Functional interactions**	**Physical interactions**
ARG80, ARG81, GCN4	Arg80, Arg81, Gcn4	Arg80, Arg81
CBF1, MET4	Cbf1, Met4	Cbf1, Met4
DIG1, STE12	Dig1, Ste12	Dig1, Ste12
FKH2, MCM1, NDD1	Fkh2, Mcm1, Ndd1	Fkh2, Mcm1, Ndd1
HAP2, HAP3, HAP5	Hap2, Hap3, Hap5	Hap2, Hap3, Hap5
HAP2, HAP4	Hap2, Hap4	Hap2, Hap4
HIR1, HIR2	Hir1, Hir2	Hir1, Hir2
INO2, INO4	Ino2, Ino4	Ino2, Ino4
MBP1, SWI4, SWI6	Mbp1, Swi6	Mbp1, Swi6
	Swi4, Swi6	Swi4, Swi6
MCM1, STE12	Mcm1, Ste12	Mcm1, Ste12
ABF1, HAP4	Abf1, Hap4	
ACE2, SWI4	Ace2, Swi4	
ACE2, MBP1, NDD1, SWI5	Ace2, Swi5	
ABF1, INO4	Abf1, Ino4	
CAD1, YAP1	Cad1, Yap1	
FKH1, FKH2	Fkh1, Fkh2	
MBP1, SWI4	Mbp1, Swi4	
SKN7, SWI4	Skn7, Swi4	
STE12, SWI4	Ste12, Swi4	

Table 2.1: Many regulator-regulator interactions predicted by the modules generated by the GRAM algorithm are confirmed by comparison to previously published literature. 55 modules have combinations of regulators. The table lists representative combinations of regulators for which there is evidence of functional or physical interactions in the Saccharomyces Genome Database [166]. Overall, we found 26 combinations of regulators for which there is evidence for functional interactions (regulators bind and regulate common genes) and 15 combinations of regulators for which there is evidence for physical interactions (regulators contact each others).

and NDI1) would not have been identified as Hap4 targets using the stringent 0.001 p-value threshold (p-values range from 0.0011 to 0.0036).

Overall, 627 out of 1560 unique regulator-gene interactions (40%) in the rich media network discovered by the GRAM algorithm would not have been detected using only location data and the stringent p-value cutoff. One would like to show that these predicted interactions are not erroneous by comparing them against some independent gold standard. Unfortunately, no such data source exists on a genome-wide scale for transcription factor-DNA interactions.

Our approach was to verify that the GRAM algorithm improves true-positive rates without significantly increasing false-positive rates by using available data from four independent sources:

1. Transcription factor-gene interactions identified previously in the literature for a well-studied biological process using non-high-throughput methods.

2. Gene-specific chromatin-IP (ChIP) experiments for selected genes.

3. The MIPS database of functional annotations [140].

4. DNA sequence motif information from the TRANSFAC database [136].

For our literature validation, we focused on known transcription factor-gene interactions involved in the cell-cycle, because this is an extensively studied system. We performed a literature search and found 51 previously identified binding relationships (see supplemental web materials [20]). Seven of these binding relationships were not detected in the genomic binding assay using a stringent cutoff [118], but three of these were identified by the GRAM algorithm. Because our method added only 59 new factor-gene relationships for genes involved in the cell-cycle, this result was significant, with a p-value $< 3 \cdot 10^{-5}$.

To further verify our results, we performed gene-specific chromatin-IP experiments for the factor Stb1 and 36 genes. The profiled genes were picked randomly from the full set of yeast genes, with representatives selected from four p-value ranges in the genome-wide binding data. In these experiments, three additional genes were determined to be bound by Stb1 that had p-values between 0.01 and 0.001 in the genomic location experiments [118], and had thus been excluded with the stringent cutoff. The GRAM algorithm identified all three genes as bound by Stb1 without adding any additional genes that were not detected in the gene-specific chromatin-IP experiments (see Table 2.2 and [20]).

We also expected that the gene modules derived by the GRAM algorithm would improve on the biological relevance of gene groupings that could be inferred from location data only. Since genes that participate in the same biological pathway often have similar expression patterns, and genes in a module share not only a common set of transcription factors but also similar expression patterns, we expected that genes in modules would more likely be functionally related than sets of genes identified by location data alone. Indeed, we found that gene modules derived using the GRAM algorithm were almost three times more likely to show enrichment for genes in a MIPS functional category than were sets of genes derived solely from location data.

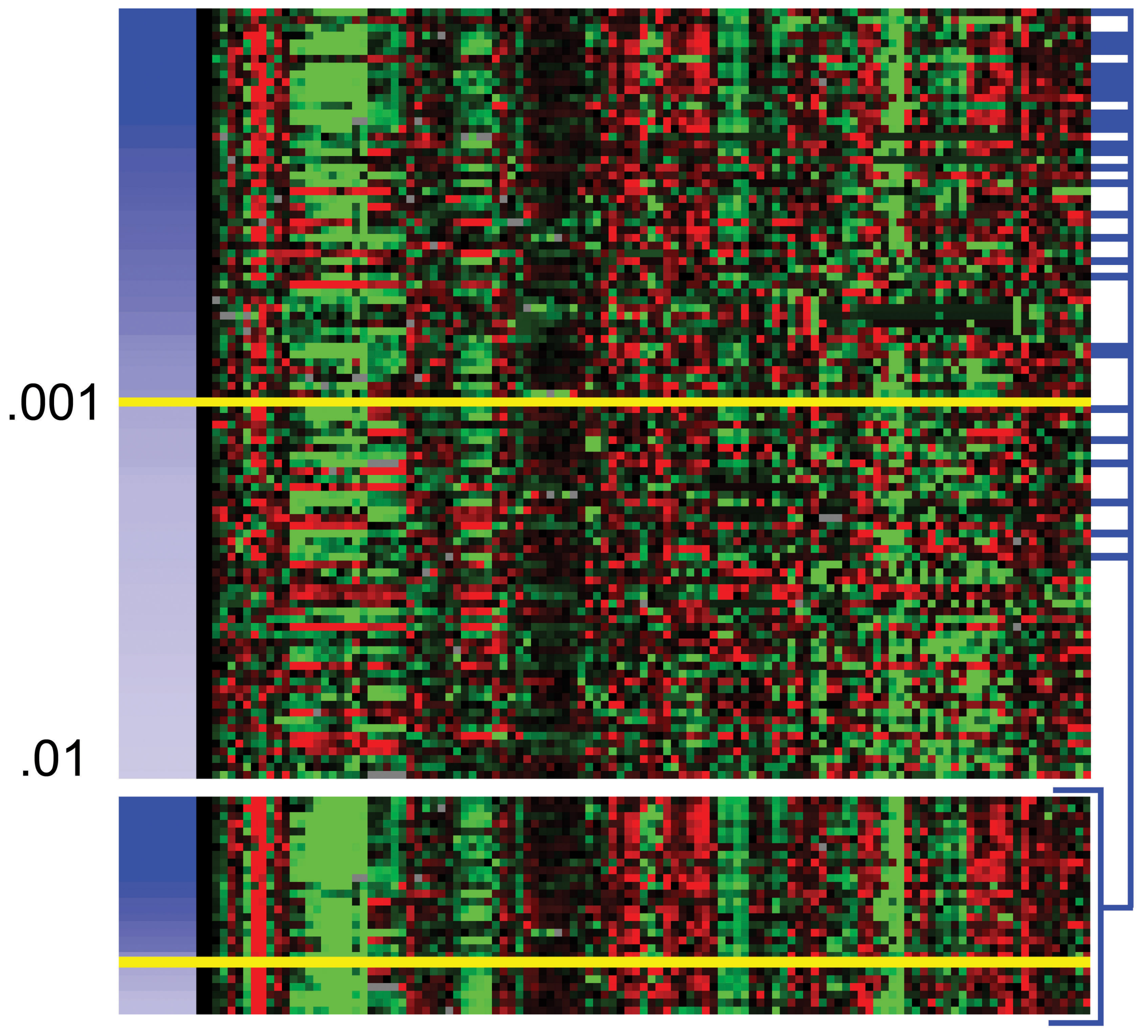


Figure 2-3: The GRAM algorithm improves the quality of DNA-binding information, because it uses expression data to avoid a strict statistical significance threshold. Shown is DNA-binding and expression information for the 99 genes bound by the regulator Hap4 with a p-value < 0.01 using the statistical model in Lee *et al.* [118]. The blue-white column on the left indicates binding p-values, and the horizontal yellow line denotes the strict significance threshold of 0.001. As can be seen, the p-values form a continuum and a strict threshold is unlikely to produce good results. The blue horizontal lines on the right indicate the 28 genes that were selected for modules by the GRAM algorithm. As can be seen, 22 (79%) have a p-value < 0.001, but 6 (21%) have p-values above this threshold. The lower portion of the figure shows together the 28 genes selected by the GRAM algorithm, and it can be seen that they exhibit coherent expression. Further, all the selected genes are involved in respiration. Six of these genes (PET9, ATP16, KGD2, QCR6, SDH1, and NDI1) would not have been identified as Hap4 targets using the stringent 0.001 p-value threshold (p-values range from 0.0011 to 0.0036).

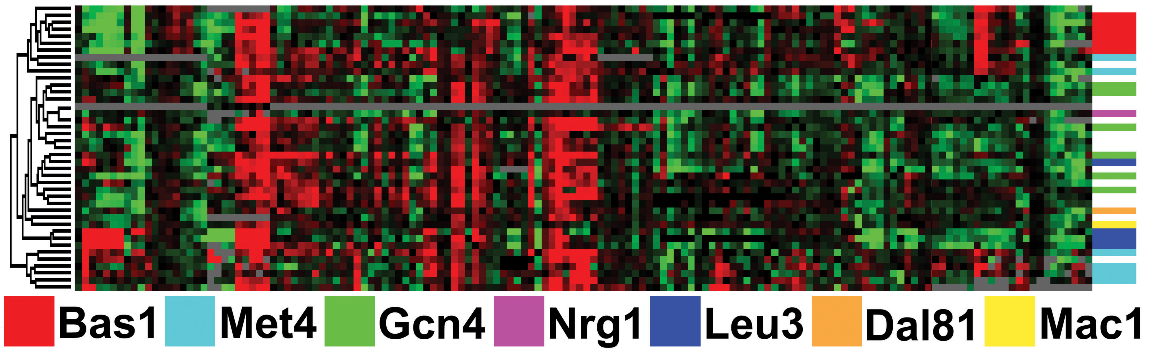


Figure 2-4: The GRAM algorithm assigns different regulators to genes with similar expression patterns that cannot be distinguished using expression clustering methods alone. Hierarchical clustering of expression data was used to obtain the sub-tree on the left. On the right, the regulators assigned to genes by the GRAM algorithm are color coded. As can be seen, many genes with very similar expression patterns are regulated by different transcription factors.

The Stb1-Swi4 Module

Orf name	**Gene name**	**Cell-cycle phase**	**Binding p-value for Stb1**
YOR065W	**Hcm1**	**G1**	**0.0012**
YDR501W	YDR501W	G1	0.00002
YGR109C	Clb6	G1	0.0013
YGR221C	YGR221C	G1	0.0009
YIL140W	**Sro4**	**G1**	**0.008**
YIL141W	YIL141W	G1	0.008
YMR179W	**Spt21**	**G1**	**0.007**
YNL289W	Pcl1	G1	0.0000005
YPL256W	Cln2	G1	0.00007

Table 2.2: The GRAM algorithm can improve the true-positive binding rate without increasing the false-positive binding rate. In the module controlled by Swi4 and Stb1, out of the nine genes contained in the module, five had a *p*-value higher than 0.001 for Stb1 binding, and were thus not considered as bound by Stb1 in Lee *et al.* [118]. However, independent gene-specific chromatin-IP experiments for Stb1 confirmed the prediction of the GRAM algorithm for three out of the five genes (HCM1, SRO4 and SPT21).

We similarly expected that genes in modules derived by the GRAM algorithm would be more likely to show independent evidence of co-regulation by the regulators assigned to the module when compared to sets of genes obtained using location data alone. One line of evidence for such an improvement would be enrichment for specific DNA sequence motifs. We identified 34 transcriptional regulators that bind to genes in at least one module and have well-characterized DNA binding motifs in the TRANSFAC database [136]. For each of these 34 transcriptional regulators, we generated a list of genes in modules bound by the regulator and a second list of genes bound by the regulator using location data alone (stringent p-value cutoff of 0.001). We then computed the percentage of genes from each list that contained the appropriate known motif in the upstream region of DNA. We found that in most cases, the percentage of genes containing the correct motif was higher when modules generated using the GRAM algorithm were used as compared to sets of genes generated from location data alone (see Figure 2-5 and the supplementary materials for [21]).

2.3.3 Identification of activators and repressors

The gene modules abstraction allowed us to label regulator-module edges in the graph to indicate whether there is significant evidence that regulators may be functioning as activators or repressors. Because a gene module provides a link between a set of regulators and the common expression pattern of a set of bound genes, we can use the relationship between a regulator's expression pattern and the common expression pattern of genes in a module to infer whether a regulator acts as an activator or repressor. Note that the use of genomic location data alone only allows us to infer the presence of regulators at promoters, but can give no information about the type of interaction.

We searched for activator/repressor relationships by examining regulators with expression profiles that are positively/negatively correlated with the expression profiles of genes in the corresponding modules. Positive correlation indicates that higher levels of regulator expression correlate with higher levels of expression of genes in the module and suggests that the transcription factor positively regulates the expression of genes in the module; negative correlation suggests an opposite, repressive relationship. We determined the statistical significance of the activator/repressor relationships by computing correlation coefficients between all transcriptional regulators studied and all gene modules and taking the 5% positive tail of the distribution of correlation coefficients. Tables 2.3 and 2.4 present the eleven activators and five repressors determined using the method described above.

Ten of the eleven activators were previously identified in the literature, suggesting that this analysis produces biologically meaningful results. The literature offered less information about the five repressors; only Nrg1 was previously identified in the literature as a repressor. In at least one case, a factor may serve in both activator and repressor roles under different conditions. Ino4 and Ino2 are thought to dimerize and activate genes in low inositol conditions, but while Ino4 is required for binding it apparently does not affect activation. In our analysis, the highest degree of negative correlation occurs in stress condition expression experiments, suggesting that Ino4

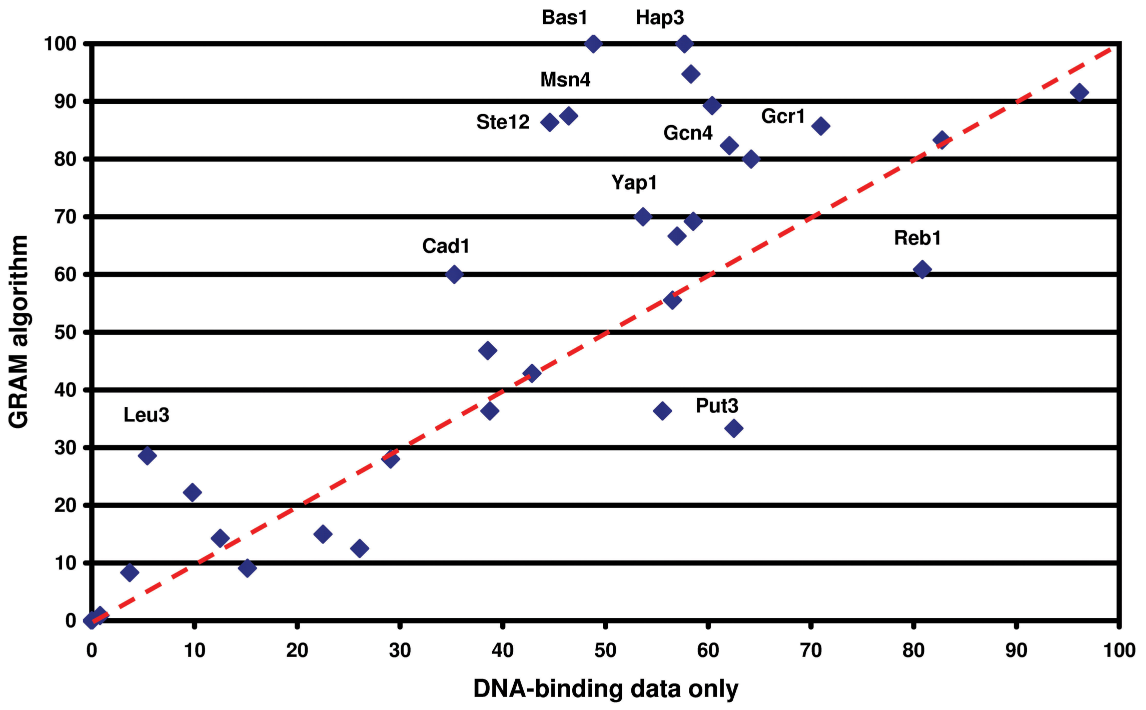


Figure 2-5: Motif enrichment: genes in modules discovered by the GRAM algorithm are more likely to show independent evidence of co-regulation by the regulators assigned to the module when compared to sets of genes obtained using genomic location analysis data alone, as demonstrated by an enrichment for the presence of known DNA-binding motifs. We identified 34 transcriptional regulators that bind to genes in at least one module and have well-characterized DNA binding motifs in the TRANSFAC database [136]. For each of these 34 transcriptional regulators, we generated a list of genes in modules bound by the regulator and a second list of genes bound by the regulator using location analysis data alone (stringent p-value cutoff of 0.001). We then computed the percentage of genes from each list that contained the appropriate known motif in the upstream region of DNA. In most cases, the percentage of genes containing the correct motif was higher when modules generated using the GRAM algorithm were used versus sets of genes generated from location analysis data alone. See the supplementary materials for [21] for a complete list of transcription factors.

might serve as a repressor under certain conditions and as an activator under others.

Activators identified by our algorithm

Factor	**Module function**	**Corr. w/ module**	**Comments**
Ste12	Pheromone response	+0.64	Activator, required for pheromone response
Hap4	Respiration	+0.60	Activator of CCAAT box containing genes
Yap1	Detoxification	+0.53	Activator, possibly involved in oxidative stress response
Nrg1	Carbohydrate transport	+0.50	Previously identified as a repressor
Fkh1	Cell-cycle	+0.49	Activator of cell-cycle genes
Cad1	Detoxification	+0.47	Activator, involved in multidrug resistance
Aro80	Energy and metabolism	+0.40	Activator, involved in regulation of amino acid synthesis
Swi6	Cell-cycle	+0.39	Activator of cell-cycle genes
Msn4	Stress response	+0.38	Activator, involved in stress response
Fkh2	Cell-cycle	+0.37	Activator of cell-cycle genes
Hsf1	Stress response	+0.36	Activator of heat shock related genes

Table 2.3: Eleven activators were identified by our algorithm by computing the correlation between the expression patterns of genes in a module and its regulators. Ten of the eleven activators were previously identified in the literature.

2.3.4 Discovery of modes of combinatorial regulation

A central feature of eukaryotic transcriptional regulation is combinatorial control, the ability of different combinations of transcription factors to specify distinct biological outcomes [200]. We sought to determine how combinations of regulators affect expression by examining the correlation between expression profiles of genes from pairs of modules that share at least one transcription factor but have no genes in common. This analysis suggests four distinct types of coordinate regulation:

- *Additive*—the binding of additional factors increases the expression levels of the bound genes. Our analysis suggested that this is the most common type of coordinate regulation. The complete set of pairs of modules that exhibit additive control appears on the supplementary website [20].

Repressors identified by our algorithm

Factor	**Module function**	**Corr. w/ module**	**Comments**
YFL044C	Unknown	-0.44	Function unknown
Azf1	rRNA transcription	-0.41	Previously identified as an activator
Nrg1	Unknown	-0.40	Transcriptional repressor for glucose gene expression
Yap5	Ribosome biogenesis	-0.39	Function unknown
Ino4	Fatty acid biosynthesis	-0.39	Previous evidence of involvement in Ino2-Ino4 dimer that activates in low inositol conditions

Table 2.4: Five repressors were identified by our algorithm. One of them was previously reported in the literature, and two have not been studied before. In at least one case, a factor may serve in both activator and repressor roles under different conditions. Ino4 and Ino2 are thought to dimerize and activate genes in low inositol conditions, but while Ino4 is required for binding it apparently does not affect activation. In our analysis, the highest degree of negative correlation occurs in stress condition expression experiments, suggesting that Ino4 might serve as a repressor under certain conditions and as an activator under others.

- *Negative*—a factor serves as an activator for one module, but addition of a partner factor for a second module abolishes activation or causes repression. This results in negative expression correlation between the two modules under all conditions. For example, the module controlled by the two cell-cycle activators Swi4 and Mbp1 was strongly negatively correlated with a module controlled by Swi4 and Skn7. This is plausible, because there is evidence that Skn7 acts as a repressor in the oxidative stress response in yeast [116].

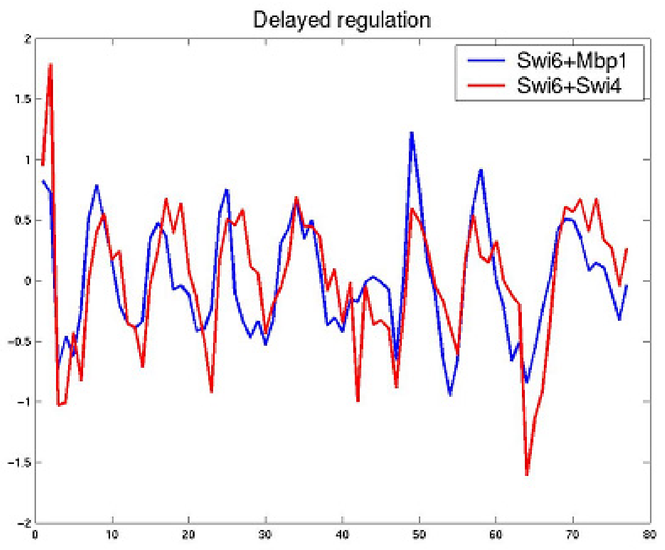
- *Delayed*—one factor regulates two or more modules in a similar way, but addition of a partner factor causes expression of the genes in the two modules to be offset temporally. Thus, the average expression of the modules can be aligned after an appropriate time shift. For example, Swi6, a cell-cycle transcription factor, is known to partner with both Swi4 and Mbp1 to regulate genes in the G1/S cell-cycle phase. However, since Swi4 itself is regulated by Swi6, expression of the set of genes regulated by Swi6/Mbp1 occurs earlier than that of those regulated by Swi6/Swi4 (see Figure 2-6). Many other cell-cycle factors exhibit this type of delayed regulation.

- *Conditional*—addition of a partner factor causes expression changes in a subset of conditions. For example, our algorithm discovered a module regulated by Met4 alone and a second regulated by Met4 and Cbf1. As shown in Figure 2-6, under many conditions the average expression profile of genes in the module regulated by Met4 is very similar to that of genes in the module regulated by Met4/Cbf1. However, there are some experiments in which the average expression profiles of genes in the two modules are anti-correlated, most notably under stress conditions.

2.4 The rapamycin response regulatory network

For the rich media network described above, we used a very large set of genome-wide binding and expression data, which allowed us to validate the results of the GRAM algorithm comprehensively with searches of the prior literature, independent chromatin-IP experiments, and analysis for enrichment for genes in the same MIPS category and for known DNA binding motifs. The results of this large-scale validation gave us confidence that the GRAM algorithm would be useful in analyzing new data sources. Because biological insights are often gained by examining responses to specialized treatments or environmental conditions, we were interested in exploring the performance of the GRAM algorithm on a smaller, more biologically targeted data set than the rich media data. So, we chose to examine a transcriptional regulatory sub-network involved in the response to Tor kinase signaling.

The Tor proteins are highly conserved and function as critical regulators in the response to nutrient stress [104, 49, 172]. Tor kinase signaling can be inhibited by the addition of the small macrolide rapamycin, which mimics nutrient starvation and results in a wide range of physiological responses including cytoskeleton reorganization,



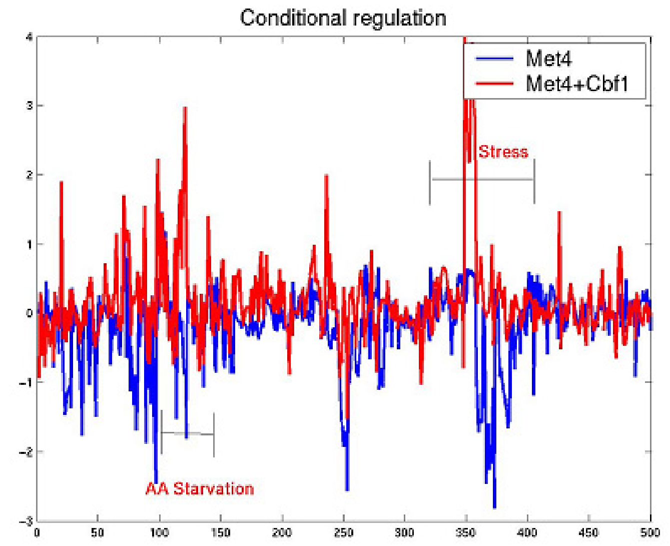


Figure 2-6: Combinatorial regulation modes: Our analysis of pairs of modules that share at least one transcription factor but have no genes in common revealed several distinct types of coordinate regulation. Examples of two such types are shown in the above figures. *(Top figure)* Delayed regulation: in delayed control, a factor regulates two or more modules in a similar way, but the expression of these sets of genes are temporally separated, an effect brought about by the activity of different bound partner factors. As can be seen, the average expression of genes in the module regulated by Swi6/Mbp1 lags that of genes in the module regulated by Swi6/Swi4, though both belong to the G1 phase. *(Bottom figure)* Conditional regulation: in conditional control, a partner factor affects expression primarily in a subset of conditions. As can be seen, under many conditions the average expression profile of genes in the module regulated by Met4 is very similar to that of genes in the module regulated by Met4/Cbf1. However, there are some experiments in which the average expression profiles of genes in the two modules are anti-correlated, most notably under stress conditions.

decreased translation initiation, decreased ribosome biogenesis, amino acid permease regulation, and autophagy [195, 38, 86, 175]. Expression analysis indicates that Tor signaling also controls transcriptional regulation of metabolic pathways involving nitrogen metabolism, glycolysis and the TCA cycle [83, 195, 38].

The rapamycin response presented an ideal opportunity for applying the GRAM algorithm to analyzing a novel transcriptional regulatory network. Previous studies suggested a specific set of regulators that are likely to function in the transcriptional response to rapamycin [83, 195]. Also, several publicly available genome-wide expression datasets measuring response after rapamycin treatment are available [83, 195]. More importantly, the fact that there is little information about the transcriptional regulatory network involved and how this transcriptional network may contribute to the overall response to rapamycin treatment presented an opportunity for new biological insights.

2.4.1 Selection of relevant factors

We selected 14 transcriptional regulators that seemed likely to function in the rapamycin response in *S. cerevisiae* based on evidence from the literature. These factors and our rationale for choosing them are:

- Gln3 and Gat1—these factors have been identified as general activators of expression of nitrogen responsive genes. The activated Tor proteins lead to sequesterization of Gln3/Gat1 in the cytoplasm, and subsequent rapamycin treatment and Tor inactivation allows Gln3/Gat to enter the nucleus [48, 195]. Gln3/Gat1 apparently activate the allantoin pathway [188].

- Gzf3—this factor and Dal80 have been identified as general repressors of the nitrogen responsive genes, and there is evidence that these repressors operate by competing with Gln3/Gat1 DNA-binding [46].

- Dal80—known to repress the allantoin pathway [188]. See also Gln3/Gat1 and Gzf3.

- Dal81 and Dal82—these factors have been identified as positive regulators of the allantoin pathway, which degrades allantoin and its metabolic products to ammonia and carbon dioxide [188].

- Msn2 and Msn4—rapamycin apparently allows the transcription factors Msn2/4 to enter the nucleus. Msn2/4 generally respond to cellular stress, including carbon source limitation [48, 195].

- Rtg1 and Rtg3—the activated Tor proteins apparently maintain the transcriptional regulators Rtg1/Rtg3 in the cytoplasm and rapamycin treatment allows nuclear release. Rtg1/Rtg3 generally regulate TCA cycle and glyoxylate cycle genes [48, 195].

- Hap2—part of the Hap2/3/4/5 complex, which has been shown to serve as a transcriptional activator for TCA cycle genes. There is evidence that rapamycin leads to the activation of both Hap2 and Rtg1/Rtg3 [195].
- Fhl1—has been identified as a transcriptional regulator of rRNA processing genes [87], although its relationship to rapamycin/Tor pathways has not been well-established.
- Gcn4—a transcriptional activator of amino acid biosynthetic genes, and has been shown to be a critical "master regulator" in the amino acid starvation response [146].
- Uga3—a transcriptional activator of genes in the GABA degradative pathway [98].

2.4.2 Genome-wide location analysis of selected factors

We performed genome-wide location analysis experiments on the 14 selected transcription factors. Epitope-tagged strains were generated as described previously [118]. Briefly, regulators were tagged at the C-terminus by using homologous recombination to insert multiple copies of the Myc epitope coding sequence into the normal chromosomal loci of these genes. Insertion of the epitope coding sequence was confirmed by PCR and expression of the epitope-tagged protein was confirmed by Western blotting analysis.

Strains containing epitope-tagged regulators were grown in 50 ml YPD (yeast extract-peptone-dextrose) at 30 degrees C. Cells were grown to an OD600 of 0.7-0.8 and rapamycin was then added to a final concentration of 100 nM. Cells were grown for 20 minutes at 30 degrees C in the presence of rapamycin.

Genome-wide location analysis was performed as previously described [118]. Briefly, cells containing an epitope-tagged regulator were fixed with formaldehyde (1% final concentration) and then harvested by centrifugation. Cells were lysed and then sonicated to shear DNA. DNA fragments representing chromosomal regions crosslinked to a protein of interest were enriched by immunoprecipitation with an anti-epitope antibody. After reversal of crosslinking, enriched DNA was purified. The ends of DNA fragments were then blunted using T4 DNA polymerase and ligated to previously prepared linkers. The enriched DNA was then amplified and labeled with a fluorescent dye by ligation-mediated PCR (LM-PCR). A sample of control DNA was similarly processed and labeled with a different fluorophore. Both IP-enriched and control DNA were then hybridized to a single DNA microarray. For each factor, three independently grown cell cultures were processed and scanned to generate binding information as previously described.

2.4.3 Genome-wide rapamycin regulatory network

We ran the GRAM algorithm using the binding data for the 14 transcription factors in rapamycin and 22 previously published expression experiments relevant to rapamycin conditions. Thirty-nine gene modules containing 317 unique genes and regulated by 13 transcription factors were discovered (see Figure 2-7 and the supplementary material for [21]). It should be noted that many of the transcription factors profiled have been demonstrated to be regulated by cytoplasmic sequesterization [195, 49], so we did not expect to be able to identify activator/repressor relationships by searching for transcription factor expression profile correlations with regulated modules. The GRAM algorithm added 192 gene-regulator interactions that would not have been identified with a strict p-value (0.001) in the location analysis experiments.

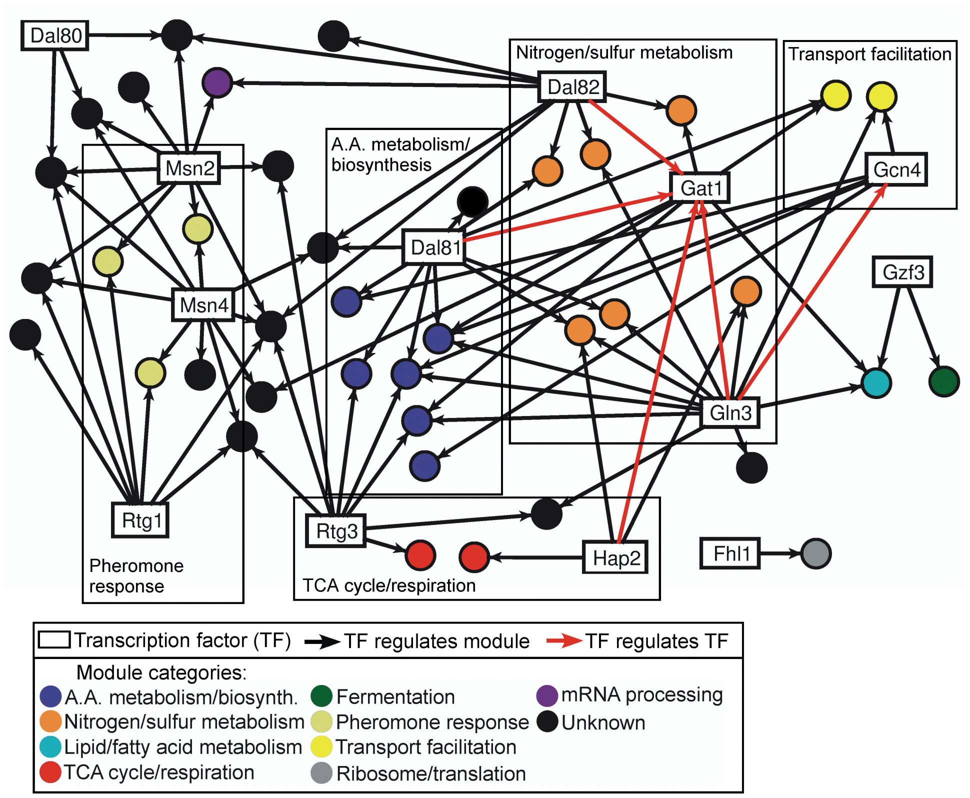


Figure 2-7: Rapamycin gene modules network: analysis of the rapamycin transcriptional regulatory network revealed a number of novel biological insights, including evidence that some transcriptional regulators may control genes involved in biological pathways different from those generally associated with these regulators. Further, analysis of the network suggested more complex regulatory interactions in which there is communication among modules. Such complicated network topologies may be important for facilitating rapid and flexible responses to changing environmental conditions. See the text for further details. Thirty-nine modules containing 317 unique genes and regulated by 13 transcription factors were discovered. Red arrows between transcriptional regulators indicate that the source transcription factor binds at least one module containing the target transcription factor. Modules are colored according to the MIPS category to which a significant number of genes belong (significance test using the hypergeometric distribution, p-value < 0.05).

As in the case for the rich media gene modules network, many features of the rapamycin regulatory network discovered by the GRAM algorithm were consistent with expectations from the literature. Twenty-three of the gene modules were found to contain a significant number of genes (p-value < 0.05) belonging to a single MIPS category. There were a total of 9 represented categories, all corresponding to biological responses associated with rapamycin treatment [104, 49, 172]. We also found that in general, regulators were assigned to genes that reflect functions described in previously published results.

In addition to identifying established regulatory interactions, analysis of the rapamycin gene modules suggested several unexpected interactions in which regulators typically assigned to a particular biological response also appear to bind genes acting in different biological pathways. Below we give several examples of such regulatory interactions. These findings suggest models of transcriptional regulation of the rapamycin response that can be validated in further more directed studies. A first example of an unexpected regulatory interaction involves the factors Msn2 and Msn4, which are generally regarded as stress response factors and have been well-studied as activators of such stress-related responses [86, 175, 35]. Unexpectedly, there were five gene modules in which Msn2 and Msn4 were bound to a significant number of genes involved in the mating pheromone response pathway. A second example involves the factors Rtg1 and Rtg3, which are generally thought to regulate directly genes of the TCA cycle and indirectly contribute to nitrogen metabolism [186, 49, 112, 124] (products of the TCA cycle are shunted to nitrogen metabolism pathways in low or poor nitrogen conditions). Our gene modules network suggests that Rtg regulators may directly regulate genes involved in nitrogen metabolism.

A third example of an unexpected regulatory interaction involves Hap2, a part of the Hap2/3/4/5 complex which has been well-characterized as a regulator of genes involved in respiration [186, 163]. Indeed, in the rich media gene modules network, members of the Hap complex were unique among the 106 regulators profiled as the only regulators controlling modules that are significantly enriched for genes involved in respiration. As expected, Hap2 regulates a module of respiration genes under rapamycin conditions. Unexpectedly, Hap2 was also found to regulate two modules containing genes involved in nitrogen metabolism. There is some genetic evidence for such cross-pathway regulation, as Hap2 was previously implicated as a regulator of two nitrogen metabolism genes [50, 51]. Our results indicate that Hap2 participates in cross-pathway regulation more extensively than previously reported.

In addition to suggesting that some transcriptional regulators may control genes involved in biological pathways different from those generally associated with these regulators, analysis of the gene modules network suggested more complex regulatory interactions in which there is communication among gene modules. Such complicated network topologies may be important for facilitating rapid and flexible responses to changing environmental conditions. As an example, we found that several transcriptional regulators may be involved in a feed-forward regulatory loop in which the gene encoding a regulator is bound by another regulator and both regulators bind to a set of common genes [118, 196]. The regulator Gat1 has been previously identified as a general activator of nitrogen responsive genes [44]. We found that Gat1 was itself contained in several modules along with genes involved in nitrogen metabolism. These gene modules were bound by the transcriptional regulators Dal81, Dal82, Gln3 and Hap2. Interestingly, Gat1 also bound several gene modules along with Dal81, Dal82, and Gln3 (see Figure 2-7). Feed-forward mechanisms may be important in regulatory responses (such as the response to rapamycin) by modulating regulatory sensitivity to sustained rather than transient inputs, providing temporal control, or amplifying the transcriptional response [196].

Analysis of the network also revealed several instances of non-transcriptional regulatory interactions between modules. For example, Msn2 bound to a module containing Crm1, which is a nuclear export factor critical in excluding Gln3 from the nucleus [195, 49]. This finding suggests that Msn2 activation upon rapamycin treatment may modulate Gln3 activation. As another example, Gcn4 bound to a module containing genes involved in amino acid biosynthesis, including Npr1, a serine/threonine protein kinase that is known to promote the function of the general permease Gap1 [195]. Gap1 itself was contained in a module regulated by Dal81 and Gln3. These findings suggest regulatory connections between these modules, in which Gap1 is transcriptionally regulated by Dal81/Gln3, Npr1 is transcriptionally regulated by Gcn4, and then Gap1 is non-transcriptionally activated by Npr1. It is possible that such regulatory relationships, which are clearly more complicated than simple activator/repressor mechanisms, may be especially important for facilitating a rapid and flexible response to environmental emergencies such as rapamycin exposure.

2.5 The cell-cycle dynamic regulatory network

When additional information is available, such as temporal expression data, even richer regulatory networks can be inferred than the static models described above. We applied our sub-network discovery algorithm to the yeast cell-cycle in combination with our continuous temporal alignment algorithm to uncover not only modules and their regulating transcription factors, but also the temporal relationships among these modules. The cell-cycle regulatory network was selected because of the importance of this biological process, the availability of extensive genome-wide expression data for the cell-cycle, the extensive literature that can be used to understand features of the regulatory network model, and our interest in determining whether a principled computational approach can reproduce substantial portions of the network that was previously discovered using a manual approach [199].

We applied the sub-network discovery algorithm as described in Section 2.2.3 to construct the cell-cycle dynamic regulatory model. The expression data used consisted of a time-series covering approximately two cell-cycles and obtained from alpha mating factor synchronized cells [202]. Eleven modules containing 75 genes and regulated by 12 factors were found (see Figure 2-8). A module regulated by the factors Swi5/Ace2 and containing genes known to be active at the G1/M boundary was chosen as the start of the cell-cycle and the other modules were aligned against this, allowing us to localize all the modules temporally. We were then able to place approximately the boundaries for S, G2, and M and thus estimate the lengths of these phases by using prior biological knowledge about when genes in four other modules peak during the cell-cycle.

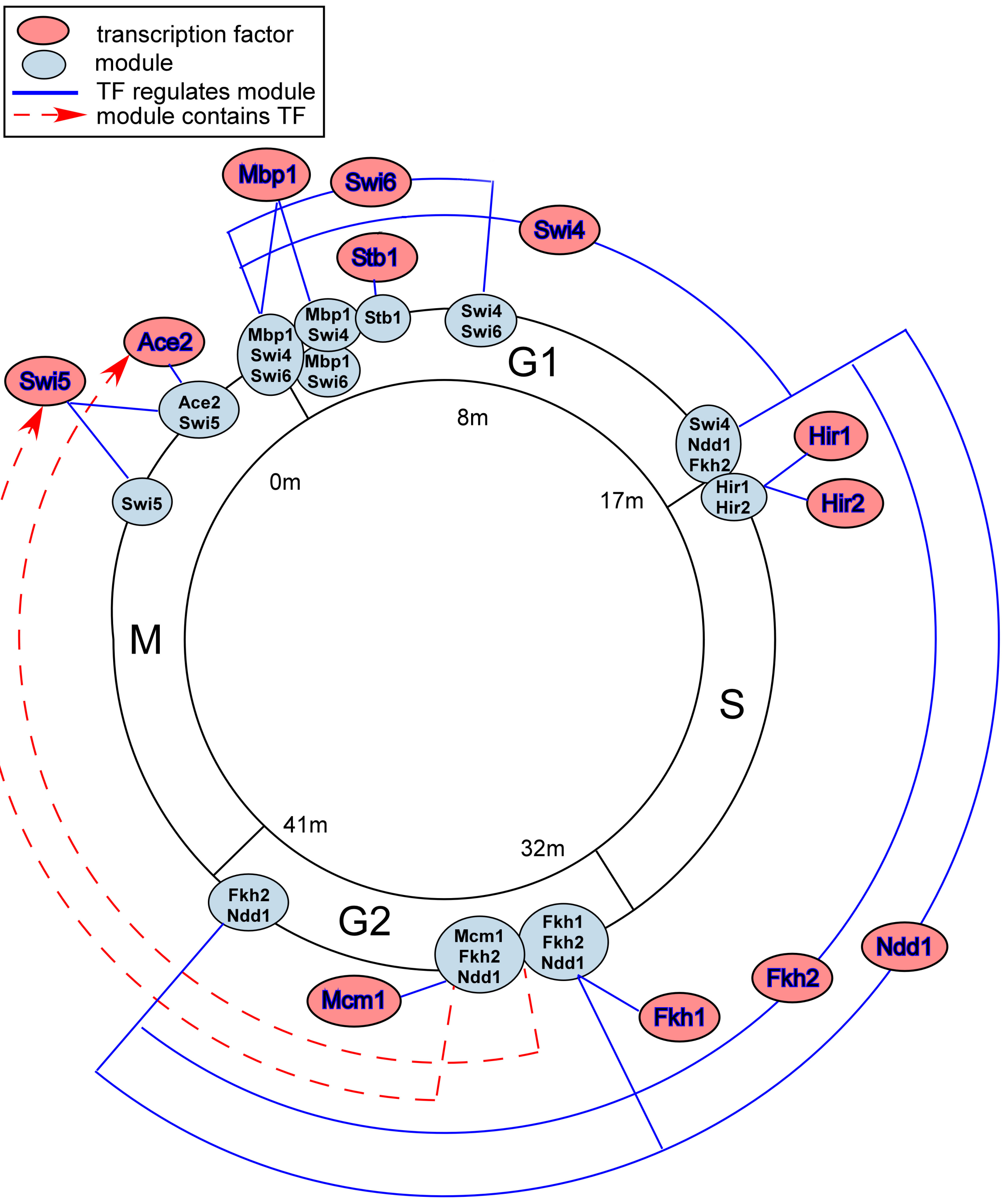


Figure 2-8: Cell-cycle dynamic regulatory network: in order to investigate the yeast cell-cycle, we applied our sub-network discovery algorithm in combination with a continuous temporal alignment algorithm to uncover not only modules and their regulating transcription factors, but also the temporal relationships among these modules. The automatically recovered network is similar to the one described in Simon *et al.* [199], which required considerable prior biological knowledge to construct. Modules are shown as ovals containing the names of regulating factors. Blue lines indicate that a transcription factor regulates a module (the arcs indicate the temporal extent of the factor's regulatory activity). Red dashed lines indicate that a transcription factor is itself contained in a module.

.

Our algorithm correctly assigned all the regulators to stages of the cell-cycle in which they have been described to function in previous studies (see Simon *et al.* [199] and references within). Significantly, this accurate reconstruction of the regulatory architecture was automatic and required no prior knowledge of the regulators that control transcription during the cell-cycle. Many of the discovered modules were regulated by sets of transcription factors that are known to be associated or in complexes, such as Swi4/Swi6, Swi6/Mbp1, Swi5/Ace2, and Fkh1/Fkh2/Ndd1/Mcm1. Interestingly, the recovered network suggests that combinatorial factor interactions may provide control that allows for sub-dividing cell-cycle phases into different biological functions. For instance, a module regulated by Mbp1/Swi4/Swi6 contained genes involved with budding and could be localized at the M/G1 boundary. Another module regulated by Mbp1/Swi6 could be localized at almost the same time, but contained a number of genes involved with DNA recombination and repair. A module in the middle of the G1 phase was regulated by Swi4/Swi6 and contained genes involved in cell-wall synthesis, and one module at the G1/S boundary was regulated by Swi4/Fkh2/Ndd1 and contained many genes involved with histone synthesis.

2.6 Discussion

2.6.1 Comparison to previous methods

Several other methods have also been used to discover gene modules. Two methods that also analyzed regulatory networks in yeast produce modules that are directly comparable to those discovered by the GRAM algorithm [162, 99]. These two methods, described in Pilpel *et al.* [162] and Ihmels *et al.* [99] start with an initial set of genes that are selected using a certain criteria (DNA binding motif in the promoter or MIPS functional category) and use expression data to refine the initial set. Figures 2-9 and 2-10 present a comparison between modules discovered by the GRAM algorithm and those generated by the methods of Pilpel *et al.* [162] and Ihmels *et al.* [99]. As can be seen in these figures, our results represent a refinement of the modules from the other studies. In particular, our method identifies not only the genes that participate in a certain module, but also provides evidence as to the factors that are used to activate these genes. Our dynamic sub-network discovery procedure provides further refinement by distinguishing between genes that are regulated differently and participate in different cell-cycle phases.

2.6.2 Limitations of the GRAM algorithm

Although we clearly demonstrated that the GRAM algorithm produces biologically meaningful results and improves on previous methods, several limitations should be pointed out. To begin, the exhaustive search used by the algorithm is efficient for the data set we analyzed, but is infeasible for extremely large data sets of thousands of transcription factors. As more binding data is collected for other organisms such as humans, data sets may approach such sizes.

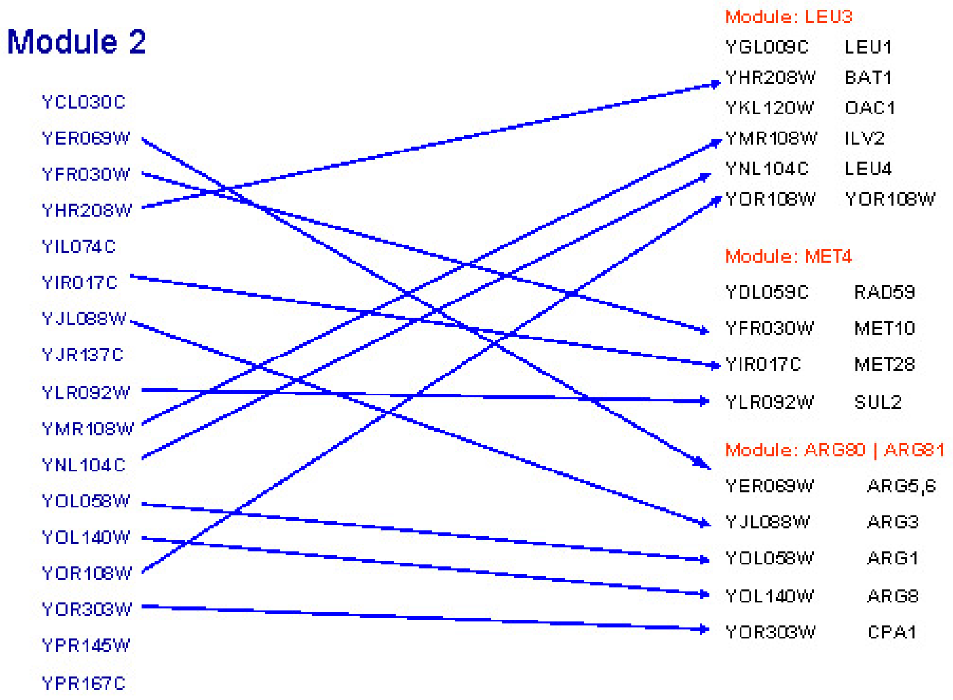


Figure 2-9: Comparison of results of the GRAM algorithm to those of Ihmels *et al.* [99]. Left: a subset of the genes belonging to Module 2 from Ihmels *et al.* Right: three of the amino acid biosynthesis modules identified by the GRAM algorithm. As can be seen, our method improves on results in the Ihmels *et al.* by identifying not only the genes that participate in a certain module, but also provides evidence as to the factors that are used to activate these genes.

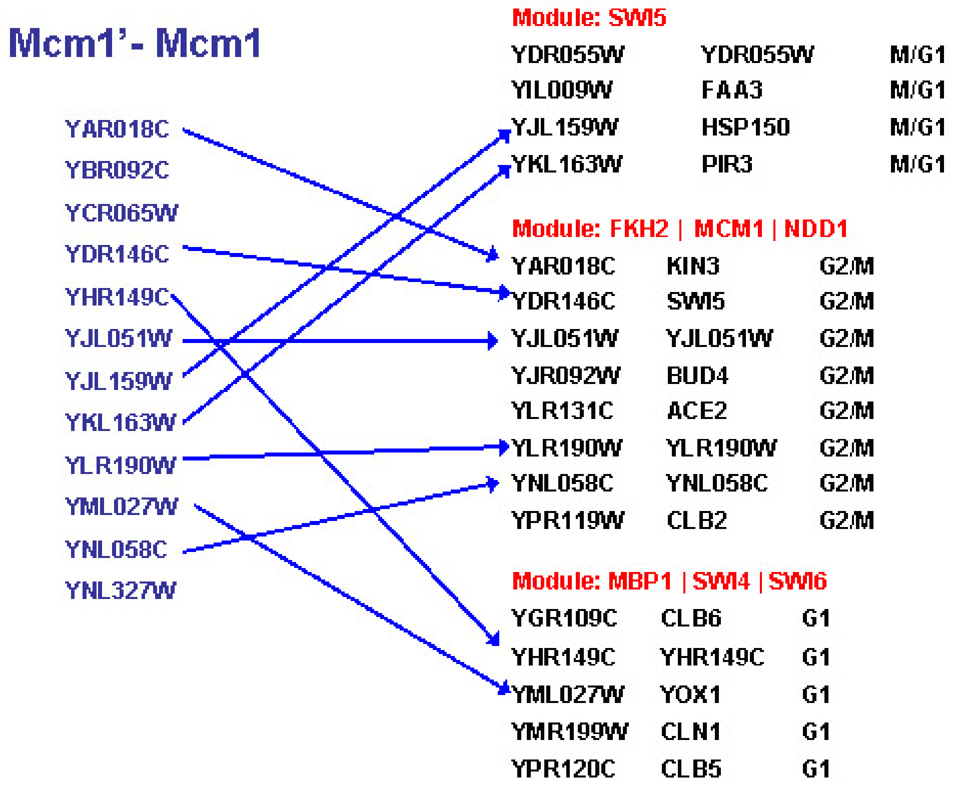


Figure 2-10: Comparison of the results of the GRAM algorithm to those of Pilpel *et al.* [162]. Left: a subset of the genes that belong to the Mcm1′-Mcm1 module from Pilpel *et al.* Right: a subset of the cell-cycle modules discovered by the GRAM algorithm. As can be seen, the modules discovered by our method are a refined version of the module from Pilpel *et al.* Note that our modules differ not only in the set of factors regulating the modules, but also in the different cell-cycle phases to which they belong, providing a better understanding of how the cell regulates the complex expression program that is associated with the cell-cycle system.

Another issue with our approach relates to the method for detecting activators and repressors. The algorithm labeled only a relatively small number of transcription factors as activators or repressors. This may be due to several issues. First, the method used cannot detect the many factors that are post-transcriptionally activated and thus have expression levels that are not expected to fluctuate significantly. Second, we used the entire set of expression experiments to determine the activator/repressor relationships. Although using more data can produce more statistically significant results, it may be that only under certain conditions a factor serves as an activator or repressor. Finally, we required a very high correlation between modules and regulating factors. In general, by relaxing threshold parameters, the algorithm can be used in an exploratory mode to discover more relationships but with less confidence.

2.6.3 Conclusion

In this chapter we presented GRAM, a novel algorithm for discovering modules of genes that are both similarly expressed and regulated by the same set of transcription factors. GRAM integrates expression and binding data to help compensate for technical limitations in either data source alone. We applied GRAM to several biological data sets in order to determine how genes are regulated in cells, and how various systems in the cell respond to external stimuli. In the future, genomic binding data obtained under a variety of conditions is likely to become available and should be of great value in further discovery of genetic regulatory networks. Overall, the GRAM algorithm facilitates a genome-wide approach to analysis of transcriptional regulatory networks that can suggest specific novel regulatory models, which can then be validated in more directed experimental studies.

CHAPTER 3

GeneProgram

"The fear of infinity is a form of myopia that destroys the possibility of seeing the actual infinite, even though it in its highest form has created and sustains us, and in its secondary transfinite forms occurs all around us and even inhabits our minds."

—*Georg Cantor*

An important research problem in computational biology is the identification of *expression programs*, sets of co-activated genes orchestrating physiological processes, and the characterization of the functional breadth of these programs. The use of mammalian expression data compendia for discovery of such programs presents several challenges, including: 1) cellular inhomogeneity within samples, 2) genetic and environmental variation across samples, and 3) uncertainty in the numbers of programs and sample populations.

In this chapter, we present GeneProgram, a new unsupervised computational framework that uses expression data to simultaneously organize genes into overlapping programs and tissues into groups to produce maps of inter-species expression programs, which are sorted by generality scores that exploit the automatically learned groupings. Our method addresses each of the above challenges by using a probabilistic model that: 1) allocates mRNA to different expression programs that may be shared across tissues, 2) is hierarchical, treating each tissue as a sample from a population of related tissues, and 3) uses Dirichlet Processes, a non-parametric Bayesian method that provides prior distributions over numbers of sets while penalizing model complexity. Using synthetic and real gene expression data, we show that GeneProgram outperforms several popular expression analysis methods in recovering coherent and biologically interpretable gene sets. From a large compendium of mouse and human expression data, GeneProgram discovers 19 tissue groups and 100 expression programs active in mammalian tissues.

Our method automatically constructs a comprehensive, body-wide map of expression programs and characterizes their functional generality. This map can be used for guiding future biological experiments, such as discovery of genes for new drug targets that exhibit minimal "cross-talk" with unintended organs, or genes that maintain general physiological responses that go awry in disease states. Further, our method is general, and can be applied readily to novel compendia of biological data.

The remainder of this chapter is organized as follows. In Section 3.1, we provide additional introductory material and discuss prior work relating to discovery of gene sets from expression data compendia. In Section 3.2, we present background material on ordinary and Hierarchical Dirichlet Process mixture models, which are a core component of the GeneProgram probability model. In Section 3.3, we provide a detailed description of the GeneProgram algorithm and probability model. In Section 3.4, we perform synthetic data experiments to explore the kinds of structures GeneProgram and several other well-known unsupervised learning algorithms can recover from noisy data. In Section 3.5, we apply GeneProgram to the Novartis Gene Atlas v2 [205], consisting of expression data for 79 human and 61 mouse tissues. Using this data set, we compare GeneProgram's ability to recover biologically relevant gene sets to that of biclustering methods, and produce a body-wide map of expression programs organized by their functional generality scores. Finally, in Section 3.6, we discuss the significance of our results and comment on possible future research directions.

3.1 Introduction and prior work

The great anatomic and physiologic complexity of the mammalian body arises from the coordinated expression of genes. A fundamental challenge in computational biology is the identification of sets of co-activated genes in a given biological context and the characterization of the functional breadth of such sets. Understanding of the functional generality of gene sets has both practical and theoretical utility. Sets of genes that are very specific to a particular cell type or organ may be useful as diagnostic markers or drug targets. In contrast, sets of genes that are active across diverse cell types can give us insight into unexpected developmental and functional similarities among tissues. While there has been considerable effort in systems biology to understand the structure and organization of co-expressed sets of genes in isolated tissues in the context of pathological processes, such as cancer and infection [107, 128, 189, 215], relatively little attention has been given to this task in the context of normal physiology throughout the entire body [198, 205]. By analyzing gene expression in this latter context, we can gain an understanding of baseline gene expression programs and characterize the specificity of such programs in reference to organism-wide physiological processes.

In this work, we use a large compendium of human and mouse body-wide gene expression data from representative normal tissue samples to discover automatically a set of biologically interpretable expression programs and to characterize quantitatively the specificity of each program. Large genome-wide mammalian expression data compendia present several new challenges that do not arise when analyzing data from simpler organisms. First, tissue samples usually represent collections of diverse cell-types mixed together in different proportions. Even if a sample consists of a relatively homogenous cell population, the cells can still behave asynchronously, due to factors such as microenvironments within the tissue that receive different degrees of perfusion. Second, each tissue sample is often from a different individual, so that the compendium represents a patchwork of samples from different genetic and environmental backgrounds. Finally, the number of expression programs and distinct cell populations present in a compendium is effectively unknown *a priori*.

We present a novel methodology, GeneProgram, designed for analyzing large compendia of mammalian expression data, which simultaneously compresses sets of genes into expression programs and sets of tissues into groups. Specific features of our algorithm address each of the above issues relating to analysis of compendia of mammalian gene expression data. First, our method handles tissue inhomogeneity by allocating the total mRNA recovered from each tissue to different gene expression programs, which may be shared across tissues. The number of expression programs used by a tissue therefore relates to its functional homogeneity. We address the second issue, of tissue samples coming from different individuals, by explicitly modeling each tissue as a sample from a population of related tissues. That is, related tissues are assumed to use similar expression programs and to similar extents, but the precise number of genes and the identity of genes used from each program may vary in each sample. Additionally, populations of related tissues are discovered automatically, and provide a natural means for characterizing the generality of expression programs. Finally, uncertainty in the numbers of tissue groups and expression programs is handled by using a non-parametric Bayesian technique, Dirichlet Processes, which provides prior distributions over numbers of sets.

To understand the novel contributions of the GeneProgram algorithm, it is useful to view our framework in the context of a lineage of unsupervised learning algorithms that have previously been applied to gene expression data. These algorithms are diverse, and can be classified according to various features, such as whether they use matrix factorization methods [8], heuristic scoring functions [42], generative probabilistic models [197], statistical tests [190, 209], or some combinations of these methods [23, 55]. The simplest methods, such as K-means clustering, assume that all genes in a cluster are co-expressed across all tissues, and that there is no overlap among clusters. Next in this lineage are biclustering algorithms [42, 209, 39, 133, 210], which assume that all genes in a bicluster are co-expressed across a subset rather than across all tissues. In many such algorithms, genes can naturally belong to multiple biclusters.

GeneProgram is based on two newer unsupervised learning frameworks, the *topic model* [64, 79] and the Hierarchical Dirichlet Process mixture model [211]. The topic model formalism allows GeneProgram to further relax the assumptions of typical biclustering methods, through a probabilistic model in which each gene in an expression program has a (potentially) different chance of being co-expressed in a subset of tissues. The hierarchical structure of our model, which encodes the assumption that groups of tissues are more likely to use similar sets of expression programs in similar proportions, also provides advantages. Hierarchical models tend to be more robust to noise, because statistical strength is "borrowed" from items in the same group for estimating the parameters of clusters. Additionally, hierarchical models can often be interpreted more easily—in the context of the present application, the inferred expression programs will tend to be used by biologically coherent sets of tissues. Finally, through the Dirichlet Process mixture model formalism, GeneProgram automatically infers the numbers of gene expression programs and tissue groups. Because this approach is fully Bayesian, the numbers of mixture components can be effectively integrated over during inference, and the complexity of the model is automatically penalized. This is in contrast to previous methods that either require the user to specify the number of clusters directly or produce as many clusters as are deemed significant with respect to a heuristic or statistical score without providing a global complexity penalty. We note that Medvedovic *et al.* have also applied Dirichlet Process mixture models to gene expression analysis, but not in the context of topic models, Hierarchical Dirichlet Processes, or mammalian data [139].

As with previous methods [9, 25, 204, 225], we leverage the power of cross-species information to discover biologically relevant sets of co-expressed genes. However, these previous analyses generally required genes to be co-expressed across large sets of experiments [25, 204, 225, 123]. In contrast, GeneProgram uses expression data more flexibly, and is thus able to produce a refined picture of gene expression across species: expression programs may be used by only a subset of tissues, and may be unique to one species or shared across multiple species; tissue groups are similarly flexible. This probabilistic view of expression programs captures the intuition that the general structure of many programs is evolutionarily conserved, but some genes may be interchanged or lost.

3.2 Dirichlet Processes

The task of assigning data to clusters is a classic problem in machine learning and statistics. A common approach to this problem is to construct a model in which data is generated from a mixture of probability distributions.

In finite mixture models, data is assumed to arise from a mixture with a predetermined number of components [138]. The difficulty with such models is that the appropriate number of mixture components is not known *a priori* for many modeling applications. This issue is generally addressed by constructing a series of models with different numbers of components, and evaluating each model using some quality score [138].

An alternate, fully Bayesian approach is to build an *infinite* mixture model, in which the number of mixture components is potentially unlimited, and is itself a random variable that is part of the overall model. Obviously, only a finite number of mixture components can have data assigned to them. However, we still imagine the data as arising from an infinite number of components; as more data is collected, more components may be used to model the data more accurately. Thus, the infinite mixture model is a nonparametric model, in the sense that the number of model parameters grows with the amount of data. The challenge with such a model is how to place an appropriate prior on the infinite number of mixture component parameters and weights.

The Dirichlet Process (DP), a type of stochastic process first introduced in the 1960's [67] and originally of mostly theoretical interest [61, 62], has recently become an important modeling tool as a prior distribution for infinite mixture models[1]. In this section, we will introduce the main concepts of DPs necessary to understand the GeneProgram model. In this regard, we will focus on a constructive definition of DPs in the context of priors for infinite mixture models. This development, which avoids measure theory, closely parallels that presented by [151] and [171].

A recent extension to the standard DP model is the Hierarchical Dirichlet Process (HDP), in which dependencies are specified among a set of DPs by arranging them in a tree structure [211]. HDPs are useful as priors for hierarchical mixture models, in which data is arranged into populations that preferentially share the usage of mixture components. In this section, we will discuss the original HDP formulation by Teh *et al.* in the context of infinite mixture models.

The use of DPs for real-world applications is predicated on practical inference methods. A great advance in this regard has been the development of efficient Markov Chain Monte Carlo (MCMC) methods for approximate inference for infinite mixture models using DP priors [193, 152, 171]. Although other approximate inference methods have been developed [141, 29, 114], MCMC remains the most widely used and versatile method. In particular, efficient MCMC schemes have been developed for HDP models [211], and can be readily extended for the GeneProgram model. Thus, our discussion of DP inference in this section will be restricted to MCMC methods.

The remainder of this section is organized as follows. First, we describe how Dirichlet Processes arise as priors in terms of the infinite limit of mixture models. Next, we describe the extension of DPs to HDPs. Finally, we describe basic MCMC sampling schemes for DPs and HDPs.

3.2.1 Probability models

Bayesian finite mixture models

We begin by defining a typical Bayesian finite mixture model, which we will subsequently extend to the infinite case. Figure 3-2 depicts the model using standard graphical model notation with plates. The model consists of J mixture components, where each component j has associated with it a mixture weight denoted π_j and a parameter vector denoted $\boldsymbol{\theta}_j$. Assume we have N data points denoted x_i, where $1 \leq i \leq N$. Each data point is assigned to a mixture component via an indicator variable z_i, i.e., the probability that data point i is assigned to component j is $\mathrm{p}(\mathsf{z}_i = j \mid \boldsymbol{\pi}) = \pi_j$ or $\mathsf{z}_i \,|\boldsymbol{\pi} \sim \mathrm{Multinomial}(\cdot \mid \boldsymbol{\pi})$. The conditional likelihood for each

[1]It is widely believed that the Dirichlet Process was named after the mathematician, Johann Peter Gustav Lejeune Dirichlet (1805–1859). However, there is some evidence that its name may be related to certain dubious and fantastical tax collection practices of the French government during its colonization of the Ivory Coast (see Figure 3-1). The sole source for this reference is a possibly forged, old, "Ripley's Believe It or Not!" newspaper cartoon, so the veracity of this claim is questionable.

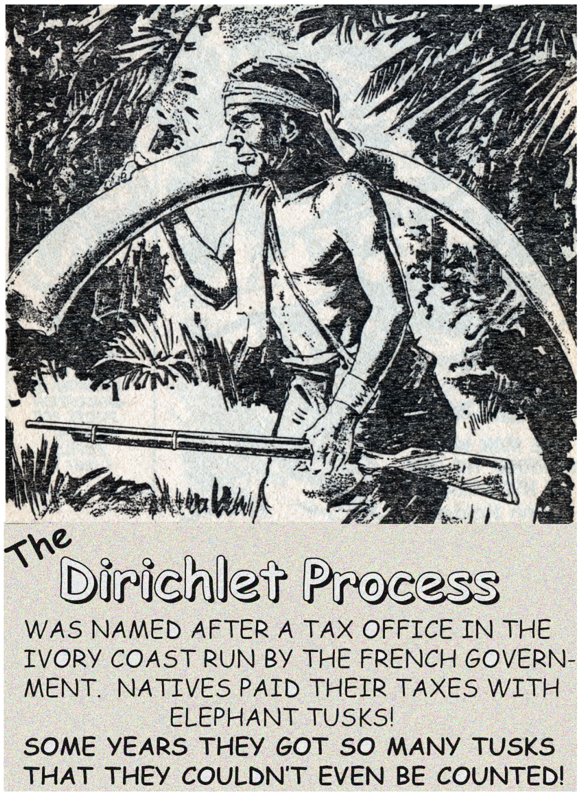

The
Dirichlet Process
WAS NAMED AFTER A TAX OFFICE IN THE IVORY COAST RUN BY THE FRENCH GOVERNMENT. NATIVES PAID THEIR TAXES WITH ELEPHANT TUSKS!
SOME YEARS THEY GOT SO MANY TUSKS THAT THEY COULDN'T EVEN BE COUNTED!

Figure 3-1: The Dirichlet Process may be related to certain dubious and fantastical tax collection practices of the French government during its colonization of the Ivory Coast. The "Ripley's Believe It or Not!" newspaper cartoon shown above lends credence to this claim.

data point may then be written as:

$$\mathrm{p}(\mathsf{x}_i \mid \mathsf{z}_i = j, \boldsymbol{\theta}) = F(\mathsf{x}_i \mid \boldsymbol{\theta}_j)$$

Here, $F(\cdot \mid \cdot)$ is a probability density function parameterized by $\boldsymbol{\theta}$.

To complete the model, we need to define prior distributions over the parameters. We will assume that the component parameters are drawn i.i.d. from some base distribution H, i.e., $\boldsymbol{\theta}_j \sim H(\cdot)$. We also need to specify a prior distribution for the weight parameters. As is typical for Bayesian mixture models, we will assume a symmetric Dirichlet prior on the mixture weights, i.e., $\boldsymbol{\pi} \mid J, \alpha \sim \text{Dirichlet}(\cdot \mid \alpha/J)$. One consequence of using a symmetric prior is that it is not sensitive to the order of the component parameters. Note that the Dirichlet prior is conjugate to the multinomially distributed weights, so that the posterior is also a Dirichlet distribution.

To summarize, our J-dimensional mixture model is defined as:

$$\boldsymbol{\pi} \mid \alpha, J \sim \text{Dirichlet}(\cdot \mid \alpha/J)$$

$$\boldsymbol{\theta}_j \mid H \sim H(\cdot)$$

$$\mathsf{z}_i \mid \boldsymbol{\pi} \sim \text{Multinomial}(\cdot \mid \boldsymbol{\pi})$$
$$\mathsf{x}_i \mid \mathsf{z}_i = j, \boldsymbol{\theta} \sim F(\cdot \mid \boldsymbol{\theta}_j)$$

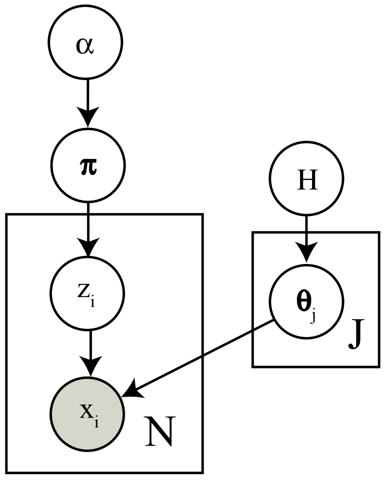


Figure 3-2: A graphical model depiction of a finite mixture model with J mixture components and N data items. Circles represent variables, and arrows denote dependencies among variables. Vectors are depicted with bold type, and observed variables are shown inside shaded circles. Rectangles represent plates, or repeated sub-structures in the model.

In mixture models, we are primarily interested in knowing which component each data point i has been assigned to—the weights $\boldsymbol{\pi}$ are to some extent "nuisance" parameters. It is possible to derive closed form expressions for the data point assignment variable posterior distributions with the mixture weights integrated out. These posterior distributions will be particularly useful in the extension to the infinite mixture model. Note that although the assignment variables $\mathbf{z}$ are conditionally independent given the weights, they become dependent if we integrate out the weights (i.e., the probability of assigning a data point to a particular component depends on the assignments of all other data points). As it turns out, the probability of assigning data point i to some component j given assignments of all other data points can be written as a simple closed form expression (see [171]):

$$\text{p}(\mathsf{z}_i = j \mid \mathbf{z}_{-i}, \alpha, J) = \int \text{p}(\mathsf{z}_i = j \mid \boldsymbol{\pi})\text{p}(\boldsymbol{\pi} \mid \mathbf{z}_{-i}, \alpha, J) d\boldsymbol{\pi}$$

$$\text{p}(\boldsymbol{\pi} \mid \mathbf{z}_{-i}, \alpha, J) \propto \text{p}(\mathbf{z}_{-i} \mid \boldsymbol{\pi})\text{p}(\boldsymbol{\pi} \mid \alpha, J)$$

$$\Rightarrow \text{p}(\mathsf{z}_i = j \mid \mathbf{z}_{-i}, \alpha, J) \propto \int \text{p}(\mathsf{z}_i = j \mid \boldsymbol{\pi})\text{p}(\mathbf{z}_{-i} \mid \boldsymbol{\pi})\text{p}(\boldsymbol{\pi} \mid \alpha, J) d\boldsymbol{\pi}$$

$$\Rightarrow \text{p}(\mathsf{z}_i = j \mid \mathbf{z}_{-i}, \alpha, J) \propto \frac{n_j^{-i} + \alpha/J}{N - 1 + \alpha} \tag{3.1}$$

Here, $\mathbf{z}_{-i}$ denotes the assignments of all data excluding data point i, and n_j^{-i} denotes the number of data points assigned to component j excluding data point i. The second line of the derivation follows simply from Bayes' theorem. The final line of the derivation follows from conjugacy between the Dirichlet prior on the weights and the multinomial distribution on the assignment variables. Thus, the density function under the integral is that of a non-symmetrical Dirichlet distribution, allowing us to derive the final closed form expression.

Infinite mixture models and Dirichlet Processes

In this subsection we show how the Dirichlet Process arises as a prior for infinite mixture models.

Figure 3-3 depicts an infinite mixture model using standard graphical model notation with plates. As can be seen from the figure, the model is almost structurally identical to the finite version. The distinguishing feature is that the weight and parameter vectors are now infinite dimensional.

The challenge with this model is then to define an appropriate prior for the infinite dimensional parameters and weights. As with any mixture model, the infinite dimensional weights must sum to one. A probability distribution that generates such weights is the *stick-breaking* distribution, denoted Stick(α), where α is a scaling or concentration parameter (discussed in more detail below). This distribution is defined constructively. Intuitively, we imagine starting with a stick of unit length and breaking it at a random point. We retain one of the pieces, and break the second piece again at a random point. This process is repeated infinitely, producing a set of random weights that sum to one with probability one [193]. To be more precise, the jth weight π_j is constructed as:

$$\pi'_j \mid \alpha \sim \text{Beta}(1, \alpha)$$

$$\pi_j = \pi'_j \prod_{l=1}^{j-1} (1 - \pi'_l)$$

The infinite mixture model can be constructed using the stick-breaking distribution as a prior on the mixture weights and the base distribution H as a prior on the component parameters. This can be summarized as:

$$\boldsymbol{\pi} \mid \alpha \sim \text{Stick}(\alpha)$$

$$\boldsymbol{\theta}_j \mid H \sim H(\cdot)$$

$$\mathsf{z}_i \mid \boldsymbol{\pi} \sim \text{Multinomial}(\cdot \mid \boldsymbol{\pi})$$

$$\mathsf{x}_i \mid \mathsf{z}_i = j, \boldsymbol{\theta} \sim F(\cdot \mid \boldsymbol{\theta}_j)$$

Note that this construction produces a vector $\boldsymbol{\pi}$ with a countably infinite number of dimensions, whose components all sum to one, and H is sampled independently a countably infinite number of times to generate the mixture component parameter values.

To establish the connection between Dirichet Processes and the model described

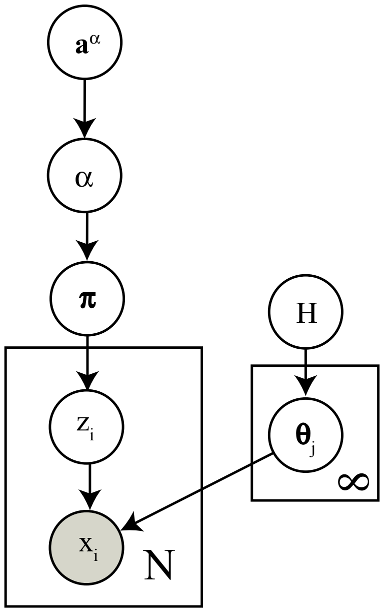


Figure 3-3: A graphical model depiction of the infinite mixture model. Circles represent variables, and arrows denote dependencies among variables. Vectors are depicted with bold type, and observed variables are shown inside shaded circles. Rectangles represent plates, or repeated sub-structures in the model.

above, we consider the distribution over all possible component parameter values for the infinite mixture model. This distribution will be non-zero at a countably infinite number of values. Formally, we denote this distribution by G and can write it as:

$$G(\boldsymbol{\psi}) = \sum_{j=1}^{\infty} \pi_j \delta(\boldsymbol{\psi} - \boldsymbol{\theta}_j)$$

Here, ψ is an arbitrary parameter value, and $\delta(\cdot)$ is the standard delta-function, which is non-zero only when its argument is zero.

Each distribution G thus constructed can be viewed as a sample from a stochastic process, which can in fact be proven to be the Dirichlet Process (see [102] and [193]). In general, we will characterize a Dirichlet Process by a scalar parameter α, called the concentration parameter, and a base distribution H. A sample from a Dirichlet Process, which we denote $G\,|\alpha, H \sim \mathrm{DP}(\alpha, H)$, is thus a distribution that is non-zero over a countably infinite number of values (with probability one). As we have seen, each sample effectively parameterizes an infinite dimensional mixture model.

The concentration parameter α affects the expected number of mixture components containing data items when the DP is used as a prior for the infinite mixture model. As shown in [12], the expected number of non-empty mixture components J

depends only on α and the number of data points N:

$$E[J \mid \alpha, N] = \alpha \sum_{l=J-1}^{N} \frac{1}{\alpha + l - 1} \approx \alpha \ln\left(\frac{N+\alpha}{\alpha}\right)$$

Thus, we see that the number of non-empty components grows approximately as the logarithm of the size of the data set. Further, we see that the number of components grows as the concentration parameter α increases.

To make our model fully Bayesian, we would like to treat the concentration parameter α as a random variable and place a prior on it. The Gamma distribution is commonly used as a prior for α, in part because efficient inference is possible with this prior, and also because appropriate parameter choices result in a relatively uninformative prior [151]. Thus, we place a Gamma prior on α with hyperparameters $\boldsymbol{a}^{\alpha}$, i.e., $\alpha \mid \boldsymbol{a}^{\alpha} \sim \text{Gamma}(a_1^{\alpha}, a_2^{\alpha})$.

Hierarchical Dirichlet Process models

In this section, we describe the Hierarchical Dirichlet Process (HDP) models introduced by Teh *et al.* [211]. As in the previous section on DPs, we will present HDPs in terms of priors for infinite mixture models. We will describe only a two-level hierarchical model for clarity; additional levels are simply added by applying the model construction process recursively.

Figure 3-4 depicts a basic HDP using standard graphical model notation with plates. In HDP models, we assume that data is divided into T subsets, each consisting of N_t data points denoted x_{ti}, where $1 \leq t \leq T$ and $1 \leq i \leq N_t$. Each such data set division is modeled by an infinite mixture model with weights $\boldsymbol{\pi}_t$ and component assignment variables z_{ti}. These infinite mixture models are not independent; the mixtures share component parameters $\boldsymbol{\theta}$ and a common Dirichlet Process prior.

The dependencies among the infinite mixture models can be understood in terms of a construction using the stick-breaking distribution. Beginning at the top of the model, we imagine drawing a sample G from a Dirichlet Process, i.e., $G \mid \alpha_0, H \sim \text{DP}(\alpha_0, H)$. Recall that we can write this sample as:

$$G(\boldsymbol{\psi}) = \sum_{j=1}^{\infty} \beta_j^0 \delta(\boldsymbol{\psi} - \boldsymbol{\theta}_j)$$

Here, $\boldsymbol{\theta}_j$ are drawn i.i.d. from the base distribution H, and $\boldsymbol{\beta}^0 \mid \alpha_0 \sim \text{Stick}(\alpha_0)$.

We next form a second DP using the sample G itself as a base distribution, i.e., we construct $\text{DP}(\alpha_1, G)$. We then generate i.i.d. samples from this DP for each of the T sub-models, i.e., $G_t \mid \alpha_1, G \sim \text{DP}(\alpha_1, G)$. Each sample can be written as:

$$G_t(\boldsymbol{\psi}) = \sum_{j=1}^{\infty} \pi_{tj} \delta(\boldsymbol{\psi} - \boldsymbol{\theta}_j)$$

Notice that these distributions must necessarily be non-zero only at the same points

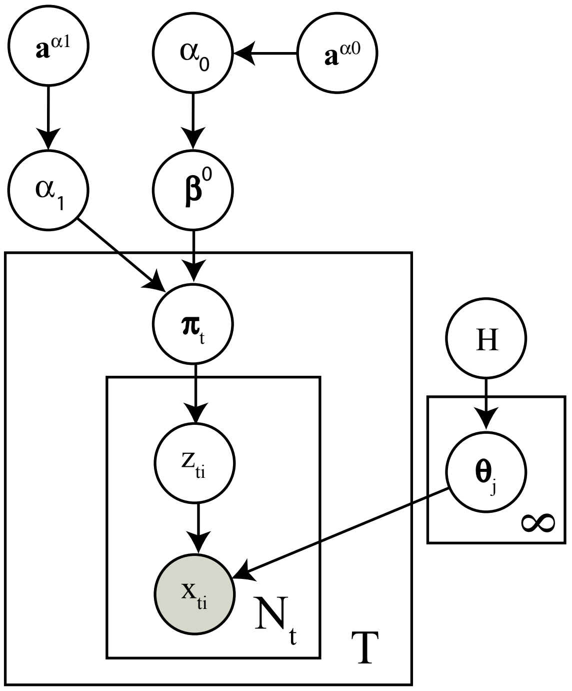


Figure 3-4: A graphical model depiction of the Hierarchical Dirichlet Process represented as an infinite mixture model. Circles represent variables, and arrows denote dependencies among variables. Vectors are depicted with bold type, and observed variables are shown inside shaded circles. Rectangles represent plates, or repeated sub-structures in the model.

$\boldsymbol{\theta}_j$ as G is. We have now constructed a set of T dependent infinite mixture models, where each model has separate (but dependent) weights $\boldsymbol{\pi}_t$ and shared component parameters $\boldsymbol{\theta}$.

It can be shown that the weights $\boldsymbol{\pi}_t$ can be constructed via a stick-breaking process using the top-level weights $\boldsymbol{\beta}^0$ (see [211]):

$$\pi'_{tj} \sim \text{Beta}\left(\alpha_1 \beta^0_j, \alpha_1 \left(1 - \sum_{l=1}^{j} \beta^0_l\right)\right)$$

$$\pi_{tj} = \pi'_{tj} \prod_{l=1}^{j-1} (1 - \pi'_{tl})$$

3.2.2 Markov Chain Monte Carlo approximate inference

Single level infinite mixture models

Markov Chain Monte Carlo (MCMC) algorithms are general tools for approximating posterior distributions of models. With these methods, one alternately samples from the distributions for subsets of variables conditioned on the remaining variables. Given some mild constraints on the model distributions, the approximation converges to the true posterior distribution in the large sample limit [72]. The utility of MCMC methods hinges on the ability to sample from a set of conditional distributions more efficiently than sampling from the full posterior.

In the case of infinite mixture models using a DP prior, sampling can be made efficient by exploiting a "trick" that requires tracking of only a finite number of non-empty mixture components and the data points already assigned to them. Figure 3-5 presents the overall MCMC sampling scheme for single level infinite mixture models.

```
Repeat for all data items $i = 1 \dots N$:
    Sample $\mathsf{z}_i$, the assignment of the data item to a mixture component,
    from its posterior, i.e., $\mathrm{p}(\mathsf{z}_i \mid \mathbf{z}_{-i}, \alpha, \boldsymbol{\theta})$
        If the data item has been assigned to a new component, sample a new
        mixture component parameter $\boldsymbol{\theta}_*$ from its posterior

Repeat for all non-empty mixture components $j = 1 \dots J$:
    Sample the component parameter $\boldsymbol{\theta}_j$ from its posterior

Sample the DP concentration parameter $\alpha$ from its posterior
```

Figure 3-5: One iteration of the basic MCMC sampling scheme for an infinite mixture model using a Dirichlet Process prior.

The key MCMC sampling step for Dirichlet Processes involves picking assignments of data points to mixture components. We sample the assignment of a data point i conditioned on the other variables from the distribution given by:

$$\mathrm{p}(\mathsf{z}_i \mid \mathbf{z}_{-i}, \alpha, \boldsymbol{\theta}, \mathbf{x}) \propto \mathrm{p}(\mathsf{z}_i \mid \mathbf{z}_{-i}, \alpha)\mathrm{p}(\mathbf{x} \mid \mathbf{z}, \boldsymbol{\theta}) \tag{3.2}$$

The proportionality simply follows from Bayes' theorem. Recall from equation 3.1 that for finite mixture models, we can write $\mathrm{p}(\mathsf{z}_i = j \mid \mathbf{z}_{-i}, \alpha, J)$ in closed form:

$$\mathrm{p}(\mathsf{z}_i = j \mid \mathbf{z}_{-i}, \alpha, J) \propto \frac{n_j^{-i} + \alpha/J}{N - 1 + \alpha}$$

For the case of infinite mixture models, and in which $n_j^{-i} > 0$ (i.e., the jth component of the mixture is non-empty), it can be proven that this distribution converges to (see [171]):

$$\mathrm{p}(\mathsf{z}_i = j \mid \mathbf{z}_{-i}, \alpha) \propto \frac{n_j^{-i}}{N - 1 + \alpha} \tag{3.3}$$

For infinite mixture models, we must consider the probability that a data point does not belong to one of the mixture components containing other data points. That is, we will need to calculate $\mathrm{p}(\mathsf{z}_i \neq \mathsf{z}_l, \forall\, l \neq i \mid \mathbf{z}_{-i}, \alpha)$. It can be proven that this probability is given by (see [171]):

$$\mathrm{p}(\mathsf{z}_i \neq \mathsf{z}_l, \forall\, l \neq i \mid \mathbf{z}_{-i}, \alpha) \propto \frac{\alpha}{N - 1 + \alpha} \tag{3.4}$$

We can thus combine equations 3.2, 3.3 and 3.4 to obtain the posterior distributions for the assignment variables:

$$\mathrm{p}(\mathsf{z}_i = j \mid \mathbf{z}_{-i}, \alpha, \boldsymbol{\theta}, \mathbf{x}) \propto \frac{n_j^{-i}}{N-1+\alpha}\mathrm{p}(\mathsf{x}_i \mid \boldsymbol{\theta}_j) \quad \text{for } n_j^{-i} > 0 \tag{3.5}$$

$$\mathrm{p}(\mathsf{z}_i \neq \mathsf{z}_l, \forall\, l \neq i \mid \mathbf{z}_{-i}, \alpha, \boldsymbol{\theta}, \mathbf{x}) \propto \frac{\alpha}{N-1+\alpha}\int F(\mathsf{x}_i \mid \boldsymbol{\psi})H(\boldsymbol{\psi})d\boldsymbol{\psi} \tag{3.6}$$

Thus, for each iteration, we sample the mixture component assignments for all data points using equations 3.5 and 3.6. For the first J components already containing data items, we use equation 3.5 to compute the assignment probability. We use equation 3.6 to compute the probability of assigning the data point to a new mixture component. Notice that in equation 3.6, we integrate over the mixture component parameters, as any component parameters are possible for a new component. Sampling is most efficient when $F(\cdot)$ and $H(\cdot)$ are conjugate. However, in cases of non-conjugacy of these distributions, Monte Carlo methods may be used [152, 171].

We also need to sample from the posterior for the concentration parameter α. It can be shown that the conditional distribution for α is given by (see [151]):

$$\mathrm{p}(\alpha \mid J, N, \boldsymbol{a}^{\alpha}) \propto \alpha^{a_1^{\alpha}+J-1}e^{-a_2^{\alpha}\alpha}\mathrm{B}(\alpha, N)$$

Here, $\mathrm{B}(\cdot,\cdot)$ is the standard Beta function defined as:

$$\mathrm{B}(u,v) = \frac{\Gamma(u)\Gamma(v)}{\Gamma(u+v)} = \int_0^1 \eta^{u-1}(1-\eta)^{v-1}d\eta$$

Escobar and West describe an efficient sampling scheme for α [60]. They noted that $\mathrm{p}(\alpha \mid J, N)$ can be written as a marginalization over an auxiliary variable η:

$$\mathrm{p}(\alpha \mid J, N, \boldsymbol{a}^{\alpha}) \propto \int_0^1 \mathrm{p}(\alpha, \eta \mid J, N, \boldsymbol{a}^{\alpha})d\eta$$

$$\mathrm{p}(\alpha, \eta \mid J, N, \boldsymbol{a}^{\alpha}) \propto \alpha^{a_1^{\alpha}+J-1}e^{-a_2^{\alpha}\alpha}\eta^{\alpha-1}(1-\eta)^{N-1}$$

From the joint distribution, we can see that:

$$\mathrm{p}(\alpha \mid \eta, J, N, \boldsymbol{a}^{\alpha}) \propto \mathrm{Gamma}(\alpha \mid a_1^{\alpha}+J-1, a_2^{\alpha}-\ln\eta)$$

$$\mathrm{p}(\eta \mid \alpha, J, N) \propto \mathrm{Beta}(\eta \mid \alpha, N)$$

Thus, by sampling from the above two conditional distributions, we can sample from the posterior for α to update the concentration parameter during the MCMC sampling iterations.

Hierarchical Dirichlet Process models

Teh *et al.* described an MCMC method for HDP infinite mixture models that uses auxiliary variables to make sampling from the conditional distributions efficient [211]. Figure 3-6 provides an overview of the sampling scheme.

```
Repeat for all data subsets $t = 1 \ldots T$ and data items $i = 1 \ldots N$:
    Sample $\mathsf{z}_{ti}$, the assignment of data item $i$ from subset $t$ to a mixture component,
    from its posterior, i.e., $\mathrm{p}(\mathsf{z}_{ti} \mid \mathbf{z}_{-i}, \boldsymbol{\beta}^0, \boldsymbol{\theta}, \mathbf{x}, \alpha_1)$
        If the data item has been assigned to a new component, sample a new
        top-level mixture weight $\beta^0_*$ from the stick-breaking distribution and
        a new mixture component parameter $\boldsymbol{\theta}_*$ from its posterior

Repeat for all non-empty mixture components $j = 1 \ldots J$:
    Sample the component parameter $\boldsymbol{\theta}_j$ from its posterior

Sample the top-level mixture weights $\boldsymbol{\beta}^0$ from their posterior

Sample the concentration parameters $\alpha_0$ and $\alpha_1$ from their posteriors
```

Figure 3-6: One iteration of the basic MCMC sampling scheme for the Hierarchical Dirichlet Process mixture model with two levels.

The first task is to sample the data point assignment variables, $\mathbf{z}$. The method for this is similar to that used for ordinary Dirichlet Process mixture models. We begin by considering a finite mixture model of dimension J and integrating out the individual mixture weights $\boldsymbol{\pi}_t$ to obtain the conditional probability of $\mathbf{z}$ given $\boldsymbol{\beta}^0$:

$$\mathrm{p}(\mathbf{z} \mid \boldsymbol{\beta}^0, \alpha_1) = \prod_{t=1}^{T} \frac{\Gamma(\alpha_1)}{\Gamma(\alpha_1 + N_t)} \prod_{j=1}^{J} \frac{\Gamma(\alpha_1 \beta^0_j + n_{tj})}{\Gamma(\alpha_1 \beta^0_j)} \tag{3.7}$$

Here, N_t denotes the number of data items in subset t, and n_{tj} represents the number of data items from subset t assigned to mixture component j. It can be shown that in the limit of an infinite mixture model, the conditional probability has a particularly simple form:

$$\mathrm{p}(\mathsf{z}_{ti} = j \mid \mathbf{z}_{-i}, \boldsymbol{\beta}^0, \alpha_1) \propto \alpha_1 \beta^0_j + n^{-i}_{tj}$$

By combining this with the conditional likelihood for data points, $F(\cdot \mid \cdot)$, we obtain the posterior distribution for assigning data points to mixture components:

$$\mathrm{p}(\mathsf{z}_{ti} = j \mid \mathbf{z}_{-i}, \boldsymbol{\beta}^0, \boldsymbol{\theta}, \mathbf{x}, \alpha_1) \propto (\alpha_1 \beta^0_j + n^{-i}_{tj}) F(\mathsf{x}_{ti} \mid \boldsymbol{\theta}_j) \tag{3.8}$$

This equation holds if j is a non-empty component. The posterior distribution for assigning a data point to a new component is given by:

$$\mathrm{p}(\mathsf{z}_{ti} \neq \mathsf{z}_{tl} \; \forall \, t, l \neq i \mid \mathbf{z}_{-i}, \boldsymbol{\beta}^0, \boldsymbol{\theta}, \mathbf{x}, \alpha_1) \propto (\alpha_1 \beta^0_*) \int F(\mathsf{x}_{ti} \mid \boldsymbol{\psi}) H(\boldsymbol{\psi}) d\boldsymbol{\psi} \tag{3.9}$$

Here, we define $\beta^0_* = 1 - \sum_{l=1}^{J} \beta^0_l$, where there are J components with data points assigned to them. As with ordinary DPs, Monte Carlo methods may be used if $F(\cdot \mid \cdot)$ and $H(\cdot)$ are non-conjugate distributions.

So, to sample the data point assignments we use equations 3.8 and 3.9. If a data point is assigned to a new component, we must also generate a new weight β^0_{J+1} using the stick-breaking distribution, i.e., we sample $b \sim \text{Beta}(1, \alpha_0)$ and set $\beta^0_{J+1} \leftarrow b\beta^0_*$.

To sample from the model posterior, we also must sample the top-level weights $\boldsymbol{\beta}^0$. The method for this relies on a "trick" using auxiliary variables. For the derivation, we need to use a general property of ratios of Gamma functions given by:

$$\frac{\Gamma(n+a)}{\Gamma(a)} = \sum_{m=0}^{n} s(n,m) a^m \tag{3.10}$$

Here, n and a are natural numbers. In equation 3.10, the ratio of Gamma functions has been expanded into a polynomial with a coefficient $s(n,m)$ for each term. These coefficients are called unsigned Stirling numbers of the first kind, which count the permutations of n objects having m permutation cycles (see [2]). By definition, $s(0,0) = 1$, $s(n,0) = 0$, $s(n,n) = 1$ and $s(n,m) = 0$ for $m > n$. Additional coefficients are then computed recursively using the equation $s(n+1,m) = s(n,m-1) + ns(n,m)$.

Note that the $\boldsymbol{\beta}^0$ weights in the conditional probability $\text{p}(\mathsf{z} \mid \boldsymbol{\beta}^0)$ in equation 3.7 occur as arguments of ratios of Gamma functions. These ratios can be expanded to yield polynomials in the $\boldsymbol{\beta}^0$ weights:

$$\frac{\Gamma(\alpha_1 \beta^0_j + n_{tj})}{\Gamma(\alpha_1 \beta^0_j)} = \sum_{m_{tj}=0}^{n_{tj}} s(n_{tj}, m_{tj}) (\alpha_1 \beta^0_j)^{m_{tj}} \tag{3.11}$$

An efficient sampling method can be derived by introducing m as auxiliary variables. The conditional distributions for sampling m and $\boldsymbol{\beta}^0$ can be shown to be:

$$\text{p}(\mathsf{m}_{tj} = m \mid \mathsf{z}, \mathsf{m}_{-tj}, \boldsymbol{\beta}^0) \propto s(n_{tj}, m)(\alpha_1 \beta^0_j)^m \tag{3.12}$$

$$\text{p}(\boldsymbol{\beta}^0 \mid \mathsf{z}, \mathsf{m}) \propto (\beta^0_*)^{\alpha_0 - 1} \prod_{j=1}^{J} \beta_j^{\sum_t m_{tj} - 1} \propto \text{Dirichlet}(\sum_t m_{t1}, \ldots, \sum_t m_{tJ}, \alpha_0) \tag{3.13}$$

Finally, we need to sample the concentration parameters α_0 and α_1 for the HDP. As with the regular DP model, we will assume Gamma priors on the concentration parameters.

For α_0, it can be shown that:

$$\text{p}(\mathsf{J} = J \mid \alpha_0, \mathsf{m}) \propto s(M, J) \alpha_0^J \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + M)}$$

Here, $M = \sum_t \sum_j m_{tj}$ and J is the number of non-empty mixture components. Combining the above equation with the prior for α_0 yields the conditional probability for α_0, which can be sampled using the same method as described for sampling concentration parameters for regular DPs.

Sampling α_1 requires the introduction of two additional auxiliary variables w and b. The following update equations can then be derived:

$$p(\mathsf{w}_t \mid \alpha_1) \propto w_t^{\alpha_1}(1-w_t)^{N_t-1}$$

$$p(\mathsf{b}_t \mid \alpha_1) \propto \left(\frac{N_t}{\alpha_1}\right)^{b_t}$$

$$p(\alpha_1 \mid \mathbf{w}, \mathbf{b}, \boldsymbol{a}^{\alpha_1}) \propto \text{Gamma}(a_1^{\alpha_1} + \sum_{t=1}^{T}(M_t - b_t), a_2^{\alpha_1} - \sum_{t=1}^{T}\log w_t)$$

Here, $a_1^{\alpha_1}$ and $a_2^{\alpha_1}$ are the hyperparameters for the Gamma prior on α_1 and $M_t = \sum_{j=1}^{J} m_{tj}$.

3.3 The GeneProgram algorithm and probability model

3.3.1 Algorithm overview

The GeneProgram algorithm consists of data integration (pre-processing), model inference, and distribution summary steps as depicted in Figure 3-7. Data integration makes data from multiple species comparable and discretizes it in preparation for input to the model. The first data integration step combines replicates and normalizes microarray data to make measurements of gene expression comparable across tissues. The second data integration step uses a pre-defined homology map to convert species specific gene identifiers into meta-gene identifiers. Meta-genes are virtual genes that correspond one-to-one with genes in each species. Some genes do not have counterparts in other species, and these are filtered out. In the final data integration step, continuous expression measurements are discretized. The model inference step seeks to discover underlying expression programs and tissue groups in the data probabilistically. To accomplish this, we use Markov Chain Monte Carlo (MCMC) sampling to estimate the model posterior probability distribution. Each posterior sample describes a configuration of expression programs and tissue groups for the entire data set; more probable configurations tend to occur in more samples. The final step of the algorithm is model summarization, which produces consensus descriptions of expression programs and tissue groups from the posterior samples.

3.3.2 The probability model

Intuitive overview

We can understand the GeneProgram probability model intuitively as a series of "recipes" for constructing the gene expression of tissues. Figure 3-8 presents a cartoon of this process, in which we imagine that we are generating the expression data for the

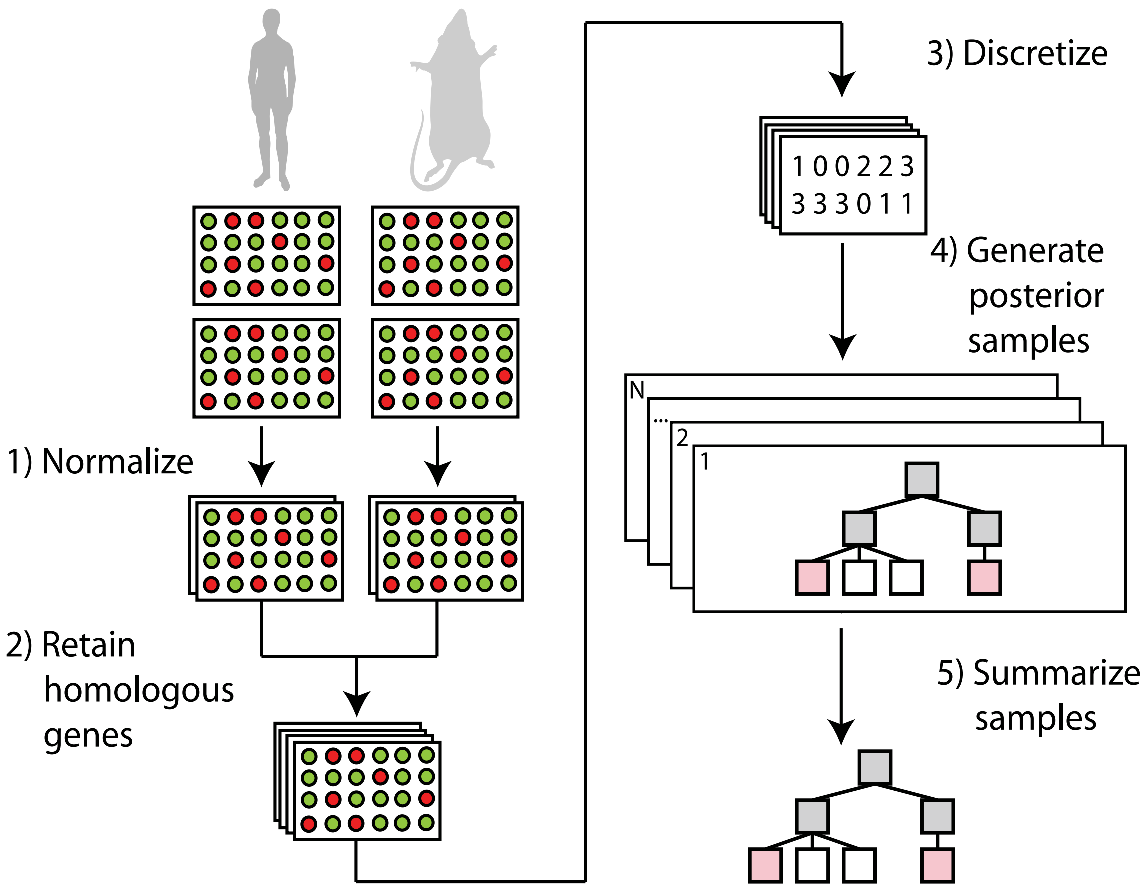


Figure 3-7: GeneProgram algorithm steps. The main steps of the algorithm are: data integration (steps 1-3), model inference (step 4), and posterior sample summarization (step 5). See the text for details.

digestive tract of a person. The digestive tract is composed of a variety of cell types, with cells of a given type living in different microenvironments, and thus expressing somewhat different sets of genes. We can envision each cell in an organ choosing to express a subset of genes from relevant expression programs; some programs will be shared among many cell types and others will be more specific. As we move along the digestive tract, the cell types present will change and different expression programs will become active. However, based on the similar physiological functions of the tissues of the digestive tract, we expect more extensive sharing of expression programs than we would between dissimilar organs such as the brain and kidneys. As can be seen in Figure 3-8, the final steps of our imaginary data generation experiment involve organ dissection, homogenization, cell lysis and nucleic acid extraction, to yield the total mRNA expressed in the tissue, which is then measured on a DNA microarray.

The conceptual experiment described above for "constructing" collections of mRNA molecules from tissues is analogous to the *topic model*, a probabilistic method developed for information retrieval applications [79, 31] and also applied to other domains, such as computer vision [206, 207] and haploinsufficiency profiling [64]. In topic models for information retrieval applications, documents are represented as unordered collections of words, and documents are decomposed into sets of related words called topics that may be shared across documents. In hierarchical versions of such models, documents are further organized into categories and topics are preferentially shared within the same category. In the GeneProgram model, a unit of mRNA detectable on a microarray is analogous to an individual word in the topic model. Related tissue populations (tissue groups) are analogous to document categories, tissues are analogous to documents, and topics are analogous to expression programs.

GeneProgram handles uncertainty in the numbers of expression programs and tissue groups by using a model based on Hierarchical Dirichlet Processes [211]. We note that in the original Hierarchical Dirichlet Processes formulation [211], data items were required to be manually assigned to groups. The GeneProgram model extends this work, automatically determining the number of groups and tissue memberships in the groups.

The GeneProgram probability model consists of a three-level hierarchy of Dirichlet Processes, as depicted in Figure 3-9 part A. Tissues are at the lowest level in the hierarchy. Each tissue is characterized by a mixture (weighted combination) of expression programs that is used to describe the observed gene expression levels in the tissue. An expression program represents a set of cross-species meta-genes that are co-activated to varying extents, as depicted in Figure 3-9 part B. When a tissue uses an expression program, the homology map translates meta-genes into the appropriate species specific genes. Tissues differ in terms of which expression programs they employ and how the programs are weighted. The middle level of the hierarchy consists of tissue groups, in which each group represents tissues that are similar in their use of expression programs. The highest and root level in the hierarchy describes a base level mixture of expression programs that is not tissue or group specific.

Each node in our hierarchical model maintains a mixture of gene expression programs, and the mixtures at the level below are constructed on the basis of those

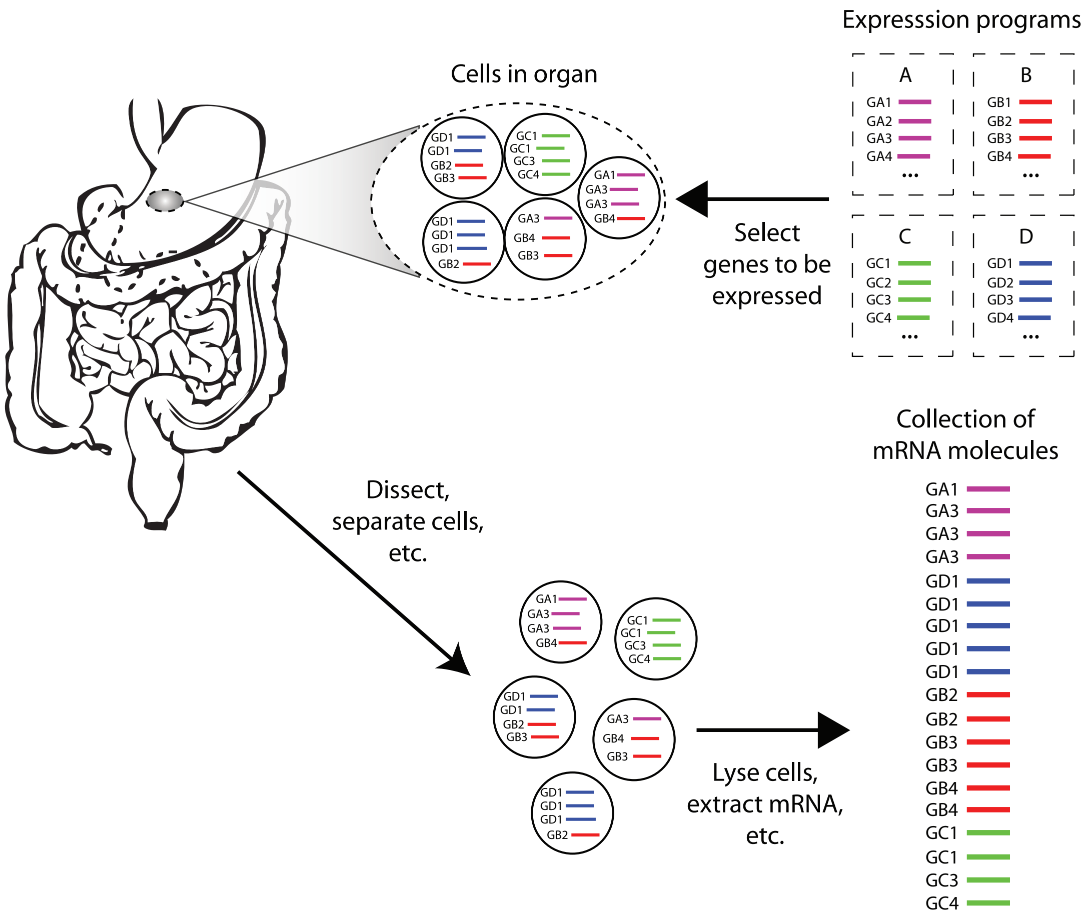
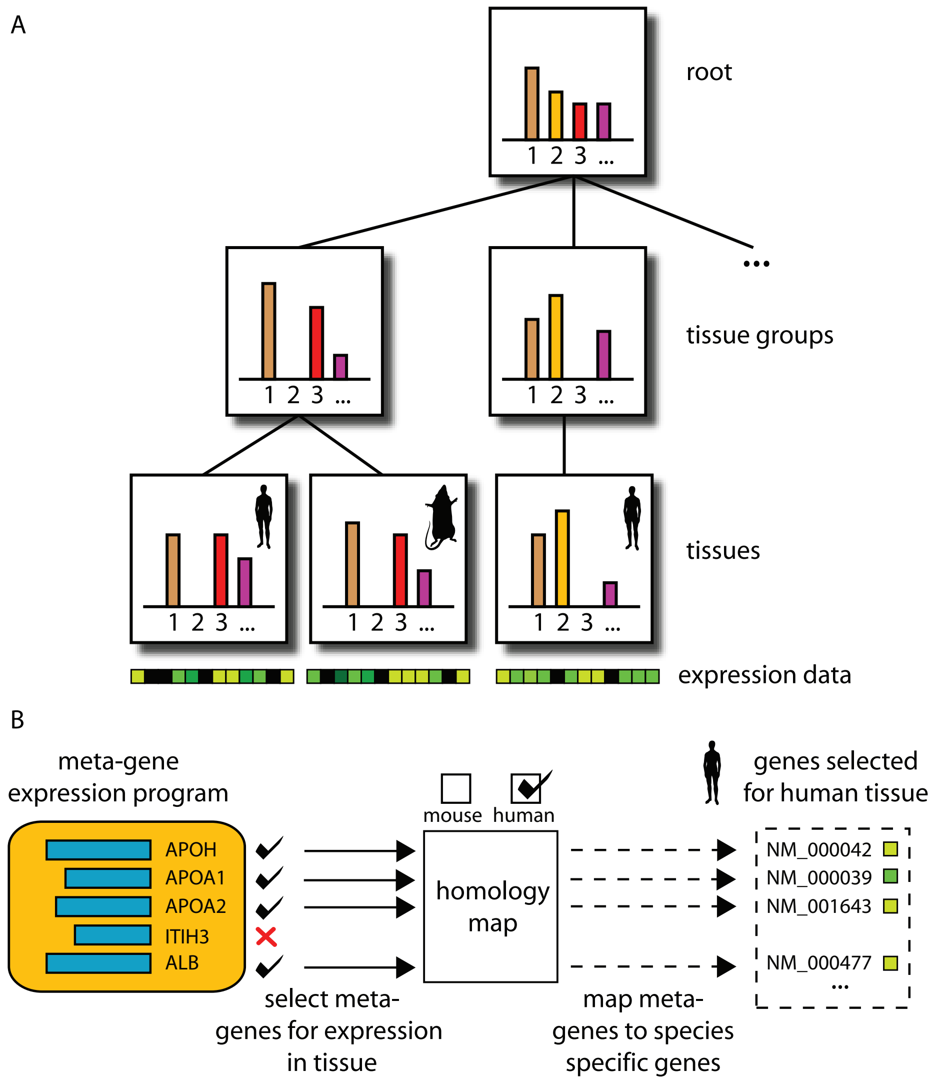


Figure 3-8: Conceptual overview of the data generation process for gene expression in mammalian tissues. The GeneProgram probability model can be thought of as a series of "recipes" for constructing the gene expression of tissues, as depicted in this cartoon example for a digestive tract. In the upper right, four expression programs (labeled A-D) are shown, consisting of sets of genes (e.g., GA1 represents gene 1 in program A). Cells (circles) throughout the digestive tract choose genes to be expressed probabilistically from the programs. The biological experimenter than collects mRNA by dissecting out the appropriate organ, taking a tissue sample, homogenizing it, lysing cells, and extracting the nucleic acids.

above. Thus, a tissue is decomposed into a collection of gene expression programs, which are potentially shared across the entire model, but are more likely to be shared by related tissues (those in the same tissue group). Because our model uses Dirichlet Processes, the numbers of both expression programs and tissue groups are not fixed and may vary with each sample from the model posterior distribution. In the next section, we describe the GeneProgram probability model in detail.


A
root
1 2 3 ...
tissue groups
tissues
expression data
B
meta-gene expression program
APOH
APOA1
APOA2
ITIH3
ALB
mouse
human
homology map
genes selected for human tissue
NM_000042
NM_000039
NM_001643
NM_000477
select meta-genes for expression in tissue
map meta-genes to species specific genes

Formal model description

The GeneProgram model consists of three levels of DPs. Starting from the leaves these are: tissues, tissue groups, and the root. Each expression program corresponds to a mixture component of the HDP. Because the model is hierarchical, the expression programs are shared by all DPs in the model. An expression program specifies a multinomial distribution over meta-genes. Discrete expression levels are treated anal-

Figure 3-9: *(Part A)* Overview of the GeneProgram probability model. The model is based on Hierarchical Dirichlet Process mixture models, a non-parametric Bayesian method. The model consists of a three-level hierarchy of Dirichlet Processes. Each node describes a weighted mixture of expression programs (each colored bar represents a different program; heights of bars = mixture weights). The mixtures at each level are constructed on the basis of the parent mixtures above. Tissues are at the leaves of the hierarchy, and may be from either species. The observed gene expression of each tissue is characterized by a vector of discretized expression levels of species specific genes (row of small shaded squares below each tissue). *(Part B)* Example of a gene expression program. A gene expression program specifies a set of cross-species meta-genes that are co-activated to varying extents in a subset of tissues. On the left is a simple program containing five meta-genes (colored bars = expression frequencies). In this example, a human tissue uses the gene expression program, choosing four meta-genes from the set with various levels of expression. The homology map (center) translates the meta-genes into species specific genes (right).

ogously to word occurrences in documents in topic models. Thus, a tissue's vector of gene expression levels is converted into a collection of expression events, in which the number of events for a given gene equals the expression level of that gene in the tissue. The model assumes that each gene expression event in a tissue is independently generated by an expression program. In the original HDP formulation [211], the entire tree structure was assumed to be pre-specified. We extend this work, by allowing the model to learn the number of groups and the memberships of tissues in these groups. Thus, groups themselves are generated by a DP, which uses samples from the root process DP as its base distribution.

Figure 3-10 depicts the model using graphical model notation with plates and Table 3.1 summarizes the random variables in the model.

We will begin by describing the model at the level of observed data, and then move up the hierarchy. Assume that there are T tissues and G meta-genes. For simplicity, we will assume that there are also G genes for each species and that the ordering of genes uniquely determines the cross-species mapping. Thus, in the following discussion, genes and meta-genes are used interchangeably. The expression data associated with each tissue t consists of a G-dimensional vector $\mathbf{e}_t$ of discrete expression levels, i.e., $\mathsf{e}_{tg} \in \{0, 1, \ldots, E\}$ is the expression level of gene g in tissue t, where there are E possible discrete expression levels.

A tissue's vector of gene expression levels is converted into a collection of expression events, in which the number of events for a given gene equals the expression level of that gene in the tissue. This representation of expression levels as an unordered "bag of expression events" is analogous to the representation of words in a document as a "bag of words" in topic models. To be precise, let $\mathbf{x}_t$ denote a set of expression events for tissue t, and define a mapping ω from $\mathbf{x}_t$ to genes, where $\omega(\mathsf{x}_{ti}) = g$ iff $\mathsf{e}_{tg} > 0$. The vector $\mathbf{x}_t$ will have N_t elements, where $N_t = \sum_{g=1}^{G} e_{tg}$, i.e., as many elements as there are discrete expression events in the tissue.

The model assumes that each gene expression event in a tissue is independently

Var.	**Dim.**	**Description**	**Cond. distribution or prior**
x_{ti}	1	Expression event i in tissue t; corresponds directly to observed data.	Multinomial, given the assignment to expression program j.
z_{ti}	1	Assignment variable of an expression event to an expression program, i.e., $\mathsf{z}_{ti} = j$ indicates that expression event i in tissue t is assigned to expression program j.	Generated from mixing probabilities over expression programs for the tissue, i.e., $\mathrm{p}(\mathsf{z}_{ti} = j \mid \boldsymbol{\pi}_t) = \pi_{tj}$.
$\boldsymbol{\pi}_t$	∞	Mixing probabilities over expression programs for tissue t.	DP, given the assignment of the tissue to group k, its parent DP mixing probabilities, and its concentration parameter, i.e., $\boldsymbol{\pi}_t \mid \mathsf{q}_t = k, \alpha_1, \boldsymbol{\beta}^k \sim \mathrm{DP}(\alpha_1, \boldsymbol{\beta}^k)$.
$\boldsymbol{\beta}^k$	∞	Mixing probabilities over expression programs at the level of tissue group k; middle level in the DP hierarchy.	DP, given its parent mixing probabilities and concentration parameters, i.e., $\boldsymbol{\beta}^k \mid \alpha_0, \boldsymbol{\beta}^0 \sim \mathrm{DP}(\alpha_0, \boldsymbol{\beta}^0)$.
$\boldsymbol{\beta}^0$	∞	Root level mixing probabilities in the DP heterarchy.	DP, generated from the stick-breaking distribution given its concentration parameter, i.e., $\boldsymbol{\beta}^0 \mid \alpha_0 \sim \mathrm{Stick}(\alpha_0)$.
$\boldsymbol{\theta}_j$	G	Parameters for expression, program j, describing a multinomial distribution over G meta-genes.	Dirichlet distribution prior (parameterized by λ).
λ	1	Pseudo-count parameter for a symmetric Dirichlet distribution.	Gamma distribution prior with a two-dimensional hyperparameter vector $\boldsymbol{a}^{\lambda}$.
q_t	1	Assignment variable of tissues to groups, i.e., $\mathsf{q}_t = k$ indicates that tissue t belongs to tissue group k.	Generated from mixing probabilities over tissue groups, i.e, $\mathrm{p}(\mathsf{q}_t = k \mid \boldsymbol{\epsilon}) = \epsilon_k$.
$\boldsymbol{\epsilon}$	∞	Mixing probabilities over the tissue groups.	DP, generated from the stick-breaking prior given its concentration parameter, i.e, $\boldsymbol{\epsilon} \mid \gamma \sim \mathrm{Stick}(\gamma)$.
α_1	1	Concentration parameter for $\boldsymbol{\pi}_t$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\alpha_1}$.
α_0	1	Concentration parameter for $\boldsymbol{\beta}^0$ and $\boldsymbol{\beta}^k$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\alpha_0}$.
γ	1	Concentration parameter for $\boldsymbol{\epsilon}$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\gamma}$.

Table 3.1: Summary of random variables in the GeneProgram model. The columns are: variable name (vectors are in bold type), dimensions of the variable, description, and the conditional or prior distribution on the variable.

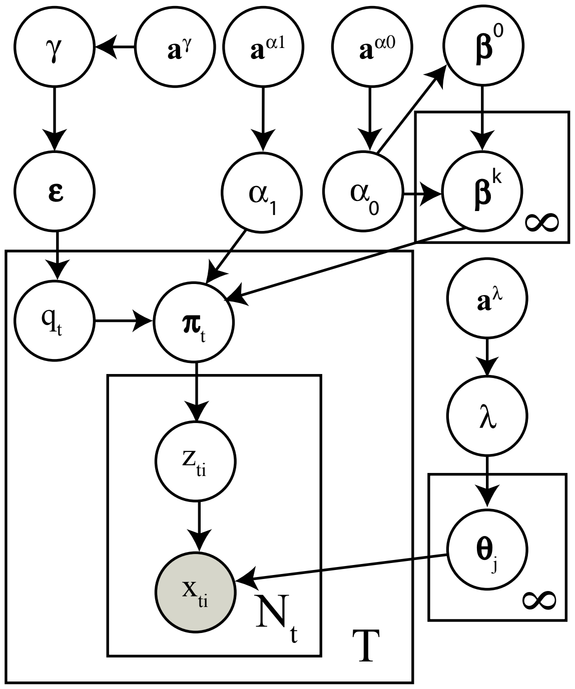


Figure 3-10: The GeneProgram model is depicted using graphical model notation with plates. Circles represent variables, and arrows denote dependencies among variables. Vectors are depicted with bold type, and observed variables are shown inside shaded circles. Rectangles represent plates, or repeated sub-structures in the model. See the text and Table 3.1 for details.

generated by an expression program. The variable z_{ti} assigns gene expression events to programs, i.e., $\mathsf{z}_{ti} = j$ indicates that x_{ti} was generated from the jth expression program. An expression program is a multinomial probability distribution over genes. To be precise, let $\boldsymbol{\theta}_j$ represent a parameter vector of size G for expression program j. Then, the probability of generating expression event x_{ti} corresponding to gene g given that it is assigned to expression program j is $\mathrm{p}(\omega(\mathsf{x}_{ti}) = g \mid \mathsf{z}_{ti} = j, \boldsymbol{\theta}_j) = \theta_{jg}$. We use a symmetric Dirichlet prior for $\boldsymbol{\theta}_j$ with parameter λ, and a Gamma prior for λ with hyperparameter vector $\boldsymbol{a}^{\lambda}$.

The mixing probabilities over expression programs are generated by the DPs in the hierarchy. To be precise, let $\boldsymbol{\pi}_t$ denote the mixing probabilities at the leaf level in the DP hierarchy. That is, $\boldsymbol{\pi}_t$ denotes the mixing probabilities over expression programs for tissue t, i.e., $\mathrm{p}(\mathsf{z}_{ti} = j \mid \boldsymbol{\pi}_t) = \pi_{tj}$. Let $\boldsymbol{\beta}^k$ denote the mixing probabilities at the middle level in the DP hiearchy. That is, $\boldsymbol{\beta}^k$ denotes the mixing probabilities over expression programs at the level of tissue group k. Finally, we let $\boldsymbol{\beta}^0$ denote the root-level mixing probabilities. In the stick-breaking construction for HDP models, it is assumed that root level mixing probabilities are generated by the stick-breaking distribution, i.e., $\boldsymbol{\beta}^0 \mid \alpha_0 \sim \mathrm{Stick}(\alpha_0)$, where $\alpha_0 \sim \mathrm{Gamma}(\boldsymbol{a}^{\alpha_0})$. The hierarchical structure of the model then implies that $\boldsymbol{\beta}^k$ is conditionally distributed as a Dirichlet Process, i.e., $\boldsymbol{\beta}^k \mid \alpha_0, \boldsymbol{\beta}^0 \sim \mathrm{DP}(\alpha_0, \boldsymbol{\beta}^0)$, where we assume that $\boldsymbol{\beta}^k$ also uses concentration parameter α_0.

The tissue level expression program mixing probabilities $\boldsymbol{\pi}_t$ depend on the group that the tissue is assigned to. The variable q_t assigns tissues to groups, i.e., $\mathsf{q}_t = k$ indicates that tissue t belongs to tissue group k and $\mathrm{p}(\mathsf{q}_t = k \mid \boldsymbol{\epsilon}) = \epsilon_k$, where $\boldsymbol{\epsilon}$ represents mixing probabilities over the tissue groups. The mixing probabilities $\boldsymbol{\epsilon}$ over tissue groups are also modeled using a Dirichlet Process. That is, $\boldsymbol{\epsilon} \mid \gamma \sim \mathrm{Stick}(\gamma)$, where γ is a concentration parameter with $\gamma \sim \mathrm{Gamma}(\boldsymbol{a}^{\gamma})$. Given an assignment of tissue t to group k, the tissue level mixing probabilities over expression programs $\boldsymbol{\pi}_t$ are then generated from the middle level mixing probabilities $\boldsymbol{\beta}^k$. That is, $\boldsymbol{\pi}_t \mid \mathsf{q}_t = k, \alpha_1, \boldsymbol{\beta}^k \sim \mathrm{DP}(\alpha_1, \boldsymbol{\beta}^k)$, where α_1 is a concentration parameter with hyperparameters $\boldsymbol{a}^{\alpha_1}$, i.e., $\alpha_1 \sim \mathrm{Gamma}(\boldsymbol{a}^{\alpha_1})$. This completes our formal description of the GeneProgram probability model.

3.3.3 Model inference

The posterior distribution for the model is approximated via Markov Chain Monte Carlo (MCMC) sampling using the follow steps:

1. Sample each assignment of an expression event to an expression program, z_{ti}; create new expression programs as necessary.

2. Sample $\boldsymbol{\beta}^0$ and $\boldsymbol{\beta}^k$ and auxiliary variables for all tissue groups.

3. Sample tissue group assignments q_t for all tissues; create new tissue groups as necessary.

4. Sample concentration parameters α_0, α_1, and γ.

5. Sample expression program Dirichlet prior parameter λ.

Steps 1, 2, and 4 are identical to those described by Teh *et al.* in their auxiliary variable sampling scheme [211] (see Section 3.2.2 for further details). Note that $\mathsf{x}_{ti} \mid \mathsf{z}_{ti} = j, \boldsymbol{\theta}_j \sim \mathrm{Multinomial}(\boldsymbol{\theta}_j)$, and $\boldsymbol{\theta}_j$ is Dirichlet distributed, allowing us to integrate out $\boldsymbol{\theta}_j$ when computing the posterior for z_{ti}. This means that we do not need to represent $\boldsymbol{\theta}_j$ explicitly during sampling. In step 3, we must compute the posteriors for tissue group assignments. This can be written as:

$$\mathrm{p}(\mathsf{q}_t = k \mid \mathbf{z}_t, \mathbf{q}_{-t}, \alpha_0, \gamma, \boldsymbol{\beta}^k) \propto \mathrm{p}(\mathsf{q}_t = k \mid \mathbf{q}_{-t}, \gamma) \prod_{i=1}^{N_t} \int \mathrm{p}(\mathsf{z}_{ti} \mid \boldsymbol{\pi}_t)\mathrm{p}(\boldsymbol{\pi}_t \mid \boldsymbol{\beta}^k, \alpha_0) d\boldsymbol{\pi}_t$$

Here, $\mathbf{q}_{-t}$ denotes all tissue group assignments excluding tissue t. Note that because the conditional distributions for z_{ti} and $\boldsymbol{\pi}_t$ are conjugate, the integral in the above equation can be computed in closed form. Step 5 uses the auxiliary variable sampling method for resampling the parameter for a symmetric Dirichlet prior, as detailed in [60].

We implemented the sampling scheme in Java. Inference was always started with all data assigned to a single expression program. We burned in the sampler for

100,000 iterations, and then collected relevant posterior distribution statistics from 50,000 samples. We set the hyperparameters for all concentration parameters to 10^{-8} to produce vague prior distributions. Both hyperparameters for the Gamma prior on λ were set to 1.0, biasing λ toward a unit pseudo-count Dirichlet distribution.

3.3.4 Summarizing the model posterior probability distribution

Overview

In order to produce interpretable results, GeneProgram needs to create a summary of the model posterior distribution that was approximated using MCMC sampling.

The final step of the GeneProgram algorithm summarizes the approximated model posterior probability distribution with *consensus tissue groups* (CTGs) and *recurrent expression programs* (REPs). The posterior distributions of Dirichlet Process mixture models are particularly challenging to summarize because the number of mixture components may differ for each sample. Previous approaches for summarizing Dirichlet Process mixture model components have used pair-wise co-clustering probabilities as a similarity measure for input into an agglomerative clustering algorithm [139]. This method is feasible if there are a relatively small number of items to be clustered, and we employ it for producing consensus tissue groups. However, this method is not feasible for summarizing expression programs in large data sets because of the number of pair-wise probabilities that would need to be calculated for each sample.

We developed a novel method for summarization of the model posterior distribution, which discovers recurrent expression programs by combining information from similar expression programs that reoccur across posterior samples. Our method is based on the observation that each expression program is significantly used by only a limited number of tissues. Thus, this limited set of tissues serves as a unique signature that allows us to track the expression program across model posterior samples. A recurrent expression program is summarized by the average frequency of expression of meta-genes across many model posterior samples.

Detailed description of recurrent expression programs and consensus tissue groups

CTGs are constructed by first computing the empirical probability that a pair of tissues will be assigned to the same tissue group. The empirical co-grouping probabilities are then used as pair-wise similarity measures in a standard bottom-up agglomerative hierarchical clustering algorithm using complete linkage (e.g., as discussed in [58]). To be precise, let S denote the total number of samples, and $q_t^{(l)}$ the tissue group assignment for tissue t in sample l. The empirical co-grouping probability for tissues t and r is then:

$$\widehat{p}_{tr} = \sum_{l=1}^{S} \mathrm{I}(q_t^{(l)} = q_r^{(l)})/S$$

Here, $I(\cdot)$ is the indicator function.

Clustering is stopped using a pre-defined cut-off c_{tg} to produce the final CTGs. We used a cut-off of $c_{tg} = 0.90$ to produce strongly coherent groups. However, we note that the empirical co-grouping probabilities tend to be either very small or close to 1.0, rendering our results relatively insensitive to the choice of c_{tg}.

REPs consist of sets of tissues and genes that appear together with significant probability in expression programs across multiple samples. For each expression program in each sample, a set of *index tissues* is determined based on the extent of overlap of genes in the program and those expressed by the tissue (significance is determined using the hypergeometric distribution). To be precise, let $J^{(s)}$ be the number of expression programs used in sample s. Let $\eta_{tj}^{(s)}$ denote the number of genes expressed in tissue t and assigned to expression program j in sample s, i.e., $\eta_{tj}^{(s)} = |\{\omega(x_{ti}) : z_{ti}^{(s)} = j\}|$. We use the hypergeometric distribution to compute a p-value, $v_{tj}^{(s)}$, for each tissue and expression program pair:

$$v_{tj}^{(s)} = 1 - \text{HyperCDF}(\eta_{tj}^{(s)} - 1, G, \sum_{l=1}^{J^{(s)}} \eta_{tl}^{(s)}, \sum_{l=1}^{T} \eta_{lj}^{(s)})$$

Here, HyperCDF denotes the cumulative distribution function for the hypergeometric distribution. We use the p-values, $v_{tj}^{(s)}$, to compute the index tissues $V_j^{(s)}$ for expression program j in sample s, i.e., $V_j^{(s)} = \{t : v_{tj}^{(s)} < c_1\}$, i.e., the set of all tissues whose p-values for expression program j are below a threshold c_1 in sample s. We used a p-value threshold c_1 of 5%.

A hash table using the index tissues enables the algorithm to efficiently determine whether an expression program has already occurred in previous samples. If it has not, a new REP is instantiated; otherwise the expression program is merged into the appropriate REP. Statistics are tracked for each REP, including the number of samples it occurs in, its average weighting in the tissue's mixture over programs, and average expression levels of species specific genes and meta-genes in the program. To be precise, let S_j denote the number of samples in which REP j occurs. Then, the empirical mean expression level for gene g in REP j is defined as:

$$\widehat{e}_{gj} = \frac{\sum_{s=1}^{S} \sum_{t,i} \mathrm{I}(z_{ti}^{(s)} = j)}{|V_j| S_j} \quad \text{s.t. } t \in V_j^{(s)}, \omega(x_{ti}) = g$$

The empirical mean gene occurrence for gene g in recurrent expression program j is defined as:

$$\widehat{o}_{gj} = \frac{\sum_{s=1}^{S} \sum_{t} \mathrm{I}\left(\sum_{i} \mathrm{I}(z_{ti}^{(s)} = j) > 0\right)}{|V_j| S_j} \quad \text{s.t. } t \in V_j^{(s)}, \omega(x_{ti}) = g$$

The empirical mean tissue weighting for tissue t in recurrent expression program

j is defined as:

$$\widehat{w}_{tj} = \frac{\sum_{s=1}^{S} \eta_{tj}^{(s)}}{N_t S}$$

After all samples have been collected, several post-processing steps are then performed, including filtering out infrequently occurring REPs and genes, and merging of similar REPs. We filtered out REPs that occurred in fewer than 50% of samples, and filtered out genes with $\widehat{o}_{gj}$ scores less than 5%. The final merging step uses the same agglomerative procedure described for CTGs. In this case, the similarity measure is the fraction of genes shared by REPs. Only common index tissues are retained in merging two REPs. Merging is stopped when the similarity measure is less than a cut-off of 50%.

3.3.5 Expression data discretization

Expression data input into GeneProgram was first discretized using a mutual information-based greedy agglomerative merging algorithm, essentially as described in Hartemink *et al.* [84]. In brief, continuous expression levels are first uniformly discretized into a large number of levels. The algorithm then repeatedly finds the best two adjacent levels to merge by minimizing the reduction in the pair-wise mutual information between all expression vectors. The appropriate number of levels to stop at is determined by choosing the inflection point on the curve obtained by plotting the score against the number of levels. In this case, we obtained three levels.

For completeness, we describe the discretization algorithm here. We begin by initializing the algorithm with sets of expression levels for each tissue. We denote gene i in tissue t by g_{ti}, where there are T tissues. Let $r(g_{ti})$ denote the rank of gene i in tissue t based on the continuous expression value of the gene. To initialize the algorithm, we begin by assigning genes in each tissue t to an ordered set $\Lambda_t^{(0)}$ of N_L discrete expression levels that induce uniform bins on the gene rankings for the tissue. That is, $\Lambda_t^{(0)} = (L_{t1}^{(0)}, \ldots, L_{tN_L}^{(0)})$, where $g_{ti} \in L_{tl}^{(0)}$ iff $l - 1 < r(g_{ti})N_L/G_t \leq l$. Here, G_t is the number of genes in tissue t that are considered expressed (e.g., expression values greater than some threshold).

Each iteration consists of a set of trial merges, in which adjacent levels are merged and a score is computed. For iteration q and for each trial h, the adjacent levels h and $h+1$ are merged, forming a new set of levels with one less element, i.e., $(L_{t1}^{(q-1)}, \ldots, L_{th}^{(q-1)} \bigcup L_{t(h+1)}^{(q-1)}, L_{t(h+2)}^{(q-1)}, \ldots, L_{t(N_L-q)}^{(q-1)})$. Let $\boldsymbol{e}_t^{(qh)}$ denote the discrete vector of expression levels for tissue t for iteration q of the algorithm and trial merge h. That is, $e_{ti}^{(qh)} = l$ iff g_{ti} is in level l for trial merge h and g_{ti} is expressed in the tissue (otherwise, we set $e_{ti}^{(qh)} = 0$). The score for a trial merge h is the mutual information between all pairs of vectors of discretized expression data, i.e., $S_h^q = \sum_{t_1=1}^{T-1} \sum_{t_2>t_1} \mathrm{MI}(\boldsymbol{e}_{t_1}^{(qh)}, \boldsymbol{e}_{t_2}^{(qh)})$. At each iteration, the single merge operation that produces the highest score is retained. Note that because the algorithm is greedy, its run-time is $O(N_L^2 T^2)$.

3.4 Synthetic data experiments

We used a simple synthetic data example to explore the kinds of structures GeneProgram and several other well-known unsupervised learning algorithms could recover from noisy data. Our objective with these experiments was to use simulated data to illustrate the capabilities of the algorithms; whether or not particular structures are present in real data can only be answered empirically, and is addressed in Section 3.5.

3.4.1 Data simulation

In creating synthetic data, we sought to simulate important features of real microarray data profiling mammalian tissues. Thus, we assumed noisy data in which there were several distinct populations of related tissues using different sets of co-expressed genes.

We generated four gene sets used by 40 tissues divided equally among four tissue populations. Each gene set contained 40 genes with varying mRNA levels; gene sets three and four overlapped in 10 genes. The simulated underlying mRNA level m_{ij} for gene i in gene set j was generated as $m_{ij} \sim \text{round}(1000 * \text{Gamma}(3, 2))$.

Each tissue population k was associated with a mean vector $\boldsymbol{N}_k$ of the numbers of genes to be used from each gene set (see Table 3.2 for the mean vectors used to generate the simulated data). For a tissue t from population k, the number of genes to be used from gene set j was sampled from a Poisson distribution with parameter N_{kj}.

Genes were picked to be expressed from each set used by the tissue. Genes were picked without replacement such that the probability of picking gene i from gene set j for tissue t when l genes had already been picked was:

$$p_{ti}^{(l)} = \frac{\log m_{ij}}{\sum_{k \notin G_{tj}^{(l-1)}} \log m_{kj}}$$

Where genes were picked sequentially and $G_{tj}^{(l-1)}$ denotes the collection of the first $l-1$ genes picked from gene set j for tissue t. The probability is zero if the gene had already been picked. Finally, the observed expression value e_{it} for gene i in tissue t was generated as:

$$e_{it} = a_{it} I_{tij} m_{ij} + b_{it}$$

Here, I_{tij} is an indicator denoting whether gene i from gene set j was picked by tissue t, and a_{it} and b_{it} are multiplicative and additive noise respectively. Noise was generated with $a_{it} \sim \text{lognormal}(0, 0.1)$ and $b_{it} \sim \text{lognormal}(\log(200), 1)$. The mean and scale of noise were chosen to approximate Affymetrix microarray data (see [155]).

Figure 3-11 (part A) depicts the synthetic data. We note that our scheme for simulating data does not simply recapitulate the assumptions present in the GeneProgram model (e.g., it does not assume discrete and independent "units" of expression signal and it introduces microarray-like noise).

gene set no.	tissue pop. 1	tissue pop. 2	tissue pop. 3	tissue pop. 4
1	30	25	3	3
2	3	30	25	3
3	5	3	37	20
4	3	3	20	20

Table 3.2: Tissue population means for synthetic data. Each tissue population was associated with a mean vector of the numbers of genes to be used from each gene set. For a tissue from a population, the number of genes to be used from a gene set was sampled from a Poisson distribution using the population mean.

3.4.2 GeneProgram accurately recovered coherent gene sets from the noisy synthetic data that other algorithms could not

Hierarchical clustering is one of the most frequently used methods for clustering microarray gene expression data. Figure 3-11 (part B) shows the results of hierarchical clustering, using Pearson correlation as a similarity metric and the average linkage method [135], applied to sorting both rows (genes) and columns (tissues) of the synthetic data. As can be seen, hierarchical clustering did reasonably well at sorting tissues and genes independently, although it did not separate gene sets three and four correctly. But, this method's failure to consider genes and tissues simultaneously is known to break up coherent "overhanging" blocks of genes, making interpretation difficult [42, 209, 39]. This issue was demonstrated in this example by gene sets one and two that "overhang," thus causing gene set two to be broken up horizontally (blue rows in Figure 3-11 part A and part B).

The inability of hierarchical clustering to handle "overhanging" block structures in data was one of the motivations for the development of biclustering algorithms that take genes and tissues into account simultaneously [42, 209, 133]. To investigate the behavior of biclustering algorithms, we used Samba, an algorithm that has been shown previously to outperform other biclustering methods [209, 194]. Samba produced 23 biclusters from the synthetic data (not shown). This method tended to find small subsets of genes co-expressed in some tissues, but did not recover the four gene sets as coherent biclusters. Presumably, this is because Samba does not attempt to incorporate more global constraints on biclusters.

Singular Value Decomposition (SVD) is a matrix factorization method that can be used to approximate a matrix using a smaller number of factors or components. In the context of gene expression data, the method has previously been used to decompose data into "eigengenes" and "eigenexperiments," linear combinations of genes and experiments respectively [8, 9]. However, SVD often produces components that are difficult to interpret [39, 36, 111, 115]. As can be seen in Figure 3-11 (part C), components produced by SVD [135] do not clearly correspond to the distinct gene sets or tissue populations in the synthetic data. For instance, the first component is to some extent a composition of gene sets one, three and four, and subsequent components then add and subtract different combinations of the gene sets.

The development of non-negative matrix factorization (NMF) methods was in part driven by the aforementioned problems with SVD. NMF algorithms decompose a matrix into the product of non-negative matrices [115]. These algorithms generally produce more interpretable factors than does SVD, and have been successfully applied to various problems including gene expression analysis [39, 36, 111]. Figure 3-11 (part D) shows the application of an NMF-based algorithm to the synthetic data. We used a publicly available implementation [36], which searches for an optimal number of factors using a cophenetic clustering coefficient metric, and in this case found three factors to be optimal. As can be seen in the figure, NMF did fairly well at recovering genes sets one and two, although there was some overlap between the sets. However, gene sets three and four were indistinguishable.

Figure 3-11 (part E) demonstrates the application of a simplified version of GeneProgram in which tissue groups were not modeled (all tissues were attached to the root of the hierarchy). As can be seen, this version of the algorithm accurately recovered gene sets one and two. However, as with NMF, gene sets three and four completely overlapped.

Figure 3-11 (part F) shows the application of GeneProgram with full automatic learning of tissue groups enabled. As can be seen, the algorithm accurately recovered all four gene sets. By leveraging hierarchical structure in the data, the algorithm had additional information (the pattern of expression program use by related tissues), which presumably allowed it to correctly recover all the synthetic gene sets—something the other methods were not capable of.

3.5 GeneProgram discovered biologically relevant tissue groups and expression programs in a large compendium of human and mouse body-wide gene expression data

Our objective was to apply GeneProgram to a large compendium of mammalian gene expression data, both to compare our method's performance against that of other algorithms, as well as to explore the biological relevance of discovered tissue groups and expression programs. In this regard, we used the Novartis Gene Atlas v2 [205], consisting of genome-wide expression measurements for 79 human and 61 mouse tissues. This dataset was chosen because it contains a large set of relatively high-quality expression experiments, with body-wide samples representative of normal tissues measured on similar microarray platforms. Further, the data is from two species, potentially allowing for the discovery of higher quality cross-species gene expression programs.

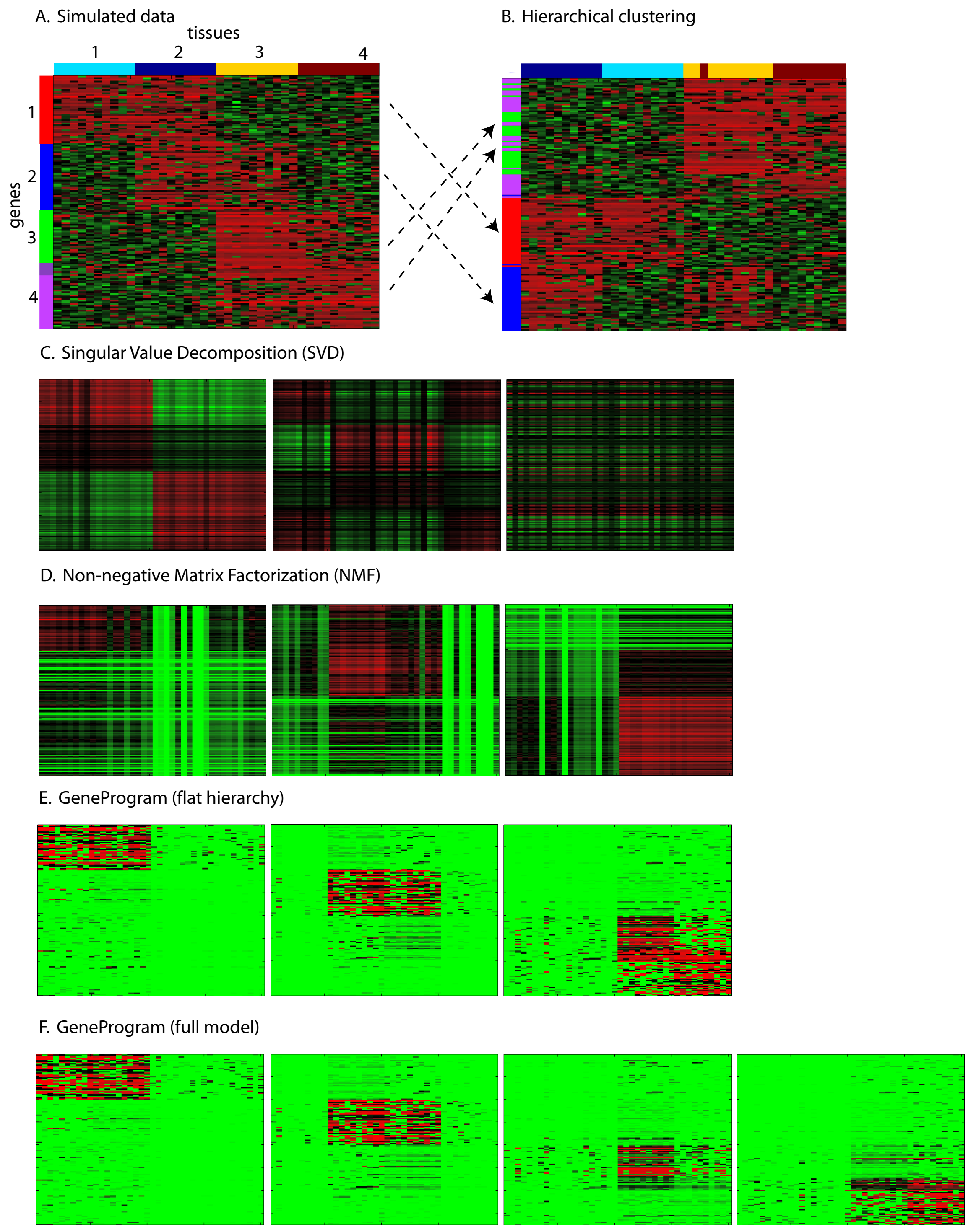

A. Simulated data
tissues
1
2
3
4
genes
1
2
3
4
B. Hierarchical clustering
C. Singular Value Decomposition (SVD)
D. Non-negative Matrix Factorization (NMF)
E. GeneProgram (flat hierarchy)
F. GeneProgram (full model)

Figure 3-11: Synthetic data experiments demonstrated GeneProgram's and other algorithms' abilities to recover gene sets from noisy data. *(Part A)* Noisy synthetic data, containing 150 genes (rows) and 40 tissues (columns), with four gene sets and four tissue sample populations. Vertical numbers and colored bars designate gene sets; corresponding horizontal elements designate tissue sample populations. Each gene set contained 40 genes with varying mRNA levels; programs 3 and 4 overlapped in 10 genes. See the text for complete details. *(Part B)* Hierarchical clustering was applied to sorting both rows (genes) and columns (tissues) of the data. Arrows from part A indicate resorting of gene sets. Note that hierarchical clustering did not separate gene sets 3 and 4 correctly, and broke up gene set 2 horizontally. *(Part C)* Singular Value Decomposition (SVD) factors did not clearly correspond to the distinct gene sets or tissue populations in the synthetic data; the first three factors are shown. *(Part D)* A non-negative matrix factorization (NMF) implementation, which searches for the optimal number of factors, was applied to the data. In this case, 3 factors were optimal. As can be seen, the method performed fairly well in recovering genes sets 1 and 2, although there was some overlap between the sets. However, gene sets 3 and 4 were not recovered as separate sets. (*Part E)* A simplified version of GeneProgram using a flat hierarchy (automatic tissue grouping disabled) accurately recovered gene sets 1 and 2, but failed to separate sets 3 and 4. *(Part F)* The full GeneProgram implementation using automatic tissue grouping correctly recovered all 4 gene sets.

3.5.1 Data set pre-processing

All arrays in the data set were first processed using the GC content-adjusted robust multi-array algorithm (GC-RMA) [221]. To correct for probe specific intensity differences, the intensity of each probe was normalized by dividing by its geometric mean in the 31 matched tissues. For genes represented by more than one probe, we used the maximum of the normalized intensities. A gene was considered expressed if its normalized level was greater than 2.0 and was called present in one or more replicates of the MAS5 Absent/Present calls [92].

We identified pairs of related genes using Homologene (build 47) [219], which attempts to find homologous gene sets among the completely sequenced eukaryotic genomes by using a taxonomic tree, conserved gene order, and measures of sequence similarity. Of the approximately 16,000 homologous human-mouse pairs identified by Homologene, 9851 gene pairs appear in the Gene Atlas v2.

3.5.2 GeneProgram discovered 19 consensus tissue groups and 100 recurrent expression programs

Figure 3-12 depicts all 19 tissue groups. Supplemental online Table 1 [74] provides a summary and supplemental online Table 2 [75] contains the full data for all 100 expression programs. The tissue groups were of various sizes, ranging from 1–38 tissues (median of 4). Expression program sizes ranged from 12–292 meta-genes (median of 72) and 1–38 tissues (median of 4). A large fraction (67%) of meta-genes appeared in at least one expression program and 31% were shared by several expression programs. Forty-two percent of tissue groups and 33% of expression programs contained at least one tissue from each species. It is important to realize that the number of cross-species tissue groups and expression programs was limited by the data set: only 62 out of the 140 tissue samples could be directly paired between species and some key tissues with distinct functions, such as the stomach and eye, were represented in only one species.

3.5.3 GeneProgram automatically assigned tissues to biologically relevant groups

To provide a quantitative assessment of the biological relevance of sets of tissues, we manually classified tissues into 10 high-level, physiologically based categories and then calculated an enrichment score for each discovered tissue group using the hypergeometric distribution. See the supplemental online material for [76] for the complete manually derived tissue categories. To correct for multiple hypothesis tests, we used the procedure of Benjamini and Hochberg [24] with a false-discovery rate cut-off of 0.05.

Seventy-nine percent of tissue groups had significant enrichment scores, and in all such cases, the score was significant for only a single category (see Figure 3-12). For instance, tissue group "L," which was significantly enriched only for the "hematological/immune" category, consisted exclusively of human immune cells such as natural killer cells, and CD4+ and CD8+ T-cells. As another example, tissue group "B," significantly enriched only for the "neural" category, consisted exclusively of neural tissues from both species. We note that GeneProgram discovered these groups in a wholly unsupervised manner, and that many of the groups clearly represent a more refined picture of the data than the 10 broad categories we had manually compiled.

3.5.4 GeneProgram outperformed biclustering algorithms in the discovery of biologically relevant gene sets

Because expression programs characterize both genes and tissues, we used both Gene Ontology (GO) categories [11] and the 10 manually derived tissue categories to assess GeneProgram's ability to recover biologically relevant gene sets and to compare this performance to that of two biclustering algorithms, Samba [209, 194] and a non-negative matrix factorization (NMF) implementation [36]. We chose these two algorithms for comparison because they are popular in the gene expression analysis community, they have previously outperformed other biclustering algorithms, and available implementations are capable of handling large data sets.

We mapped genes to GO annotations using RefSeq identifiers from the May 2004 (hg17) and August 2005 (mm7) assemblies of human and mouse genomes [11, 219]. For calculating enrichments, we used both mouse and human GO annotations from the biological process categories with between 5 and 200 genes. Enrichment score

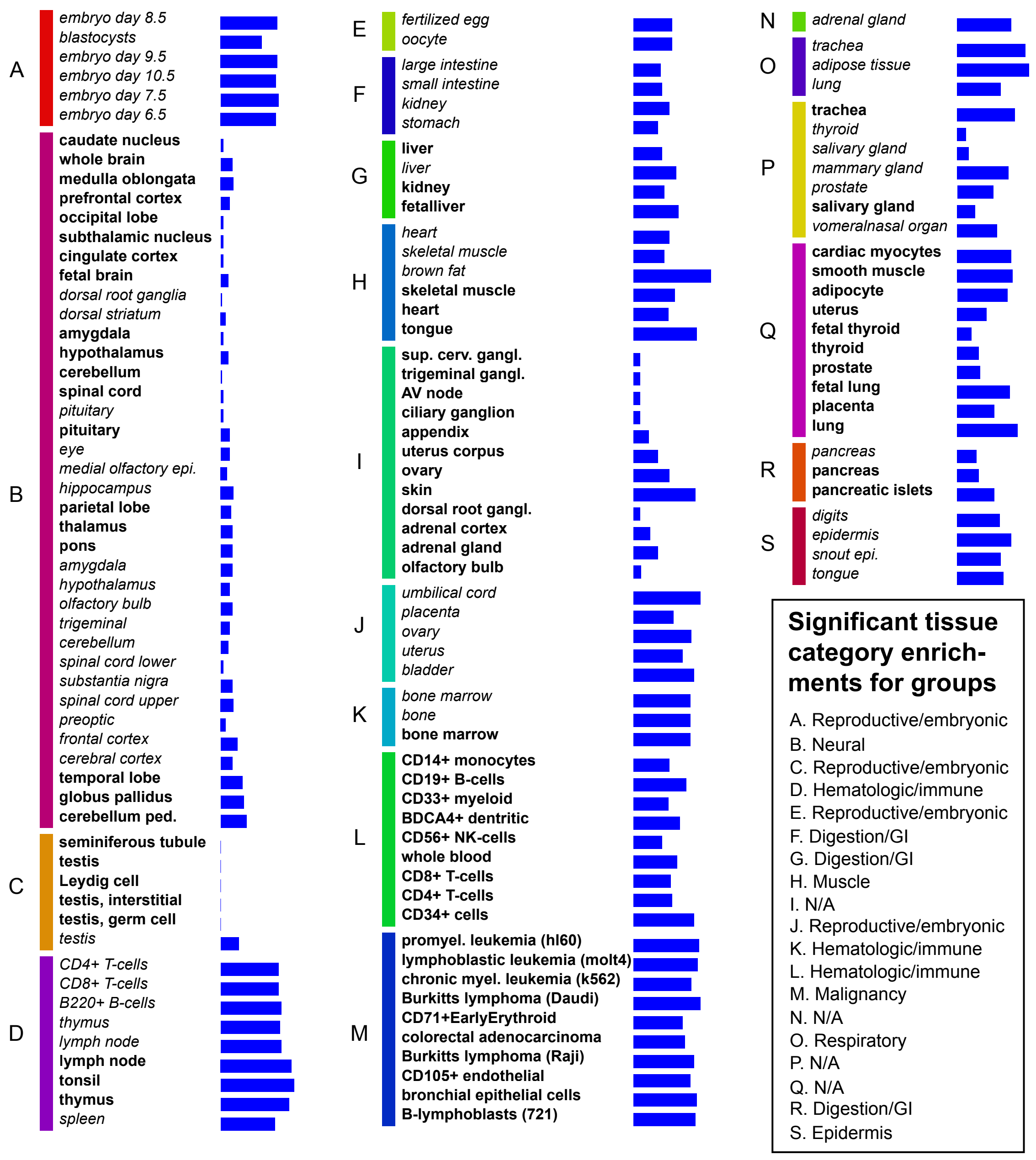

A
embryo day 8.5
blastocysts
embryo day 9.5
embryo day 10.5
embryo day 7.5
embryo day 6.5
B
caudate nucleus
whole brain
medulla oblongata
prefrontal cortex
occipital lobe
subthalamic nucleus
cingulate cortex
fetal brain
dorsal root ganglia
dorsal striatum
amygdala
hypothalamus
cerebellum
spinal cord
pituitary
pituitary
eye
medial olfactory epi.
hippocampus
parietal lobe
thalamus
pons
amygdala
hypothalamus
olfactory bulb
trigeminal
cerebellum
spinal cord lower
substantia nigra
spinal cord upper
preoptic
frontal cortex
cerebral cortex
temporal lobe
globus pallidus
cerebellum ped.
C
seminiferous tubule
testis
Leydig cell
testis, interstitial
testis, germ cell
testis
D
CD4+ T-cells
CD8+ T-cells
B220+ B-cells
thymus
lymph node
lymph node
tonsil
thymus
spleen
E
fertilized egg
oocyte
F
large intestine
small intestine
kidney
stomach
G
liver
liver
kidney
fetalliver
H
heart
skeletal muscle
brown fat
skeletal muscle
heart
tongue
I
sup. cerv. gangl.
trigeminal gangl.
AV node
ciliary ganglion
appendix
uterus corpus
ovary
skin
dorsal root gangl.
adrenal cortex
adrenal gland
olfactory bulb
J
umbilical cord
placenta
ovary
uterus
bladder
K
bone marrow
bone
bone marrow
L
CD14+ monocytes
CD19+ B-cells
CD33+ myeloid
BDCA4+ dentritic
CD56+ NK-cells
whole blood
CD8+ T-cells
CD4+ T-cells
CD34+ cells
M
promyel. leukemia (hl60)
lymphoblastic leukemia (molt4)
chronic myel. leukemia (k562)
Burkitts lymphoma (Daudi)
CD71+EarlyErythroid
colorectal adenocarcinoma
Burkitts lymphoma (Raji)
CD105+ endothelial
bronchial epithelial cells
B-lymphoblasts (721)
N
adrenal gland
O
trachea
adipose tissue
lung
P
trachea
thyroid
salivary gland
mammary gland
prostate
salivary gland
vomeralnasal organ
Q
cardiac myocytes
smooth muscle
adipocyte
uterus
fetal thyroid
thyroid
prostate
fetal lung
placenta
lung
R
pancreas
pancreas
pancreatic islets
S
digits
epidermis
snout epi.
tongue
Significant tissue category enrich-ments for groups
A. Reproductive/embryonic
B. Neural
C. Reproductive/embryonic
D. Hematologic/immune
E. Reproductive/embryonic
F. Digestion/GI
G. Digestion/GI
H. Muscle
I. N/A
J. Reproductive/embryonic
K. Hematologic/immune
L. Hematologic/immune
M. Malignancy
N. N/A
O. Respiratory
P. N/A
Q. N/A
R. Digestion/GI
S. Epidermis

Figure 3-12: GeneProgram discovered 19 consensus tissue groups using gene expression data from 140 human and mouse tissue samples. The algorithm identified these groups in a wholly unsupervised manner. In each tissue group (denoted A-S), human tissues are designated with bold type and mouse tissues with italic type. Tissues were classified manually into 10 broad categories based on physiological function, and it was found that 79% of tissue groups were significantly enriched for at least one category (boxed legend, lower right corner). To the right of each tissue is a blue vertical bar depicting its weighted average of generality scores for expression programs, which provides a measure of the extent to which the tissue uses programs shared by diverse tissue types (see the text for details).

calculation and correction for multiple hypothesis tests were the same as described in Section 3.5.3.

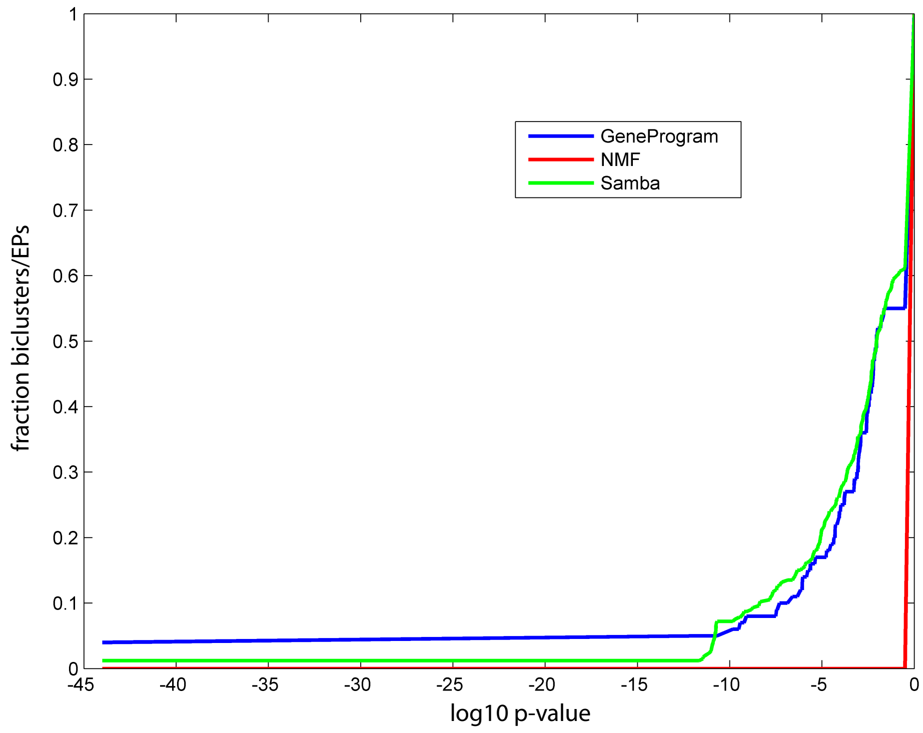
As Table 3.3 shows, GeneProgram clearly outperformed the other two algorithms in the tissue dimension (60% of expression programs significantly enriched for tissue categories, versus 10% for Samba and 20% for NMF). GeneProgram outperformed NMF and had equivalent performance to Samba in the gene dimension (61% of expression programs significantly enriched for GO categories, versus 62% for Samba and 27% for NMF).

Figure 3-13 shows the same trends using correspondence plots, which are sensitive, graphical methods for comparing biclustering algorithms [210]. These plots depict log p-values on the horizontal axis and the fraction of biclusters with p-values below a given value on the vertical axis. Depicted p-values are from the most abundant class for each bicluster (i.e., that with the largest number of genes or tissue in the overlap) and calculated using the hypergeometric distribution. Note that biclusters with large p-values are not significantly enriched for any class, and may represent noise.

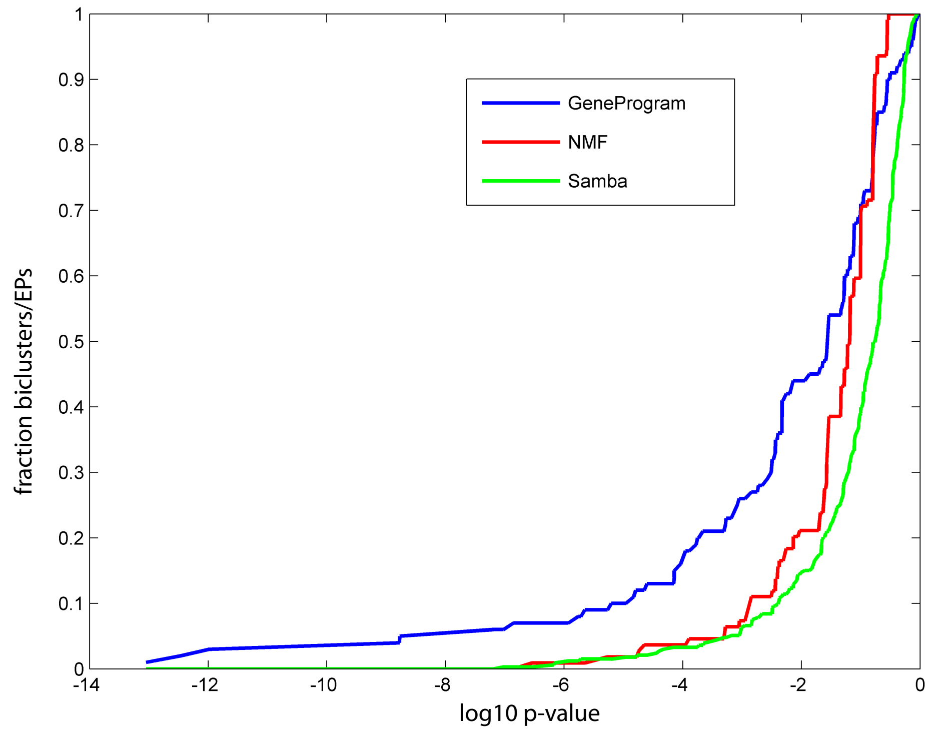
These results suggest several performance trends related to features of the different algorithms. As noted in the section on synthetic data experiments, Samba was successful at finding relatively small sets of genes that are co-expressed in subsets of tissues, but had difficulty uncovering larger structures in data. Presumably, our algorithm's clear dominance of both Samba and NMF in the tissue dimension was partly attributable to GeneProgram's hierarchical model. Both of the other algorithms lack such a model, so the assignment of tissues to biclusters was not guided by global relationships among tissues.

We note also that the algorithms differed substantially in runtimes: Samba was fastest (approximately 3 hours), GeneProgram the next fastest (approximately 3 days), and NMF the slowest (approximately 6 days), with all software running on a 3.2 GHz Intel Xenon CPU. Although these runtime differences may be attributable in part to implementation details, it is worth noting that GeneProgram, a fully Bayesian model using MCMC sampling for inference, ran faster than the NMF algorithm, which uses a more "traditional" objective maximization algorithm to search for the appropriate number of biclusters.

A. Gene dimension



B. Tissue dimension



algorithm	**gene dimension** (GO category enrichment)	**tissue dimension** (manually derived category enrichment)
GeneProgram	61%	60%
Samba	62%	10%
NMF	27%	20%

Table 3.3: Comparison of GeneProgram to biclustering algorithms for recovery of biologically interpretable gene sets. GeneProgram's ability to recover biologically interpretable gene sets from a large compendium of mammalian tissue gene expression data was compared against that of two popular biclustering algorithms, Samba and a non-negative matrix factorization (NMF) implementation. GeneProgram dominated the other two algorithms in the tissue dimension; it outperformed NMF and had equivalent performance to Samba in the gene dimension. Biological interpretability of gene sets was assessed using Gene Ontology (GO) categories in the gene dimension, and manually constructed categories in the tissue dimension. Each cell in the table shows the percentage of sets significantly enriched for at least one category in a given dimension (p-value < 0.05, corrected for multiple hypothesis tests).

Figure 3-13: Correspondence plots comparing GeneProgram to biclustering algorithms. These plots compare GeneProgram's ability to recover biologically interpretable gene sets from a large compendium of mammalian tissue gene expression data against that of two popular biclustering algorithms, Samba and a non-negative matrix factorization (NMF) implementation. GeneProgram clearly dominated the other two algorithms in the tissue dimension; it outperformed NMF and had equivalent performance to Samba in the gene dimension. Biological interpretability of gene sets was assessed using Gene Ontology (GO) categories in the gene dimension, and manually derived high-level, physiologically based categories in the tissue dimension. The plots depict p-values (enrichment scores) on the horizontal axis and the fraction of biclusters with p-values below a given value on the vertical axis (the p-value for the most abundant class was used).

3.5.5 GeneProgram cross-species expression programs outperformed single-species programs in terms of biological relevance in both the gene and tissue dimensions

Seventy-nine percent of cross-species programs were significantly enriched for GO categories versus 52% of single-species programs, and 82% of cross-species programs were significantly enriched for the manually derived tissue categories versus 51% of single-species programs. These results suggest that combining data from both species was valuable for discovery of biologically relevant expression programs. However, this conclusion must be interpreted cautiously for the gene dimension, because GO annotations may be biased toward extensively studied genes that are expressed in both species.

It is also relevant to ask whether single species expression programs represent biologically important differences in gene expression between mice and humans. Unfortunately, substantial differences in how samples from the two species were obtained, prepared and experimentally analyzed were confounding factors. Nonetheless, some single-species expression programs appeared to reflect real biological differences between mice and humans. For instance, expression program 78 contained only mouse tissues, including general and snout epidermis. Interestingly, the program contained many keratin genes, which are components of hair fibers, and the Cochlin gene, which has been detected in spindle-shaped cells located along nerve fibers that innervate hair cells [178]; such structures are considerably more abundant in fur-covered mouse skin than in human skin.

3.5.6 Automatic inference of tissue groups resulted in significant improvements in model performance

We used cross-validation to analyze the importance of automatic tissue group inference in our model. We tested the full GeneProgram model versus a simplified version in which there were no groups and all tissues are attached directly to the root of the hierarchy.

We used 10-fold cross-validation on the 140 tissues; the order of the tissues was first randomly permuted so that there would be no bias toward selecting training sets from only a single species.

The *perplexity* was then calculated for each held-out tissue; perplexity is a measure commonly used for evaluating statistical language and information retrieval models [183]. In this context, it is inversely related to the predicted model likelihood of the expression data in the held-out tissue given the training data. Thus, smaller perplexity values indicate a better model fit.

The model was burned in with 100,000 iterations as described in Section 3.3.3. After burn-in, the model posterior was sampled ten times (we allowed 100 iterations between samples to reduce dependencies). For each of the ten samples, the held-out tissue t was then added back, the model was burned in for 10,000 iterations and 500 samples were generated to compute:

$$L_t^{(s)} = \sum_{i=1}^{N_t} \log \mathrm{p}(\mathsf{x}_{ti} \mid \boldsymbol{D})$$

Here, t denotes the tissue, s the sample, and $\boldsymbol{D}$ all the training data. An estimate of the tissue log-likelihood $\widehat{L}_t$ was then computed from the 5,000 $L_t^{(s)}$ samples using the harmonic mean method described by Kass and Raftery [110]. The tissue perplexity was then estimated as:

$$\text{perplexity} = 2^{\widehat{L}_t / N_t}$$

The full GeneProgram model consistently yielded reduced perplexity values compared to the simplified model, with a median perplexity reduction of 24%. Figure 3-14 shows a graph of these results. Perplexity reductions of 10% or greater have typically been considered significant [183]. Thus, we conclude that allowing the model to infer tissue groups automatically significantly improves performance.

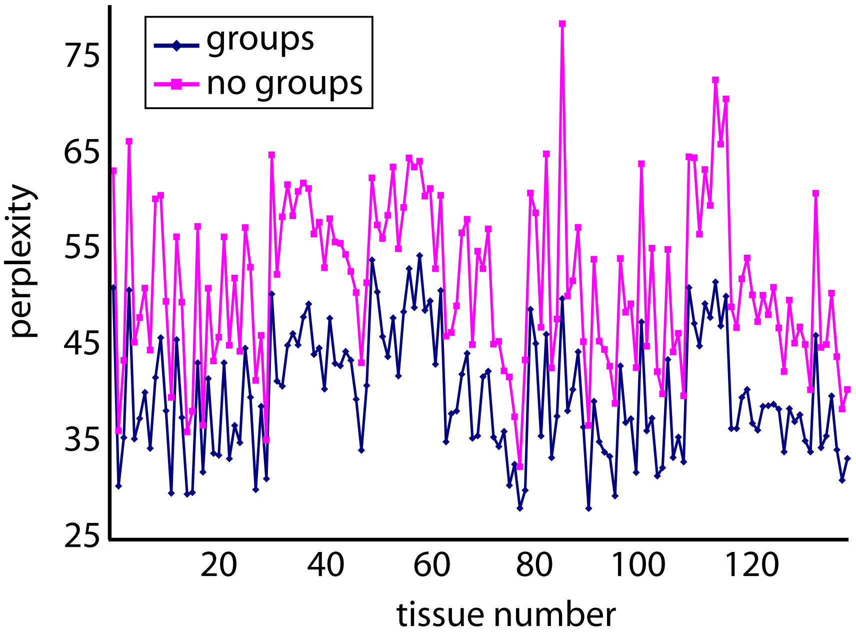


Figure 3-14: Automatic inference of tissue groups improves cross-validation performance of the GeneProgram model. We used 10-fold cross-validation to test the full GeneProgram model ("groups") versus a simplified version in which there are no groups and all tissues are attached directly to the root of the hierarchy ("no groups"). Each data point represents the calculated perplexity value for each held-out tissue (1-79 = human tissues, 80-140 = mouse tissues). Lower perplexity values indicate better model performance. The median reduction in perplexity for the full versus the simplified model was 24%.

3.5.7 The generality score quantified the functional specificity of expression programs and organized programs into a comprehensive body-wide map

We developed a score for assessing the functional generality of expression programs, and demonstrated its utility for automatically characterizing the spectrum of discovered programs—from gene sets involved in general physiologic processes to highly tissue-specific ones.

The *generality* score is the entropy of the normalized distribution of usage of an expression program by all tissues in each tissue group. Because the distribution employed for calculating the score is normalized, tissue groups that only use an expression program a relatively small amount will have little effect on the score. Thus, a high generality score indicates that an expression program is used fairly evenly across many tissue groups; a low score indicates the program is used by tissues from a small number of groups.

To be precise, let $\overline{q}_t$ denote the consensus tissue group (CTG) assignment for tissue t. We compute the usage for CTG k of recurrent expression program (REP) j as:

$$h_{kj} = \sum_{t=1}^{T} \widehat{w}_{tj} \mathrm{I}(\overline{q}_t = k)$$

The normalized usage is then computed as:

$$\widehat{h}_{kj} = \frac{h_{kj}}{\sum_{l=1}^{K} h_{lj}}$$

Here, K is the total number of CTGs. The generality score for REP j is then computed as:

$$\mathrm{GS}_j = -\sum_{k=1}^{K} \widehat{h}_{kj} \log \widehat{h}_{kj}$$

We note that the generality score requires a global organization of tissues into groups, rather than just the local associations of subsets of tissues with individual gene sets provided by biclustering algorithms. Because there is uncertainty in the number of tissue groups, GeneProgram's Dirichlet Process-based model provides a natural framework for computing the generality score.

Evaluation of the weighted average of generality scores across all expression programs for each tissue uncovered several trends relating to tissue function and anatomic location

Figure 3-12 depicts the weighted scores for all tissues. As is evident from this figure, some tissues types, including neural, testicular and thyroid samples, had very low average generality scores, presumably reflecting the highly specialized functions of these tissues. In contrast, a number of other tissue types, including embryologic, hematologic progenitors, immune, malignant, epithelial and adipose samples, had very high average generality scores. In the case of embryologic and relatively undifferentiated malignant tissues, high scores presumably reflected the activation of large numbers of expression programs shared with many other types of tissues. The other high-scoring tissues mentioned also shared programs with many types of tissues, but this sharing may be attributed to both common biological functions as well as the fact that cells from these high-scoring tissues are found in many organs throughout the body.

We note that it is likely that some tissues had artificially high average generality scores due to sample contamination from anatomically nearby tissues. For instance, expression program 24, a program associated with muscle function, was used by fetal thyroid (3%), prostate (8%), lower spinal cord (4%), bone marrow (5%), and brown fat (12%). Each of these tissues is underneath substantial amounts of muscle, making contamination likely [142]. As another example, expression program 83 contained many genes involved in pancreatic function. However, this program was also used by mouse spleen (18%) and stomach (4%). Because the pancreas is anatomically proximal to both the stomach and spleen and can leave pancreatic tissue surrounding the duodenum as a result of its migration during development [161], contamination of these tissues seems likely.

Generality scores classified the functional specificity of individual expression programs

Figure 3-15 displays a histogram of generality scores for all expression programs (EPs) with non-zero scores. Based on the generality score, we divided expression programs into three broad categories: 1) general body-wide physiology, 2) specialized organ physiology, and 3) tissue specific. Below we provide illustrative examples from each category.

General body-wide physiology expression programs. EPs with high generality scores were involved in common physiological functions of cells present in a variety of tissues throughout the body. For instance, EP 13 (generality = 2.50, 25 tissues) contained many genes critical for DNA replication and EP 33 (generality = 2.34, 28 tissues) contained a striking number of genes involved with RNA processing, including numerous nuclear ribonucleoprotein components [119]. Interestingly, both EPs were used by many of the same tissues containing rapidly dividing cells, including embryologic, immune, and malignant tissues. Two additional examples include, EP 39 (generality = 2.88, 13 tissues), significantly enriched for genes involved in epithelial function, such as keratins and collagens; and, EP 24 (generality = 1.88, 15 tissues), significantly enriched for genes involved in general muscle function, including several known to be expressed in both cardiac and skeletal muscle such as alpha-actin-1 [81], myoglobin [70], and phosphoglycerate mutase isozyme M [57]. Interestingly, the tongue used both EPs 39 and 24 to a considerable extent, reflecting its mixed muscular and epithelial physiological functions.

Specialized organ physiology expression programs. EPs with intermediate generality scores were involved in specialized functions of a few closely related—but not necessarily anatomically proximate—tissues. For instance, EP 15 (generality = 1.44, 6 tissues) was significantly enriched for genes involved in erythropoiesis and was used primarily by adult bone marrow from both species and human fetal liver. Interestingly, the fetal liver is known to be critical for erythropoiesis during embryonic development, after which bone marrow becomes the predominant organ involved in this process [158]. Another example in this category includes EP 73 (generality = 0.93, 6 tissues), which was used by the kidney and liver in both species, and was enriched for genes involved in oxidative metabolism and gluconeogenesis. A final interesting example of this type is EP 88 (generality = 0.84, 3 tissues), which was used by the pituitary in both species and to a smaller extent by human pancreatic islets (5%). This EP contained a number of specific genes involved with pituitary function such as PIT1 [157] and the prolactin precursor [45]. A literature search revealed that several of the genes contained in this EP are known to be shared between the pituitary and islets, including prohormone convertase I [226] and proopiomelanocortin preproprotein [34, 95]. However, many of the genes in EP 88 have not previously been characterized as shared between the two endocrine organs, and thus may constitute interesting future candidates for experimental biology work.

Tissue specific expression programs. Finally, EPs with very low generality scores were used by essentially a single type of tissue, and represented very specialized aspects of organ functions. For instance, EP 19 (generality = 0.0, 6 tissues) was used exclusively by testicular tissues in both species, and was significantly enriched for genes involved in spermatogenesis. Two additional examples clearly illustrate GeneProgram's ability to automatically allocate tissues' gene expression to both general and specific programs. EP 43 (generality = 0) was used exclusively by the eye and was highly enriched for lens and retina specific genes. The eye also used EP 39, the general epithelial program described above, reflecting its more prosaic components. EP 58 (generality = 0) was exclusively used by the heart in both species, and contained cardiac specific genes such as atrial natriuretic peptide [52] and cardiac troponin T [192]. The heart also used the general muscle topic, EP 24, described above. Finally, EP 53 (generality = 0.26, 38 tissues), which was significantly enriched for genes involved in neurotransmission, illustrates that the generality score can be low despite usage of a program by a large number of tissues. Neural tissues were very abundant in the data set (31% of all tissues); because GeneProgram collapsed these tissues into a small number of groups, the generality score for EP 53 accurately reflected the biological homogeneity of the exclusively neural tissues using the expression program.

3.6 Conclusion and discussion

We presented a new computational methodology, GeneProgram, specifically designed for analyzing large compendia of mammalian expression data. Through synthetic data experiments, we showed that GeneProgram was able to correctly recover gene sets that other popular analysis methods could not. We then applied our method to a large compendium of human and mouse body-wide gene expression data from representative normal tissue samples, and demonstrated that GeneProgram outperformed other methods in the discovery of biologically interpretable gene sets. We further showed that allowing the GeneProgram model to infer tissue groups automatically

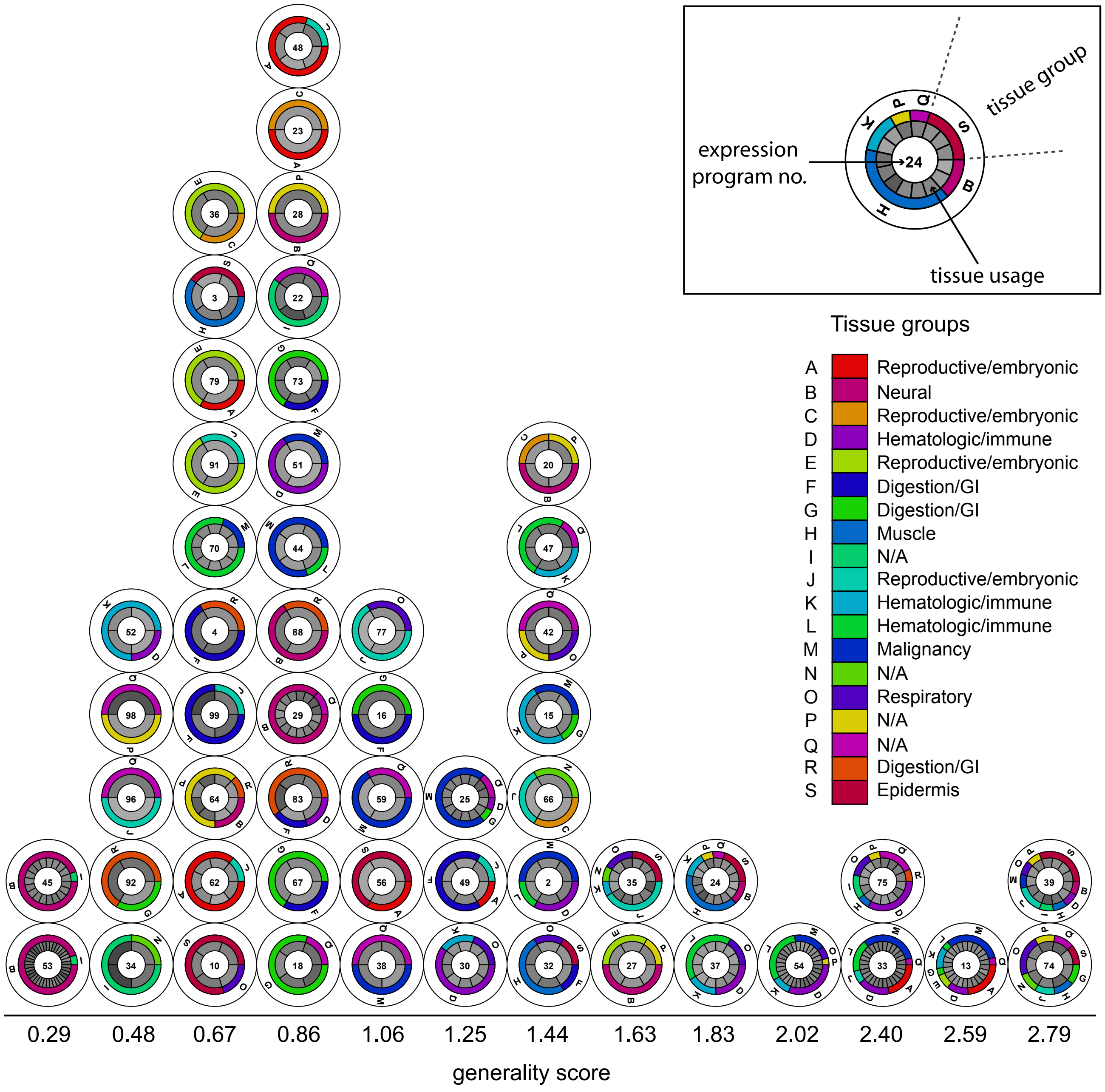

expression program no.
tissue group
tissue usage
24
Tissue groups
A Reproductive/embryonic
B Neural
C Reproductive/embryonic
D Hematologic/immune
E Reproductive/embryonic
F Digestion/GI
G Digestion/GI
H Muscle
I N/A
J Reproductive/embryonic
K Hematologic/immune
L Hematologic/immune
M Malignancy
N N/A
O Respiratory
P N/A
Q N/A
R Digestion/GI
S Epidermis
0.29
0.48
0.67
0.86
1.06
1.25
1.44
1.63
1.83
2.02
2.40
2.59
2.79
generality score

Figure 3-15: The generality score organized expression programs discovered by GeneProgram into a comprehensive body-wide map. A histogram using the generality score summarizes the functional specificity of the expression programs (EPs) discovered by GeneProgram in a large compendium of human and mouse gene expression data. The horizontal axis displays bins of generality scores. A high generality score indicates that an expression program is used fairly evenly across many tissue groups; a low score indicates the program is used by tissues from a small number of groups. Only EPs with non-zero scores are shown. EPs are depicted as numbered circles stacked within score bins. Two rings around each EP provide additional information. The innermost ring shows individual tissue usage percentages as shaded wedges (darker shading = higher usage). The outer ring depicts tissue groups, with arc sizes proportional to the number of tissues in the group using the EP. The boxed example shows EP 24 (generality = 1.88), which is used by 15 tissues from 6 groups. The legend below the boxed example depicts the broad physiological category that each tissue group was significantly enriched for.

significantly improved performance. Using the data compendium, GeneProgram discovered 19 tissue groups and 100 expression programs active in mammalian tissues. We introduced an expression program generality score that exploits the tissue groupings automatically learned by GeneProgram, and showed that this score characterizes the functional spectrum of discovered expression programs.

GeneProgram encodes certain assumptions that differ from some previous methods for analyzing expression data and so merit further discussion. First, we model expression data in a semi-quantitative fashion, assuming that discrete levels of mRNA correspond to biologically interpretable expression differences. We believe this is appropriate because popular array technologies can only reliably measure semi-quantitative, relative changes in expression; many relevant consequences of gene expression are threshold phenomena [94, 127, 224]; and it is difficult to assign a clear biological interpretation to a full spectrum of continuous expression levels. Second, GeneProgram assumes that discrete "units" of mRNA are independently allocated to expression programs, which captures the phenomena that mRNA transcribed from the same gene can be translated into proteins that may participate in different biological processes throughout a cell or tissue. Independence of mRNA units is an unrealistic assumption, but this approximation, which is important for efficient inference, has worked well in practice for many other applications of topic models [64, 79, 31]. Finally, although GeneProgram does not directly model down-regulation of genes, it does capture this phenomenon implicitly in that a tissue's non-use of an expression program provides critical information for the algorithm. However, this approach does not take into account the magnitude of a gene's down-regulation or distinguish down-regulation from a lack of significant change in a gene's expression. As shown in Chapter 4, GeneProgram can be usefully extended to take such information into account for application to datasets consisting of time-series or samples and controls, such as two-color microarray data.

Our method produced a comprehensive, body-wide map of expression programs active in mammalian physiology with several distinguishing features. First, by simultaneously using information across 140 tissue samples, GeneProgram was able to finely dissect the data, automatically splitting mRNA expressed in tissues among both general and specific programs. Second, because our model explicitly operates on probabilistically ranked gene sets throughout the entire inference process, rather than finding individual differentially expressed genes or merging genes into sets in pre-processing steps, our results are more robust to noise. Third, the fact that expression programs provide probabilistically ranked sets of genes also provides a logical means for prioritizing directed biological experiments. Fourth, because our model is fully Bayesian, providing a global penalty for model complexity including for the number of tissue groups and expression programs, the generated map represents a mathematically principled compression of gene expression information throughout the entire organism. Finally, although such a large, comprehensive map is inherently complicated, we believe that GeneProgram's automatic grouping of tissues and the associated expression program generality score aid greatly in its interpretation.

We believe that the features of the discovered map discussed above will make it particularly useful for guiding future biological experiments. Tissue-specific expression programs can provide candidate genes for diagnostic markers or drug targets that exhibit minimal "cross-talk" with unintended organs. General expression programs may be useful for identifying genes important in regulating and maintaining general physiological responses, which may go awry in disease states such as sepsis and malignancy. Both general and tissue-specific discovered programs contained many functionally unannotated genes, and in some cases the programs were shared among unexpected sets of tissues. Additionally, some such unannotated genes appear in cross-species expression programs, making them particularly attractive candidates for further biological characterization.

The map's utility can be further enhanced by adding new data as it becomes available, particularly body-wide tissue samples profiling gene expression in additional species. Further, our method is general, making it suitable for analyzing any large expression data compendium, including those relating to developmental or disease processes. Our framework is also flexible, and could accommodate other genome-wide sources of biological data in future work, such as DNA-protein binding or DNA sequence motif information. GeneProgram's ability to discover tissue groups and expression programs *de novo* using a principled probabilistic method, as well as its use of data in a semi-quantitative manner, makes it especially valuable for novel "meta-analysis" applications involving large data sets of unknown complexity in which direct fully quantitative comparisons are difficult.

CHAPTER 4

GeneProgram++

"What distinguishes a mathematical model from, say, a poem, a song, a portrait or any other kind of 'model,' is that the mathematical model is an image or picture of reality painted with logical symbols instead of with words, sounds or watercolors."

—*John Casti, Reality Rules*

IN MANY microarray gene expression experiments, we are interested in genes' behavior relative to some baseline condition. For instance, we may be interested in the extent of induction or repression of gene expression after cells are exposed to environmental stresses [71], infected with microorganisms [202, 149, 150, 107, 33, 91, 160, 80], or observed throughout development [220]. The simplest such studies may consider only a few experiment-control pairs. However, in more complex studies, researchers may seek to explore complex patterns of change, such as temporal dynamics.

In analyzing patterns of gene expression change, we would like to discover sets of genes that behave coherently. In Chapter 3, we presented the GeneProgram algorithm, and showed that it has a number of advantages over previous methods in the discovery of biologically relevant sets of genes from large compendia of data. However, a limitation of the GeneProgram algorithm is that it does not explicitly model patterns of gene expression change.

In this chapter, we present GeneProgram++, an extension of our original algorithm that explicitly models general patterns of expression changes, including not only induction or repression, but also temporal dynamics. Patterns of expression change are modeled using the novel concept of program *usage modifiers*. A usage modifier is a variable that is specific to a tissue-expression program pair and describes how a tissue uses the program. For instance, usage modifiers can specify the temporal phase and direction (induction or repression) of expression. Thus, the genes used by a tissue from a program and the manner in which they are expressed (e.g., early induction versus late repression) are chosen probabilistically and influenced by the behavior of similar tissues. Further, usage is by definition consistent across a program for a particular tissue, which facilitates biological interpretation. Figure 4-1 presents an overview of the extended model.

GeneProgram++ usage modifier variables are perhaps best understood intuitively by imagining several ways in which we might have extended the original algorithm to handle different patterns of expression change. To simplify the discussion, we will describe examples in terms of gene expression induction or repression.

There are at least three ways we could imagine extending the GeneProgram algorithm. First, we could introduce an additional parameter for each gene in each expression program, indicating whether it is induced or repressed. In this scenario, expression programs would consist of sets of genes in which each gene is consistently either induced or repressed (but not both) in a subset of tissue samples. The problem with this approach is that expression programs could be difficult to interpret and would not really match our intuition for what constitutes an “atomic” or modular biological process, in which we expect all genes’ expression to coordinately change in the same direction in response to some stimuli. This expectation suggests a second possibility for extending the algorithm, in which we could introduce an additional parameter at the level of the expression program, indicating whether the genes in the program are induced or repressed. With this model, the direction of expression change for genes in a program would be consistent and would be the same for all tissue samples using the program. However, it is easy to imagine compendia of experiments in which an expression program is induced in some tissue samples and repressed in others. This then suggests the third possibility—the one GeneProgram++ uses—in which we introduce a parameter for each tissue sample at the level of the expression program, specifying the direction of expression change for genes in the program. Thus, with this model, the direction of expression change is consistent for all genes in a program for a particular tissue.

A variety of algorithms have been developed to analyze time-series expression data [17], but to our knowledge, none have been specifically designed for analysis of large compendia of such data. Analysis methods for combining time-series of different durations or that use different sampling rates have focused on long time-series over a few experimental conditions [1, 21], rather than short series over many conditions as we do. Jenner and Young performed a meta-analysis using hierarchical clustering of a superset of the infection time-course experiments we analyze in this chapter [107]. However, their analysis was not automated or statistically principled, relying on extensive prior biological knowledge, and using visual assessment of clusters to manually assign genes to pathways of interest. Further, as described in the Section 4.3, our analysis implicated several surprising signaling pathways and receptor types in the response to infection that previous analyses of these data sets have not.

The remainder of this chapter is organized as follows. In Section 4.1, we present the GeneProgram++ algorithm in detail. We first describe how a simplified version of GeneProgram without tissue groups is extended to include program usage modifiers. We then describe the full GeneProgram++ model and Markov Chain Monte

Carlo (MCMC) sampling methods for the model. In Section 4.2, we describe some additional technical improvements relating to posterior distribution summarization, and demonstrate the performance of these improvements on expression data. In Section 4.3, we apply GeneProgram++ to a compendium of 62 short time-series gene expression experiments in which various human cell types have been exposed to different infectious agents or immune-modulating substances [107], and produce a map of expression programs organized by functional generality scores. We evaluate the biological relevance of the discovered expression programs using biological process categories and pathway databases, as well as genome-wide data profiling binding of human transcription factors. Finally, we provide examples of discovered expression programs involved in key pathways related to the response to infection. We conclude the chapter with Section 4.4, in which we discuss the significance of our results.

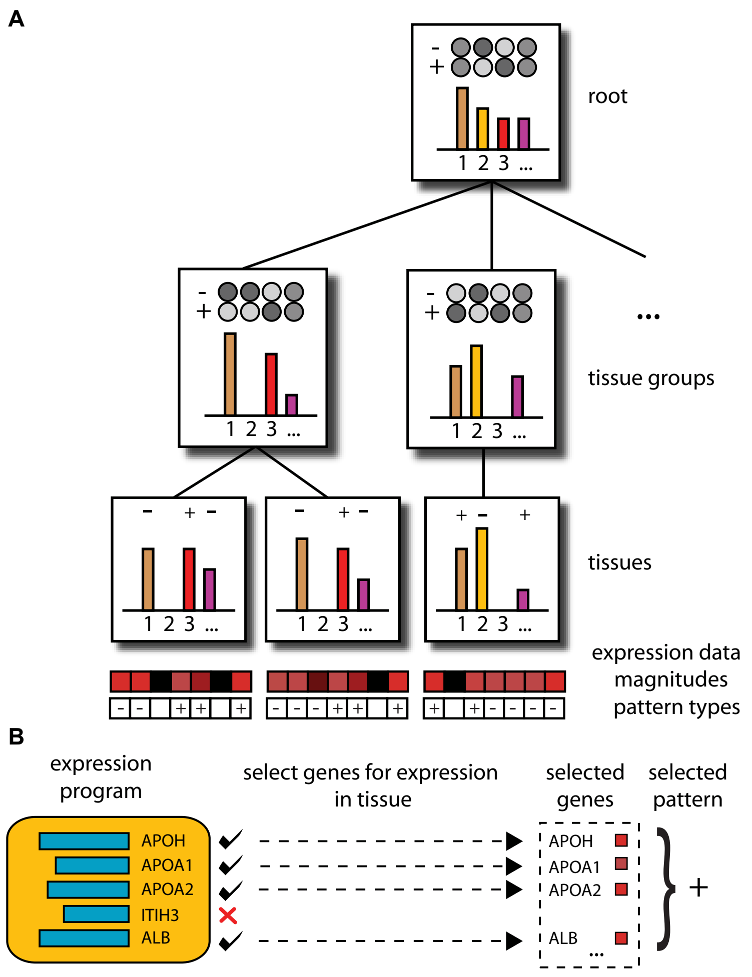


Figure 4-1: Overview of the GeneProgram++ model. *(A)* The model consists of a three-level hierarchy of Dirichlet Processes. Each node describes a weighted mixture of expression programs (each colored bar represents a different program; heights of bars = mixture weights). The distributions at each level are constructed on the basis of the parent mixtures above. Tissue group and root level nodes maintain distributions over usage modifiers (shaded circles above bars, darker circles = more probable), which are variables that alter the manner in which each tissue uses an expression program. In this example, there are two possible values for usage modifier variables (+ or -), corresponding to gene induction or repression; more complex patterns such as temporal dynamics can be captured by using more values. Tissues are at the leaves of the hierarchy, and choose particular values for usage modifier variables. The observed gene expression in each tissue is characterized by a vector of discretized expression magnitudes (first row of small shaded squares below each tissue) and pattern types (second row of squares with +/- designations below each tissue). *(B)* Example of a gene expression program, which represents a set of genes that are likely to behave coordinately in particular tissues that use the program. On the left is a simple program containing five genes (colored bars = expression frequencies). A tissue probabilistically chooses a set of genes from a program, and also a setting for its usage modifier variable. Note that usage is consistent across a program for a particular tissue, which facilitates biological interpretation.

4.1 Extending GeneProgram to model patterns of expression change

4.1.1 A simplified model without tissue groups

Model overview

We first develop the methodology for expression program usage modifiers in a simplified version of GeneProgram++ that does not include tissue groups. This version of the algorithm is useful for models in which we have a small number of tissue samples and so tissue groups are not applicable. Further, it provides a simpler context for initially understanding the inclusion of modifiers.

Figure 4-2 illustrates the model using standard graphical model notation with plates; Table 4.1 provides a summary of the random variables. See Section 3.3.2 for a description of the original GeneProgram probability model.

To describe the extensions to the simplified GeneProgram model, we will begin at the level of observations. In the original model, a tissue t had associated with it N_t units of expression denoted x_{ti}, where $1 \leq i \leq N_t$. In the extended model, we associate an observed pattern type with each unit of expression. The pattern type, denoted by y_{ti}, can take one of V values. For instance, if we are modeling induction and repression, $V = 2$ and $\mathsf{y}_{ti} \in \{-1, 1\}$, representing the direction of expression change for the gene.

Program usage modifier variables influence which pattern types are generated for genes in an expression program used by a specific tissue. We denote the usage modifier variable for tissue t using expression program j by u_{jt}, where u_{jt} can take on one of V values. Usage modifier variables influence how modifiers are generated through a compatibility function $\psi(\cdot,\cdot)$, which simply specifies the probability of generating a particular observed pattern type given some usage modifier value, i.e., $\psi(y_{ti}, u_{jt}) = \mathrm{p}(\mathsf{y}_{ti} \mid \mathsf{u}_{jt})$. As an example, if we are modeling induction and repression, we might specify a symmetrical compatibility function that returns a large probability when the usage modifier and pattern type variables take on the same value, and a small probability otherwise, i.e., $\psi(-1,-1) = \psi(1,1) = 0.99$ and $\psi(-1,1) = \psi(1,-1) = 0.01$.

Usage modifier variables themselves are generated via multinomial distributions parameterized by expression program level parameters $\mathbf{\Omega}$. In the simplified GeneProgram++ model, the $\mathbf{\Omega}$ and $\boldsymbol{\theta}$ parameters comprise a vector of mixture component parameters associated with each expression program. Each $\mathbf{\Omega}_j$ is a V-dimensional vector of parameters that specifies a multinomial distribution, i.e., $\mathrm{p}(\mathsf{u}_{jt} = v \mid \mathbf{\Omega}_j) = \Omega_{jv}$. Finally, we place a Dirichlet prior on the $\mathbf{\Omega}$ parameters. That is, $\mathbf{\Omega}_j \sim \mathrm{Dirichlet}(a_1^{\Omega}, \ldots, a_V^{\Omega})$, where $\boldsymbol{a}^{\Omega}$ is a V-dimensional vector of hyperparameters.

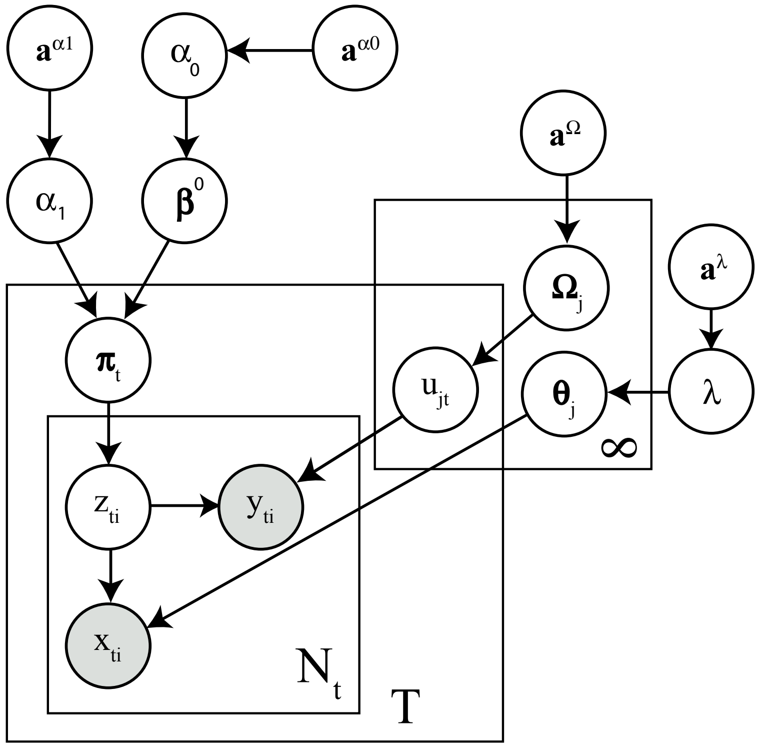


Figure 4-2: A simplified version of the GeneProgram++ model without tissue groups is depicted using graphical model notation with plates. Circles represent variables, and arrows denote dependencies among variables. Vectors are depicted with bold type, and observed variables are shown inside shaded circles. Rectangles represent plates, or repeated sub-structures in the model. See the text and Table 4.1 for details.

Var.	**Dim.**	**Description**	**Cond. distribution or prior**
x_{ti}	1	Expression event i in tissue t; corresponds directly to observed data.	Multinomial, given the assignment to expression program j.
y_{ti}	1	Pattern type for expression event i in tissue t.	Multinomial, given the assignment to expression program j and selection of the program usage by the tissue.
z_{ti}	1	Assignment variable of an expression event to an expression program, i.e., $\mathsf{z}_{ti} = j$ indicates that expression event i in tissue t is assigned to expression program j.	Generated from mixing probabilities over expression programs for the tissue, i.e., $\mathrm{p}(\mathsf{z}_{ti} = j \mid \boldsymbol{\pi}_t) = \pi_{tj}$.
$\boldsymbol{\pi}_t$	∞	Mixing probabilities over expression programs for tissue t.	DP, given its parent mixing probabilities and concentration parameters, i.e., $\boldsymbol{\pi}_t \mid \alpha_1, \boldsymbol{\beta}^0 \sim \mathrm{DP}(\alpha_1, \boldsymbol{\beta}^0)$.
$\boldsymbol{\beta}^0$	∞	Root level mixing probabilities in the DP heterarchy.	DP, generated from the stick-breaking distribution given its concentration parameter, i.e., $\boldsymbol{\beta}^0 \mid \alpha_0 \sim \mathrm{Stick}(\alpha_0)$.
$\boldsymbol{\theta}_j$	M	Parameters for expression, program j, describing a multinomial distribution over meta-genes.	Dirichlet distribution prior (parameterized by λ).
λ	1	Pseudo-count parameter for a symmetric Dirichlet distribution.	Gamma distribution prior with a two-dimensional hyperparameter vector $\boldsymbol{a}^{\lambda}$.
u_{jt}	1	Usage modifier variable for expression program j by tissue t.	Multinomial, given the expression program level shared parameters, i.e., $\mathsf{u}_{jt} \mid \boldsymbol{\Omega}_j \sim \mathrm{Multinomial}(\boldsymbol{\Omega}_j)$.
$\boldsymbol{\Omega}_j$	V	Shared multinomial parameters for tissue usages of expression program j.	Dirichlet distribution prior (parameterized by $\boldsymbol{a}^{\Omega}$).
α_1	1	Concentration parameter for $\boldsymbol{\pi}_t$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\alpha_1}$.
α_0	1	Concentration parameter for $\boldsymbol{\beta}^0$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\alpha_0}$.

Table 4.1: Summary of random variables used in the simplified GeneProgram++ model. The columns are: variable name (vectors are in bold type), dimensions of the variable, description, and the conditional or prior distribution on the variable.

MCMC sampling for the simplified extended model

The MCMC sampling scheme for the simplified GeneProgram++ model requires only a few changes to the method used for the original GeneProgram algorithm (see Section 3.3.3). First, the expression program assignment variable z_{ti} posteriors now must incorporate the information about pattern types. For assignment to a non-empty expression program j, the conditional distribution for z_{ti} is now given by:

$$\mathrm{p}(\mathsf{z}_{ti} = j \mid \mathbf{z}_{-ti}, \boldsymbol{\beta}^0, \boldsymbol{\theta}, \boldsymbol{\Omega}, \mathbf{u}, \mathbf{x}, \mathbf{y}) \propto (\alpha_1 \beta^0_j + n^{-i}_{tj}) F(\mathsf{x}_{ti} \mid \boldsymbol{\theta}_j) \psi(y_{ti}, u_{jt}) \mathrm{p}(\mathsf{u}_{jt} \mid \boldsymbol{\Omega}_j)$$

For assignment to a new expression program, the conditional distribution for z_{ti} is now given by:

$$\mathrm{p}(\mathsf{z}_{ti} \neq \mathsf{z}_{tl} \; \forall \, t, l \neq i \mid \mathbf{z}_{-i}, \boldsymbol{\beta}^0, \boldsymbol{\theta}, \boldsymbol{\Omega}, \mathbf{u}, \mathbf{x}, \mathbf{y}) \propto$$
$$(\alpha_1 \beta^0_*) \int F(\mathsf{x}_{ti} \mid \boldsymbol{\xi}) H(\boldsymbol{\xi}) d\boldsymbol{\xi} \left(\sum_{v=1}^{V} \Omega_{*v} \psi(y_{ti}, v) \right)$$

Here, β^0_* represents the mixture weight for the new component and $\boldsymbol{\Omega}_*$ the respective new parameter values. The new weight is sampled as described previously. The new $\boldsymbol{\Omega}_*$ is simply sampled from its prior, i.e., $\boldsymbol{\Omega}_* \sim \mathrm{Dirichlet}(\boldsymbol{a}^{\Omega})$.

The usage modifier variables u_{jt} must also be sampled. The posterior distribution for these variables is given by:

$$\mathrm{p}(\mathsf{u}_{jt} = v \mid \mathbf{z}_{-i}, \boldsymbol{\Omega}, \mathbf{y}) \propto \Omega_{jv} \prod_{i=1}^{N_t} \psi(y_{ti}, v)$$

Finally, the $\boldsymbol{\Omega}$ parameters must be sampled. The posterior distribution for these parameters is given by:

$$\mathrm{p}(\boldsymbol{\Omega}_j \mid \mathbf{u}, \boldsymbol{a}^{\Omega}) \propto$$
$$\mathrm{p}(\boldsymbol{\Omega}_j \mid \boldsymbol{a}^{\Omega}) \prod_{t=1}^{T} \mathrm{p}(\mathsf{u}_{jt} \mid \boldsymbol{\Omega}_j) \propto$$
$$\mathrm{Dirichlet}(\boldsymbol{\Omega}_j \mid a^{\Omega}_1, \ldots, a^{\Omega}_V) \prod_{v=1}^{V} \Omega_j^{c^j_v} \propto$$
$$\mathrm{Dirichlet}(\boldsymbol{\Omega}_j \mid a^{\Omega}_1 + c^j_1, \ldots, a^{\Omega}_V + c^j_V)$$

Here, c^j_v denotes the number of tissues using expression program j with pattern type value v. The final line follows from conjugacy between the Dirichlet and multinomial distributions.

4.1.2 The full GeneProgram++ model

In this section we describe the addition of expression program usage modifiers to the full GeneProgram model that includes tissue groups. The difference between this version of the extended model and the simplified one is the introduction of a hierarchal prior for the expression program usage modifier variables. Figure 4-3 depicts the full extended version using graphical model notation with plates and Table 4.2 summarizes the random variables used in the model.

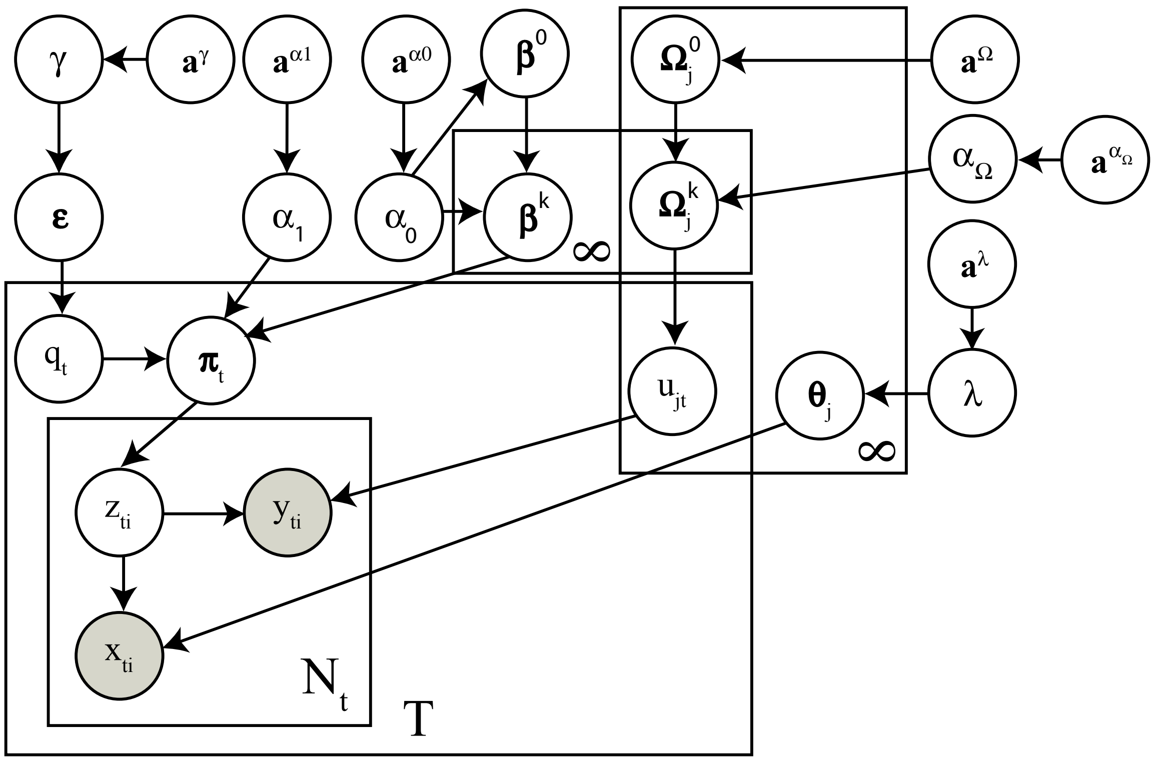


Figure 4-3: The full GeneProgram++ model is depicted using graphical model notation with plates. See the text for details. Circles represent variables, and arrows denote dependencies among variables. Vectors are depicted with bold type, and observed variables are shown inside shaded circles. Rectangles represent plates, or repeated sub-structures in the model.

As in the simplified extended model, usage modifier variables are generated via multinomial distributions parameterized by expression program level parameters. However, in the full extended model, we now have a separate set of such parameters, $\boldsymbol{\Omega}_j^k$, for each expression program j and tissue group k. For each expression program j, these parameters share a common top-level Dirichlet prior parameterized by $\boldsymbol{\Omega}_j^0$ and α_Ω. That is, $\boldsymbol{\Omega}_j^k \sim \text{Dirichlet}(\alpha_\Omega \Omega_{j1}^0, \ldots, \alpha_\Omega \Omega_{jV}^0)$. We assume that α_Ω has a Gamma prior with hyperparameter vector $\boldsymbol{a}^{\alpha_\Omega}$. As in the simplified model, $\boldsymbol{\Omega}_j^0 \sim \text{Dirichlet}(a_1^\Omega, \ldots, a_V^\Omega)$, where $\boldsymbol{a}^\Omega$ is a V-dimensional vector of hyperparameters.

MCMC sampling for the full extended model is essentially the same as for the

Var.	**Dim.**	**Description**	**Cond. distribution or prior**
x_{ti}	1	Expression event i in tissue t; corresponds directly to observed data.	Multinomial, given the assignment to expression program j.
y_{ti}	1	Pattern type for expression event i in tissue t.	Multinomial, given the assignment to expression program j and selection of the program usage by the tissue.
z_{ti}	1	Assignment variable of an expression event to an expression program, i.e., $\mathsf{z}_{ti} = j$ indicates that expression event i in tissue t is assigned to expression program j.	Generated from mixing probabilities over expression programs for the tissue, i.e., $\text{p}(\mathsf{z}_{ti} = j \mid \boldsymbol{\pi}_t) = \pi_{tj}$.
$\boldsymbol{\pi}_t$	∞	Mixing probabilities over expression programs for tissue t.	DP, given its parent mixing probabilities and concentration parameters, i.e., $\boldsymbol{\pi}_t \mid \alpha_1, \boldsymbol{\beta}^0 \sim \text{DP}(\alpha_1, \boldsymbol{\beta}^0)$.
$\boldsymbol{\beta}^k$	∞	Mixing probabilities over expression programs at the level of tissue group k; middle level in the DP hierarchy.	DP, given its parent mixing probabilities and concentration parameters, i.e., $\boldsymbol{\beta}^k \mid \alpha_0, \boldsymbol{\beta}^0 \sim \text{DP}(\alpha_0, \boldsymbol{\beta}^0)$.
$\boldsymbol{\beta}^0$	∞	Root level mixing probabilities in the DP heterarchy.	DP, generated from the stick-breaking distribution given its concentration parameter, i.e., $\boldsymbol{\beta}^0 \mid \alpha_0 \sim \text{Stick}(\alpha_0)$.
$\boldsymbol{\theta}_j$	M	Parameters for expression, program j, describing a multinomial distribution over genes.	Dirichlet distribution prior (parameterized by λ).
λ	1	Pseudo-count parameter for a symmetric Dirichlet distribution.	Gamma distribution prior with a two-dimensional hyperparameter vector $\boldsymbol{a}^{\lambda}$.
u_{jt}	1	Usage modifier variable for expression program j by tissue t.	Multinomial, given the tissue group k level shared parameters, i.e., $\mathsf{u}_{jt} \mid \boldsymbol{\Omega}_j^k, \mathsf{q}_t = k \sim \text{Multinomial}(\boldsymbol{\Omega}_j^k)$.
$\boldsymbol{\Omega}_j^k$	V	Tissue group k level parameters for usage modifiers of expression program j.	Dirichlet distribution prior, i.e., $\boldsymbol{\Omega}_j^k \sim \text{Dirichlet}(\alpha_\Omega \Omega_{j1}^0, \ldots, \alpha_\Omega \Omega_{jV}^0)$.
$\boldsymbol{\Omega}_j^0$	V	Root level parameters for usage modifiers of expression program j.	Dirichlet distribution prior (parameterized by $\boldsymbol{a}^{\Omega}$).
q_t	1	Assignment variable of tissues to groups, i.e., $\mathsf{q}_t = k$ indicates that tissue t belongs to tissue group k.	Generated from mixing probabilities over tissue groups, i.e, $\text{p}(\mathsf{q}_t = k \mid \boldsymbol{\epsilon}) = \epsilon_k$.
$\boldsymbol{\epsilon}$	∞	Mixing probabilities over the tissue groups.	DP, generated from the stick-breaking prior given its concentration parameter, i.e, $\boldsymbol{\epsilon} \mid \gamma \sim \text{Stick}(\gamma)$.
α_1	1	Concentration parameter for $\boldsymbol{\pi}_t$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\alpha_1}$.
α_0	1	Concentration parameter for $\boldsymbol{\beta}^0$ and $\boldsymbol{\beta}^k$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\alpha_0}$.
γ	1	Concentration parameter for $\boldsymbol{\epsilon}$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\gamma}$.
α_Ω	1	Concentration parameter for $\boldsymbol{\Omega}_j^k$.	Gamma distribution prior with two-dimensional hyperparameter vector $\boldsymbol{a}^{\alpha_\Omega}$.

Table 4.2: Summary of random variables used in the full GeneProgram++ model. The columns are: variable name (vectors are in bold type), dimensions of the variable, description, and the conditional or prior distribution on the variable.

simplified version, except for the sampling procedure for tissue assignments to groups, and for $\boldsymbol{\Omega}$ and α_Ω.

The posterior for assignment variables of tissues to groups, q_t, is now given by:

$$\mathrm{p}(\mathsf{q}_t = k \mid \mathbf{z}_t, \mathsf{q}_{-t}, \mathbf{y}, \alpha_0, \gamma, \boldsymbol{\beta}^k, \boldsymbol{\Omega}^0, \boldsymbol{\Omega}^k) \propto$$
$$\mathrm{p}(\mathsf{q}_t = k \mid \mathsf{q}_{-t}, \gamma) \prod_{i=1}^{N_t} \int \mathrm{p}(\mathsf{z}_{ti} \mid \boldsymbol{\pi}_t) \mathrm{p}(\boldsymbol{\pi}_t \mid \boldsymbol{\beta}^k, \alpha_0) d\boldsymbol{\pi}_t \left(\sum_{j=1}^{J} \sum_{i=1}^{N_t} \sum_{v=1}^{V} \Omega^k_{jv} \psi(y_{ti}, v) \right)$$

Here, J is the number of non-empty expression programs. If a tissue is assigned to a new group, we must also sample new parameters, i.e., $\boldsymbol{\Omega}^*_j \sim \text{Dirichlet}(\alpha_\Omega \Omega^0_{j1}, \ldots, \alpha_\Omega \Omega^0_{jV})$ for $j = 1, ..., J$. Similarly, when a gene expression unit is assigned to a new expression program, we must sample new parameters $\boldsymbol{\Omega}^k_*$ for each tissue group k from the prior.

We next need to sample from the posterior distributions for $\boldsymbol{\Omega}^k_j$. Because the prior for $\boldsymbol{\Omega}^k_j$ is Dirichlet given $\boldsymbol{\Omega}^0_j$ and the usage modifier variable posteriors are multinomial conditioned on $\boldsymbol{\Omega}^k_j$, the sampling posterior will simply be a Dirichlet distribution:

$$\mathrm{p}(\boldsymbol{\Omega}^k_j \mid \mathbf{u}, \boldsymbol{\Omega}^k_j, \alpha_\Omega) \propto \text{Dirichlet}(\boldsymbol{\Omega}^k_j \mid \alpha_\Omega \boldsymbol{\Omega}^0_{j1} + c^{jk}_1, \ldots, \alpha_\Omega \boldsymbol{\Omega}^0_{jV} + c^{jk}_V)$$

Here, c^{jk}_v denotes the number of tissues in group k using expression program j with pattern type v.

Sampling from the posterior for $\boldsymbol{\Omega}^0_j$ uses an auxiliary variable sampling scheme essentially the same as described in Section 3.2.2 for sampling Hierarchical Dirichlet Process mixture weights. As it turns out, the same sampling method can be used for a finite Dirichlet distribution with only a slight modification. As before, we introduce auxiliary variables $\mathbf{m}$. The conditional distributions for sampling $\mathbf{m}$ and $\boldsymbol{\Omega}^0_j$ are then:

$$\mathrm{p}(\mathsf{m}_{kjv} = m \mid \mathbf{u}, \mathbf{m}_{-kjv}, \boldsymbol{\Omega}^0_{jv}) \propto s(n_{kjv}, m)(\alpha_\Omega \boldsymbol{\Omega}^0_{jv})^m$$

$$\mathrm{p}(\boldsymbol{\Omega}^0_j \mid \mathbf{m}) \propto \text{Dirichlet}(\sum_k m_{kj1}, \ldots, \sum_k m_{kjV})$$

Here, n_{kjv} is the number of tissues in group k using expression program j with value v.

The parameter α_Ω is also sampled using an auxiliary variable sampling scheme essentially the same as described in section 3.2.2 for sampling Hierarchical Dirichlet Process concentration parameters. We introduce two auxiliary variables $\mathbf{w}$ and $\mathbf{b}$. The update equations are then given by:

$$\mathrm{p}(\mathsf{w}_{kj} \mid \alpha_\Omega) \propto w^{\alpha_\Omega}_{kj} (1 - w_{kj})^{T_k - 1}$$

$$\mathrm{p}(\mathsf{b}_{kj} \mid \alpha_\Omega) \propto \left(\frac{T_k}{\alpha_\Omega} \right)^{b_k j}$$

$$\mathrm{p}(\alpha_\Omega \mid \mathbf{w}, \mathbf{b}, \boldsymbol{a}^{\alpha_\Omega}) \propto \text{Gamma}(a^{\alpha_\Omega}_1 + \sum_{k=1}^{K} \sum_{j=1}^{J} (M_{kj} - b_{kj}), a^{\alpha_\Omega}_2 - \sum_{k=1}^{K} \sum_{j=1}^{J} \log w_{kj})$$

Here, T_k is the number of tissues in group k, $a_1^{\alpha\Omega}$ and $a_2^{\alpha\Omega}$ are the hyperparameters for the Gamma prior on α_Ω and $M_{kj} = \sum_{v=1}^{V} m_{kjv}$.

4.2 Additional GeneProgram++ algorithmic improvements

4.2.1 Recurrent expression programs

The GeneProgram++ algorithm addresses two limitations in the original recurrent expression programs (REPs) method presented in Section 3.3.4.

The first limitation of the original method related to the calculation of REP usage scores for tissues. REP usage for a tissue in the original method was equal to the number of expression events in the tissue allocated to the expression program divided by the total number of expression events in the tissue. Because tissues can have different numbers of expression events, the original usage score made comparisons of scores difficult across tissues. Further, the original usage score did not take into account expression probabilities for genes in a program. GeneProgram++ improves upon the original usage score by weighting by the probabilities of genes in the expression program. For tissue t using expression program j in sample s, GeneProgram++ calculates the tissue usage score $v_{tj}^{(s)}$ as:

$$v_{tj}^{(s)} = \sum_i \theta_{\omega(x_{ti})j} I(z_{ti}^{(s)} = j)$$

Here, $\omega(\cdot)$ is the function mapping expression events to genes, $\boldsymbol{\theta}_j$ is the probability vector for genes in program j, $I(\cdot)$ is the indicator function, and $z_{ti}^{(s)}$ is a variable denoting the assignment of expression event i to an expression program in iteration s. Note that this score will be higher if a tissue uses more genes from the program, regardless of the total number of expression events in the tissue. Further, the score will be higher if a tissue uses genes with large $\boldsymbol{\theta}_j$ values (i.e., higher probabilities of being expressed). Thus, the score better reflects how "typical" a tissue's usage of the REP is.

The empirical mean expression level $\widehat{e}_{gj}$ for gene g in REP j was also modified to reflect the improved usage scores:

$$\widehat{e}_{gj} = \sum_{s=1}^{S} \frac{\sum_{t,i} \mathrm{I}(z_{ti}^{(s)} = j)\theta_{gj}}{S \sum_{t=1}^{T} v_{tj}^{(s)}} \ \text{ s.t. } \omega(x_{ti}) = g$$

The second limitation of the original method related to how REPs were tracked. In the original method, REPs were tracked using a "signature" based on the tissues that used them significantly. A limitation of this method was that tissues using the expression program slightly below the significance threshold would not be included in the REP. This could lead to the creation of many REPs with essentially the same gene probabilities, but different "signatures" of significant tissues. Although the

REPs could be merged in subsequent post-processing steps, having to track a large number of REPs substantially reduced the algorithm's performance and became infeasible for very large data sets. To deal with these issues, GeneProgram++ no longer tracks REPs based on significant tissues. Instead, the new algorithm saves all expression programs at each iteration and then sequentially merges similar programs based on how similar the gene expression probabilities are for programs. Similarity is calculated using the Hellinger distance, a general distance metric for probability measures [22]. To be precise, the Hellinger distance between expression programs j_1 and j_2 is calculated as:

$$D(\boldsymbol{\theta}_{j_1}||\boldsymbol{\theta}_{j_2}) = \sum_{g=1}^{G} \sqrt{\theta_{gj_1}\theta_{gj_2}}$$

Here, G is the total number of genes. Expression programs are merged if the distance between them is less than some threshold c_2. A tissue t is reported as associated with a REP j if its mean usage score $\overline{v}_{tj}$ is greater than some threshold c_1. We used values of $c_1 = 0.25$ and $c_2 = 0.50$ for the applications in this chapter.

In the applications in this chapter, we used 1,000 samples to generate REPs as follows. After burn-in of the MCMC sampler for 100,000 iterations, 10,000 samples were generated, with 1,000 samples saved and 100 iterations between each saved sample discarded. Spaced samples from the MCMC sampler better approximate independent samples from the posterior, and can thus result in more accurate results [72]. Note that the improved method for merging REPs as well as the use of spaced samples allowed us to use fewer overall samples than we did for the original GeneProgram applications.

4.2.2 Generality score

GeneProgram++ improves on the generality score calculations of the original algorithm, discussed in Section 3.5.7, in two ways. First, GeneProgram++ employs the improved tissue usage scores described in Section 4.2.1 to compute the generality scores. Second, GeneProgram++ computes a generality score for each expression program in each MCMC sample, and then averages scores across all samples. The original algorithm instead computed generality scores based on consensus tissue groups determined after all samples had been summarized. The limitation of the original method is that generality scores did not reflect different assignments of tissues to groups in samples, and thus did not take full advantage of the availability of the approximate posterior distribution.

GeneProgram++ computes the generality score as follows. First, the algorithm computes the usage $h_{kj}^{(s)}$ for tissue group k of REP j in sample s as:

$$h_{kj}^{(s)} = \sum_{t=1}^{T} v_{tj}^{(s)} \mathrm{I}(q_t^{(s)} = k)$$

Here, $q_t^{(s)}$ is the assignment of tissue t to a tissue group in sample s, and $v_{tj}^{(s)}$ is the tissue usage score described in Section 4.2.1. The $h_{kj}^{(s)}$ values are then normalized across all tissue groups in the sample, i.e.,:

$$\widehat{h}_{kj}^{(s)} = \frac{h_{kj}^{s}}{\sum_{l=1}^{K} h_{lj}^{(s)}}$$

Here, K is the total number of non-empty tissue groups in the sample. The generality score for expression program j in sample s is then computed as:

$$\mathrm{GS}_j^{(s)} = -\sum_{k=1}^{K} \widehat{h}_{kj}^{(s)} \log \widehat{h}_{kj}^{(s)} \tag{4.1}$$

The final generality score for a REP is then simply the mean of generality scores computed in equation 4.1, averaged across all samples in which the relevant expression programs occur.

4.2.3 Validation of improvements using real expression data

In Section 3.5.4, we showed that GeneProgram outperformed two popular biclustering methods, Samba [209, 194] and a non-negative matrix factorization (NMF) implementation [36], in the discovery of biologically relevant gene sets from real expression data. Here, we demonstrate that GeneProgram++ outperforms these same biclustering methods to an even greater extent.

As in Section 3.5.4, in this section we also used the Novartis Gene Atlas v2 data set [205] for our evaluation; preprocessing of this data set was performed as described previously. We also used a second data set, from Shyamsundar *et al.*, consisting of 115 human tissue samples obtained from surgeries or autopsies, with expression measured on custom cDNA microarrays [198]. For this data set, the reference channel on the microarrays consisted of mRNA pooled from 11 established human cell lines. We considered a gene expressed if its ratio was greater than 2.0. In order to combine the data from the Novartis Gene Atlas v2 and Shyamsundar *et al.*, we mapped genes to common identifiers using the IDConverter software [6], and retained only the 7,404 genes present in both data sets.

As in Section 3.5.4, in this section we also used both Gene Ontology (GO) categories [11] and manually derived, broadly physiologically based tissue categories to assess the algorithms' performance. For a description of the tissue categories, see the supplemental online material for [76].

However, GO categories and the manually derived tissue categories represent only limited biological knowledge. So, we were also interested in assessing the consistency of gene sets discovered by each algorithm across the two data sets. Because the two data sets used different microarray platforms and sources for tissues, similarities in discovered gene sets between data sets were likely to be biologically relevant. To analyze gene set consistency for each algorithm, we used the gene sets discovered from one data set to compute the significance of the overlap with sets produced using the second data set. We then inverted the analysis and averaged the results

data source	**algorithm**	**gene dimension** (GO cat. enrich.)	**tissue dimension** (man. derived cat. enrich.)
N	GeneProgram++	93%	76%
N	NMF	35%	29%
N	Samba	53%	9%
S	GeneProgram++	66%	53%
S	NMF	28%	19%
S	Samba	51%	28%

Table 4.3: GeneProgram++ outperformed two popular biclustering algorithms, a non-negative matrix factorization (NMF) implementation and Samba, in recovering biologically interpretable gene sets from two compendia of mammalian gene expression experiments. Further, GeneProgram++ substantially improved on the results obtained using GeneProgram for this same analysis (see Section 3.5.4). Biological interpretability of gene sets was assessed using Gene Ontology (GO) categories in the gene dimension, and manually constructed categories in the tissue dimension. Each cell in the table shows the percentage of sets significantly enriched for at least one category in a given dimension (p-value < 0.05, corrected for multiple hypothesis tests). The data compendia are the Novartis Tissue Atlas v2 (N) [205] and the Shyamsundar *et al.* human tissue data (S) [198].

to produce correspondence plots. See Section 3.5.4 for details on the generation of correspondence plots.

GeneProgram++ clearly outperformed the other algorithms in both the tissue and gene dimensions on the two data sets considered separately (Table 4.3), and substantially improved on the results obtained using GeneProgram for this same analysis (see Section 3.5.4). GeneProgram++ also clearly outperformed NMF and Samba in terms of gene set consistency between the data sets (Figure 4-4).

4.3 Application to human infection time-series data

We applied GeneProgram++ to a compendium of sixty-two short time-series gene expression data sets exploring the responses of human cells to various infectious agents or immune-modulating molecules.

4.3.1 Data sources

Expression data

The expression data used was compiled by Jenner and Young [107] from six separate studies. A total of 5042 genes were present in the combined data sets. All data analyzed were converted to log ratios for each time-point versus the first (pre-exposure) time-point in the series. Below we briefly describe the data sets; see Table 4.4 for a summary.

Nau *et al.* exposed primary human macrophages collected from different donors to a variety of live bacteria and bacterial cell components [149]. The macrophages were

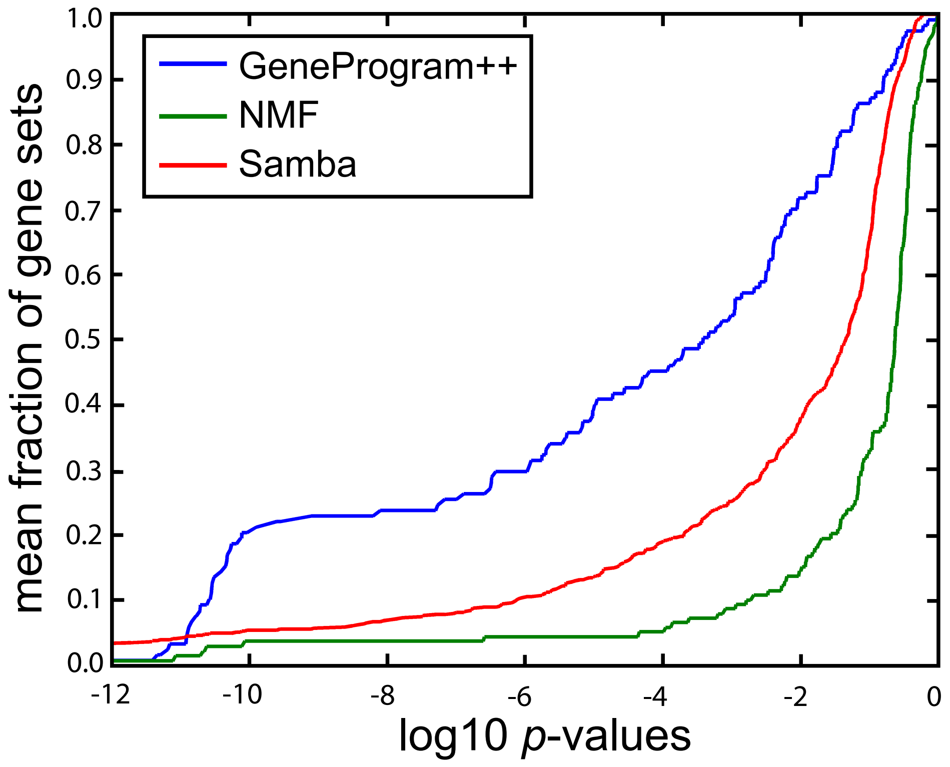


Figure 4-4: GeneProgram++ outperformed two popular biclustering methods, a non-negative matrix factorization (NMF) implementation and Samba, in terms of gene set consistency between two large compendia of mammalian tissue gene expression data. Because the two data compendia used different microarray platforms and sources for tissues, similarities in discovered gene sets between compendia were likely to be biologically relevant. For each algorithm, we used gene sets discovered from one data compendium to compute the significance of the overlap (p-values) with sets produced using the second compendium. We then inverted the analysis and averaged the results to produce the correspondence plots shown. The plots depict log p-values on the horizontal axis and the fraction of gene sets with p-values below a given value on the vertical axis (see Section 3.5.4 for details). The larger fraction of gene sets at most p-values suggests that GeneProgram++ generally produces the most consistent results between the data compendia.

exposed to an infectious agent or component for twenty-four hours, and four to five time-points were collected. Infectious agents included representative species from the major classes of human bacterial pathogens: Gram-positive organisms (*Staphylococcus aureus* and *Listeria monocytogenes*), Gram-negative organisms (*Escherichia coli*, enterohemorrhagic *E. coli* O157:H7 [EHEC], *Salmonella typhi* and *Salmonella typhimurium*), and Mycobacteria (*Mycobacterium tuberculosis* and *Mycobacterium bovis*). The macrophages were also exposed to bacterial cell components, including those specific to Gram-negative organisms (lipopolysaccharide [LPS]) or Gram-positive organisms (lipoteichoic acid [LTA] and protein A), and components common to all three classes of bacteria (heat shock proteins [hsp65 and hsp70], muramyl dipeptide [MDP], formyl-methionine-leucine-phenylalanine [f-MLP] and mannosylated proteins [D-(+)-mannose]).

In a follow-up study, Nau *et al.* exposed human macrophages to five different immune-modulating molecules [150]. The molecules used were: interferon-alpha (IFN-α), interferon-beta (IFN-β), interferon-gamma (IFN-γ), interleukin 10 (IL-10) and interleukin 12 (IL-12). The macrophages were exposed for twenty-four hours and five time points were collected.

Huang *et al.* exposed primary human dendritic cells collected from different donors to several live infectious agents and relevant components of the agents [91]. Dendritic cells, which reside in tissues in an immature state, are involved in initiating both innate and adaptive immune responses. They recognize and phagocytose antigens, which then leads to their maturation, expression of co-stimulatory molecules, and subsequent interactions with naive T-cells. Dendritic cells were exposed to representative organisms and relevant components: bacteria (*E. coli* and LPS), fungi (*Candida albicans* and mannan, a fungal cell-wall component) and viruses (Influenza A and synthetic dsRNA). Cells were exposed for twenty-four or thirty-six hours, and seven to eight time-points were collected.

Boldrick *et al.* exposed primary human peripheral blood mononuclear cells (PBMCs) from different donors and cell-line derived macrophages to several heat-killed and live bacterial pathogens, a bacterial component, avirulent bacterial strains, and immunostimulatory chemicals [33]. Cells were exposed to heat-killed *E. coli*, *S. aureus* and *Bordatella pertussis* bacteria for six, twelve, or twenty-four hours and five to seven time-points were collected. Additional experiments were done using live virulent *B. pertussis*, avirulent strains, and LPS from the virulent organism. Additionally, cells were exposed to ionomycin and phorbol 12-myristate 13-acetate (PMA), chemicals that induce cellular responses mimicking antigen exposure. The authors noted that heat-killed strains were used to reduce confounding effects from differential bacterial growth rates and cytotoxic effects on host cells. *B. pertussis* was chosen for live infections, because this organism is relatively slow-growing and is not known to cause significant cytotoxicity. The authors additionally noted that the use of peripheral blood mononuclear cells, which consist of diverse cell populations involved in both innate and adaptive immunity, had advantages because interactions among different immune cells might be observed.

Pathan *et al.* exposed whole human blood cells from two donors to the live pathogenic bacteria *Neisseria meningitidis* [160]. Cells were exposed for twenty-four

agents	**host cell-type(s)**	**no. time-series**	**no. time-points**	**ref.**
live pathogenic bacteria and bacterial cell components; representative Gram-positive, Gram-negative and Mycobacteria organisms	primary macrophages	23	4–5 (24 hrs)	[149]
interferons and interleukins	primary macrophages	5	5 (24 hrs)	[150]
representative pathogenic bacteria, fungi, and viruses and relevant components	primary dendritic cells	9	7–8 (24–36 hrs)	[91]
live and heat-killed pathogenic bacteria, avirulent strains and bacterial components	primary peripheral blood mononuclear cells and cell-culture macrophages	15	5–7 (6–24 hrs)	[33]
live pathogenic bacteria	whole blood cells	4	5 (24 hrs)	[160]
live pathogenic bacteria and mutant strains	cell-culture gastic epithelial cells	6	5 (24 hrs)	[80]

Table 4.4: Summary of infection gene expression time-series data sets analyzed. All host cells in the experiments were human primary or cell-culture derived.

hours and five time-points were collected.

Guillemin *et al.* exposed cell-culture derived human gastric epithelial cells to the live pathogen *Helicobacter pylori*, a leading cause of peptic ulcers [80]. Cells were exposed for twenty-four hours and five time-points were collected. Guillemin *et al.* also exposed cells to four different *H. pylori* mutants deficient in various genes of the *cag* pathogenicity island, a contiguous collection of genes that confer virulence properties. Unlike the host cells used in the other data sets described above, gastric epithelial cells are not involved in principle immune system functions.

Genome-wide binding data

Members of the NF-κB family of transcription factors are key controllers of mammalian immune and inflammatory responses [41]. The NF-κB family members form homodimers or heterodimers that bind to specific 9-10 base pair sites in promoters through a conserved DNA-binding/dimerization domain termed the Rel homology domain (RHD). Some NF-κB family members also contain a transactivation domain (TAD), which interacts directly with co-activators or components of the general transcription machinery; apparently both the TAD and RHD domains are necessary for transcriptional activation. The various homodimers and heterodimers formed by the NF-κB proteins are believed to have at least some degree of target specificity. As mentioned, some NF-κB proteins do not contain TADs, which may allow relevant dimers to act as repressors. Table 4.5 provides a summary of NF-κB proteins and their proposed dimerization partners.

protein	**TAD?**	**comments**	**partners**
p50	no	amino terminal half of p105; homodimer is thought to act as a repressor, but may also act as an activator when it interacts with BCL-3	p50, RELA, RELB, c-REL
p52	no	amino terminal half of p100; homodimer is thought to act as a repressor	p52, RELB
RELA (p65)	yes	generally thought to be an activator, but some evidence that it can function as a repressor	p50, RELA, c-REL
c-REL	yes		p50, RELA, c-REL
RELB	yes	no homodimer	p50, p52

Table 4.5: Overview of the NF-κB proteins profiled in genome-wide binding assays in human cell-culture deived macrophages by Schreiber *et al.* [185]. Information from the table is from [185, 41]. The "TAD?" column indicates whether the protein contains a transactivation domain (TAD). The "partners" column indicates proposed dimerization partners.

As a validation data set, we used static genome-wide data profiling binding of five transcription factors in the NF-κB family in untreated or lipopolysaccharide (LPS) stimulated human cell-culture derived macrophages [185]. Binding data was obtained after one hour of LPS stimulation and 9492 genes were arrayed for the ChIP-chip analysis. We used the same criteria for determining binding events as in the original study [185] (a p-value threshold of 0.002).

4.3.2 Temporal pattern and expression level discretization

Behavior for each gene over each time-series experiment was mapped to one of six simple temporal patterns. The six patterns characterize three temporal phases (early, middle, or late) with two possible directions (induction or repression) for the first significant expression change for each gene in each time-series experiment. See Figure 4-5 for a summary of the temporal patterns used, and Figure 4-6 for example genes with expression profiles corresponding to the patterns.

To be precise, time-points for all experiments were divided into three general phases: early (less than two hours), middle (greater than two hours and not more than twelve hours) and late (greater than twelve hours). For each gene in each time-series experiment, the gene was assigned to the pattern corresponding to the earliest phase in which the gene's expression value exceeded a two-fold increase (decrease) in at least one experiment in the respective time interval. Further, to be assigned to a pattern, a gene's expression across the time-series was required to be consistent in direction (either a two-fold increase or decrease in expression but not both). If a gene's expression profile did not meet all these criteria, it was not assigned to any pattern. The absolute expression value for a gene's earliest induction (repression) in each time-series was then used to represent the magnitude of differential expression.

These values were discretized using the method described in Section 3.3.5, with two levels chosen.

A limited set of possible temporal patterns was intentionally chosen for two reasons. First, in the original studies, a primary feature of interest for all the experiments analyzed was the time of earliest induction or repression of each gene [33, 91, 149, 150]. Thus, a small set of relevant temporal patterns aids in the biological interpretability of our results. Second, the time-series data sets analyzed had different durations, sampling rates and numbers of samples. By considering only simple temporal patterns that extract features present in all time-series, we could produce meaningful results spanning all the data sets.

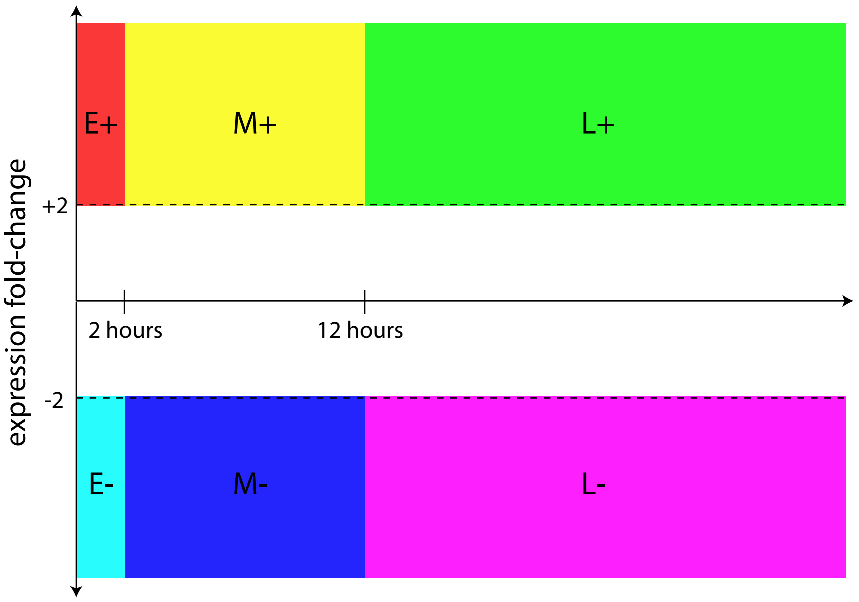


Figure 4-5: Schematic of six pre-defined temporal patterns used in analyzing the infection time-series compendium. Time-points for all experiments were divided into three general phases: early (less than two hours), middle (greater than two hours and not more than twelve hours) and late (greater than twelve hours). For each gene in each time-series experiment, the gene was assigned to the pattern corresponding to the earliest phase in which the gene's expression value exceeded a two-fold increase (decrease) in at least one experiment in the respective time interval.

4.3.3 GeneProgram++ discovered 5 consensus tissue groups and 104 expression programs in the infection time-series data

Figures 4-7 (programs 1–25), 4-8 (programs 26–50), 4-9 (programs 51–75) and 4-10 (programs 76–104) summarize the discovered tissue groups and expression programs. See the supplemental online material for [76] for the complete data.

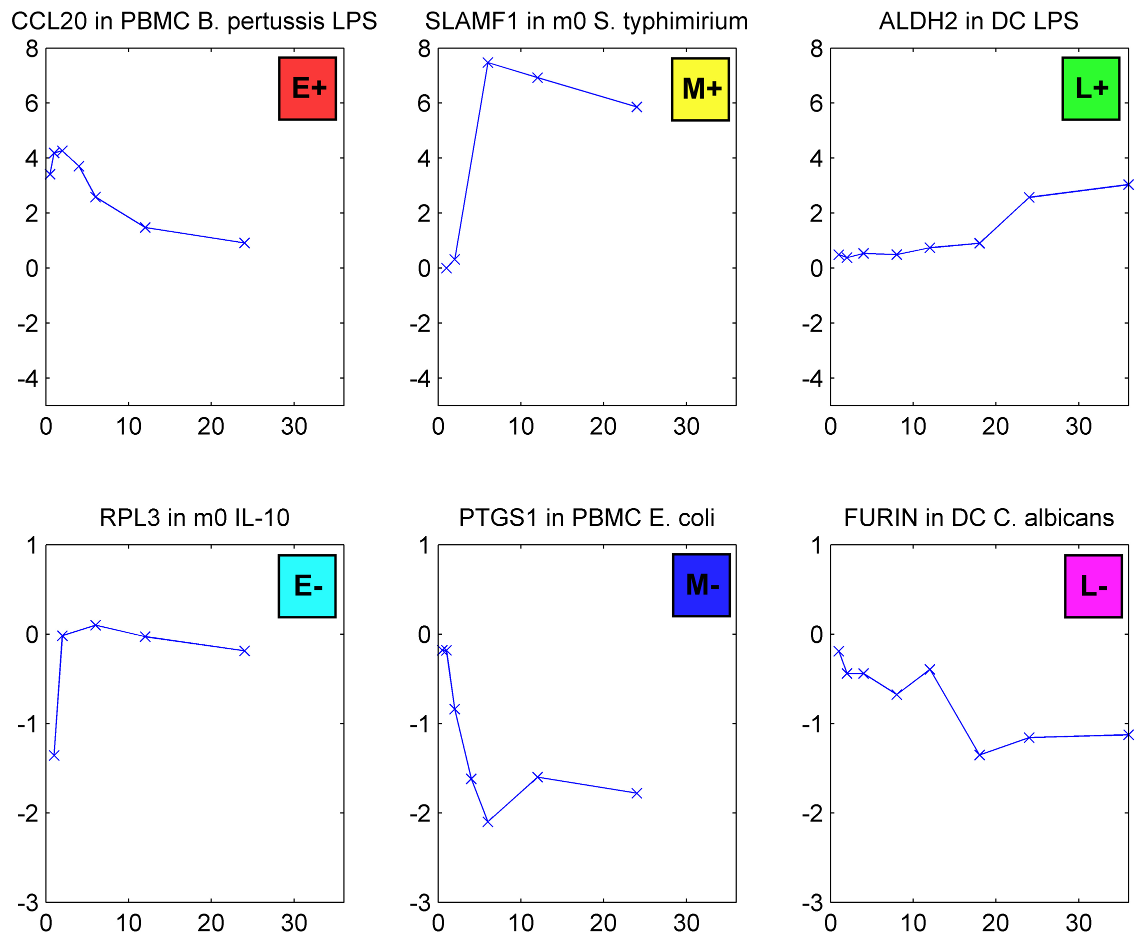


Figure 4-6: Examples of genes with the six temporal expression patterns used for data analysis. The top plots depict, from left to right, early ($\leq$ 2 hours), middle ($>$ 2 and $\leq$ 12 hours) and late ($>$ 12 hours) induction patterns; the bottom plots depict analogous repression patterns. For all plots, the vertical axis represents $\log_2$ transformed expression fold-changes and the horizontal axis represents time in hours. See the text for complete details.

The tissue groups essentially corresponded to the different host-cell types used for the infection experiments, although there was some intermingling of the dendritic and peripheral blood mononuclear cell experiments and those using other host-cell types. Expression programs were used by 2-27 experiments (median = 7) and contained 101-410 genes (median = 201).

All six temporal patterns were used, although late induction and late repression were used least frequently, likely in part because the corresponding time intervals were the most sparsely sampled in the compendium analyzed. In many cases, usage patterns for a single program were uniformly inductive or repressive, although programs were sometimes used with differently phased patterns by different experiments. However, in other interesting cases, usage patterns were not uniformly inductive or repressive. Specific examples of expression programs with different temporal pattern usage are discussed in Section 4.3.5.

4.3.4 Validation of the biological relevance of the discovered expression programs

Expression programs overlapped extensively with key human signaling pathways and biological processes

To evaluate the biological relevance of expression programs, we used two external sources of information about gene function: GO biological process categories [11] and Kyoto Encyclopedia of Genes and Genomes (KEGG) pathways [109]. Computation of enrichment scores, significance testing and correction for multiple hypotheses were done as described in Section 3.5.3.

A large number of expression programs were significantly enriched for GO categories (50%) or KEGG pathways (59%). Specific examples of expression programs enriched for genes involved in key pathways and biological processes are discussed in Section 4.3.5; below, we describe general trends for the expression programs.

As expected, many significant GO categories and KEGG pathways were specifically involved with the response to infection. Examples include inflammatory response (GO: 0006954), response to virus (GO: 0009615), chemotaxis (GO: 0006935), positive regulation of T-cell proliferation (GO: 0042102), endogenous antigen processing via MHC class II (GO: 0019886), lymph node development (GO: 0048535), cytokine-cytokine receptor interaction (KEGG: hsa04060), natural killer cell mediated cytotoxicity (KEGG: hsa04650), leukocyte transendothelial migration (KEGG: hsa04670) and complement and coagulation cascades (KEGG: hsa04610).

There were also a number of significantly enriched biological processes or pathways not directly labeled as being infection-related, but that are involved with significant changes in cellular physiology consistent with infection. Examples include apoptosis (GO: 0006915), nucleotide-excision repair (GO: 0006289), nuclear mRNA splicing (GO: 0000398), glycolysis (GO: 0006096), glycogen metabolism (GO: 0005977), anti-apoptosis (GO: 0006916), cell cycle (GO: 00070490), positive regulation of cell proliferation (GO: 0008284), tricarboxylic acid cycle (GO: 000609), regulation of adenylate cyclase activity (GO: 0045761), chloride transport (GO: 0006821), focal adhe-

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	m0 control1																									
	m0 control2																									
	m0 control3					E-																				M-
	m0 control4																									
	m0 Latex																									
	m0 E. coli																									
	m0 EHEC					E-																				E-
	m0 S. typhi																									
	m0 S. typhimirium																									
	m0 S. aureus					E-																				E-
	m0 L. monocytogenes					E-																				M-
	m0 M. tuberculosis																									
	m0 M. bovis BCG																									
	m0 LPS E																									
	m0 LPS S																									
	m0 LTA																									
	m0 MDP																									
	m0 TB hsp70					E-																				E-
	m0 BCG hsp65					E-																				
	m0 MPA																									
	m0 f-MLP																									
	m0 Protein A																									
	m0 Mannose																									
	m0 IFN-alpha																									
	m0 IFN-beta																									
	m0 IFN-gamma																									M+
	m0 IL-10																									
	m0 IL-12																									
	DC control1																									
	DC control2																									
	DC control3					E-																				E-
B	m0 (U937) control1										E-															
	m0 (U937) B. pertussis 338								E-													E+				
	m0 (U937) B. pertussis 537 (avirulent)			E-					E-		M-		E-									E+				
	m0 (U937) B. pertussis A2-6 (AC-)												E-													
	m0 (U937) B. pertussis Tox6 (PT-)			E-					E+				E+									E+				
	PBMC control1			E+	M-			M+			M-		M-			E+			E-		E+					
	PBMC B. pertussis LPS			E-	E+			M+	E-		E-		M-			E+			E+		E+	E-				
	PBMC B. pertussis 338			E+															E+							
	PBMC B. pertussis Minnesota1			E+																		M-				
	PBMC E. coli			E+																		E-				
	PBMC S. aureus (1)				E-			M-			E-					E+			M-			M-				
	PBMC S. aureus (2)			E+	M-			E+	E-		E-					M-			M-		E-	E-				
	PBMC Ionomycin+PMA			E+	E-														M-		E-					
C	DC E. coli																									
	DC LPS																									
	DC Influenza																									
	DC C. albicans																									
	DC Mannan																									
	DC PolyIC																									
D	PBMC B. pertussis 338 Killed																E+							E+	E-	
	PBMC B. pertussis 338 Live																E+			E-				M+	M-	
	WB control NP																M+	M-								
	WB control SW																E+	M-		M-						
	WB N. meningitidis NP																E+	M-		M-						
	WB N. meningitidis SW																E+	E-		M-						
E	Epithelial cell control	E-	M-				E-			E-		M+		E+	M-											
	Epithelial cell H. pylori G27	E-	E-				E-			M-		E-		E+	M+								M+			
	Epithelial cell H. pylori cagN-	M-	M-				E-			M-		E-		E+	E-								M+			
	Epithelial cell H. pylori cagA-	M-	E-				E-			M-		E-		E+	M+								M+			
	Epithelial cell H. pylori cagE-	E-	E-				E-			M-		E-		E+	M+								M+			
	Epithelial cell H. pylori PAI-	M-	M-				E-			E-		M+		E+	M+								M+			
		0.001							0.005							0.012							0.011			

Figure 4-7: Summary of expression programs (EPs) 1–25 discovered by GeneProgram++ in the infection data (see Figures 4-8, 4-9 and 4-10 for the remaining EPs). Letters A–E label the five tissue groups discovered by the algorithm. Each entry in the matrix represents the usage of an expression program (darker shading = higher usage), and matrix entries are colored and labeled with temporal patterns. The patterns are early induction (E+, red), middle induction (M+, yellow), late induction (L+, green), early repression (E-, cyan), middle repression (M-, blue) and late repression (L-, light purple). Generality scores are shown at the bottom of the figure; programs are sorted from lowest to highest scores (left to right).

sion (KEGG: hsa04510), oxidative phosphorylation (KEGG: hsa00190), proteasome (KEGG: hsa03050), androgen and estrogen metabolism (KEGG: hsa00150), regulation of actin cytoskeleton (KEGG: hsa04810), gap junction (KEGG: hsa04540), and fatty acid metabolism (KEGG: hsa00071).

Interestingly, a substantial number of the significantly enriched GO categories or KEGG pathways corresponded to signaling cascades. Examples include MAPKKK (GO: 0000165 and KEGG: hsa04010), JAK-STAT (GO: 0007259 and KEGG: hsa04630), Toll-like receptor, (KEGG: hsa04620), B-cell receptor (KEGG: hsa04662), T-cell receptor (KEGG: hsa04660), insulin (KEGG: hsa04910), VEGF (KEGG: hsa04370), calcium (KEGG: hsa04020), phosphatidylinositol (KEGG: hsa04070), I-κB kinase/NF-κB (GO: 0043123), and transmembrane receptor protein tyrosine kinase (GO: 0007169) signaling pathways.

There were also some surprising significantly enriched signaling pathways or biological processes. These are discussed further in Section 4.3.5.

A surprising number of expression programs contained many genes bound by NF-κB transcription factor family members

Because NF-κB transcription factors are key controllers of mammalian immune and inflammatory responses (see Section 4.3.1), we expected that some genes differentially expressed in the infection time-series compendium would also be bound by NF-κB transcription factors in the ChIP-chip experiments. Thus, to further evaluate the biological relevance of discovered expression programs, we used genome-wide data profiling binding of NF-κB family transcription factors in human cell-culture derived macrophages [185]. Computation of enrichment scores, significance testing and correction for multiple hypotheses were done as described in Section 3.5.3.

Fifteen of the expression programs discovered by GeneProgram++ were significantly enriched for sets of genes bound by at least one NF-κB family member. This overlap with the binding data was quite large, considering that only 348 genes were bound by NF-κB family members in the ChIP-chip experiments [185].

Overall, the fifteen significantly enriched programs tended to be those used by a diverse array of host-cell types exposed to a range of pathogens and their components. This suggests that these expression programs may represent common processes that

		26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
A	m0 control1										M+				M-											
	m0 control2															E+				E-	E+	E-				
	m0 control3															E+				E-					E+	
	m0 control4												M-								E+	M-	E+			E-
	m0 Latex						E-				M+				M-							E-				
	m0 E. coli														M-						E+					
	m0 EHEC															E+				E-	M-				E+	E-
	m0 S. typhi																								E+	
	m0 S. typhimirium																									
	m0 S. aureus												E-			E+				E-	M-				E+	E-
	m0 L. monocytogenes										M-					M+				E-	M-				E+	E-
	m0 M. tuberculosis														M-	E+				E-	E+	E-				
	m0 M. bovis BCG														M-											
	m0 LPS E						E-				M-									E-						
	m0 LPS S						E-				M-				M-							E+				M-
	m0 LTA										M-		M+													E+
	m0 MDP												E-									M-	M+			
	m0 TB hsp70															E+				E-					E+	
	m0 BCG hsp65															E+				E-					E+	E-
	m0 MPA						E-															E+				
	m0 f-MLP												E-									E-	E+			
	m0 Protein A												E-										M+			E-
	m0 Mannose												E-									E-	M+			M-
	m0 IFN-alpha										M-		E+							E+						M+
	m0 IFN-beta												E+							E+						M+
	m0 IFN-gamma										M-		E+							E+						
	m0 IL-10										E-		E+													M+
	m0 IL-12														M+					M+						M+
	DC control1																									
	DC control2																									
	DC control3																					E-				
B	m0 (U937) control1				E-									E-												
	m0 (U937) B. pertussis 338				E+																					
	m0 (U937) B. pertussis 537 (avirulent)			E-		E-		M-																		
	m0 (U937) B. pertussis A2-6 (AC-)																									
	m0 (U937) B. pertussis Tox6 (PT-)				E+	M-		E+																		
	PBMC control1	E-		E-	M+			M-						E+												
	PBMC B. pertussis LPS	M-		E-	E+	M+		E-						E-												
	PBMC B. pertussis 338			E-																						
	PBMC B. pertussis Minnesota1													E-												
	PBMC E. coli	E+		E-	E+									E-												
	PBMC S. aureus (1)	E-		E-	E-									E+												
	PBMC S. aureus (2)	E-		E-	M+			M+						E+												
	PBMC Ionomycin+PMA				E-									E+												
C	DC E. coli																									
	DC LPS																									
	DC Influenza																									
	DC C. albicans																									
	DC Mannan																									
	DC PolyIC																					E-				
D	PBMC B. pertussis 338 Killed		M+						M+	M-							M-									
	PBMC B. pertussis 338 Live		M+						E+	M-							M-									
	WB control NP								M+								M+							M-		
	WB control SW								M+								M+							M-		
	WB N. meningitidis NP								M+								M-							M-		
	WB N. meningitidis SW		M+						M+								M-							M-		
E	Epithelial cell control											E+						M-								
	Epithelial cell H. pylori G27											E-						M-	E+							
	Epithelial cell H. pylori cagN-											E-						E-	E+							
	Epithelial cell H. pylori cagA-											E-						M-	E+							
	Epithelial cell H. pylori cagE-											M-						E-	E+							
	Epithelial cell H. pylori PAI-											E-						E-	M+							
		0.013							0.022							0.057							0.113			

Figure 4-8: Summary of expression programs (EPs) 26–50 discovered by GeneProgram++ in the infection data. See Figure 4-7 containing EPs 1–25 for an explanation of the matrix; see also Figures 4-9 and 4-10 for the remaining EPs.

are strongly induced or repressed during infection via NF-κB family member regulatory activity. Examples of such programs are discussed in Section 4.3.5.

4.3.5 The generality score naturally categorized programs into a spectrum of responses to infection

We used the generality score to organize the discovered expression programs. Below we discuss representative examples of programs with low, intermediate and high generality scores. We focus the discussion on biological processes and pathways in which the programs are involved.

Programs with low generality scores were used by experiments spanning a limited number of host-cell and infection types

Experiments involving exposure of gastric epithelial cells to *H. pylori* used several low generality expression programs (EPs). This is biologically plausible, because gastric epithelial cells are not involved in principle immune system functions, unlike all other host-cells profiled in the data set [80]. As an example of a program used exclusively by *H. pylori* infected epithelial cells, EP 22 (generality = 0.011, 5 experiments) was enriched for genes involved in regulation of the actin cytoskeleton (KEGG: hsa04810), and was used with a middle induction modifier by all associated experiments. The induction of this pathway is consistent with extensive host-cell shape changes known to occur in *H. pylori* infection; delayed induction likely reflects the time necessary for bacterial attachment and secretion of proteins that induce host-cell cytoskeletal rearrangements [80].

Peripheral blood mononuclear cell (PBMC) and whole blood experiments also used several low generality programs. This is biologically reasonable, because PBMCs and whole blood represent mixtures of innate and adaptive immunity-mediating cell types, and thus contain cell types not profiled in the other experiments analyzed. Further, the diversity of cell types in PBMC and whole blood cultures allows for critical interactions that are necessary to trigger certain cellular responses [33]. As an example, EP 33 (generality = 0.027, 9 experiments) was used by experiments involving PBMCs and whole blood exposed to the Gram-negative bacteria *N. meningitides* or *B. pertussis*, and was enriched for genes with anti-apoptotic function (GO: 0006916). We hypothesize that this program, which was generally induced in the middle of the time-courses, may involve stabilization of an anti-apoptotic state necessary for maturation and differentiation of peripheral immune cells following infection.

		51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
A	m0 control1		M-								M+					M-										
	m0 control2		E-			E+						E-											E+			
	m0 control3				E+				E+			M-						M-		E-		E-				E-
	m0 control4	E-					E+					M-				E-										
	m0 Latex		E-								M+			E+					M+				E-			
	m0 E. coli		E-	M+							M+															
	m0 EHEC							E+		E+	E-	E-			E-			E-	E-	E-		E-				E-
	m0 S. typhi		E+					E+			M+	M-		E-		E+		E-	E-	E-	M-					
	m0 S. typhimirium										M+	E-		E-				E-	E-			E-				E-
	m0 S. aureus		E-					E+				E-			E-	E-		E-		E-						E-
	m0 L. monocytogenes							E+		M+	E-	M-			E-			M-	M-	M-	M-	E-				E-
	m0 M. tuberculosis		E-									E-														
	m0 M. bovis BCG		M-								M+															
	m0 LPS E			E+								E-		E-	E-			E-								
	m0 LPS S		E-	E+		E-								E-												
	m0 LTA				E-	M+				M+	E+	M-				M+		M-			M-	E+		M+		E+
	m0 MDP	E-				E-	E+					M-		E-		E-										
	m0 TB hsp70				E+	E+		M+	M+	E+	E-	E-						E-	E-	E-	E-	E-	E+			E-
	m0 BCG hsp65				E+			E+	E+	M+					M-			M-				E-				E-
	m0 MPA		E-	E+		E-		M+		E+				E-												
	m0 f-MLP	E-					E+					E-		E-		E-										
	m0 Protein A	E-				E-	E+				M-			M-		E-										
	m0 Mannose	E-				E-	E+				M-	E-		E-		E-										
	m0 IFN-alpha				E+	E+			E+							E+		E+	E-		E-	E-	E+	M+		
	m0 IFN-beta				M+	E+			E+	E+						E+			E-	E-	E-	E-	E+	E+		
	m0 IFN-gamma				E+	E+			M+	M+						E+		E+	E-		E-	E-	E+	E+	M-	
	m0 IL-10				E+	E+				E+						E+		E+	E-		E-	E-	E+	E+		
	m0 IL-12		E-												E-			M+	E-	E-	M-	E-	E+		E-	E-
	DC control1				E+				E+	E+								E-	E-	E-	E-		E+	E+		E+
	DC control2															E+			E-	E+		E-	E+	L+		E-
	DC control3				E-				E+	E+	E-						E+	E-	E-		E-	E-	E+		L-	M-
B	m0 (U937) control1																									
	m0 (U937) B. pertussis 338																									
	m0 (U937) B. pertussis 537 (avirulent)																									
	m0 (U937) B. pertussis A2-6 (AC-)																									
	m0 (U937) B. pertussis Tox6 (PT-)																									
	PBMC control1																									
	PBMC B. pertussis LPS												E+													
	PBMC B. pertussis 338												M+													
	PBMC B. pertussis Minnesota1																									
	PBMC E. coli																									
	PBMC S. aureus (1)												M+													
	PBMC S. aureus (2)												M+													
	PBMC Ionomycin+PMA												M+													
C	DC E. coli																E-					L+				M+
	DC LPS																									
	DC Influenza								L+								E-					L+				M+
	DC C. albicans																E-					L+				M+
	DC Mannan																E-									
	DC PolyIC				E-					E+		E-											E-	E-		
D	PBMC B. pertussis 338 Killed																									
	PBMC B. pertussis 338 Live																									
	WB control NP																									
	WB control SW																									
	WB N. meningitidis NP																									
	WB N. meningitidis SW																									
E	Epithelial cell control																									
	Epithelial cell H. pylori G27																									
	Epithelial cell H. pylori cagN-																									
	Epithelial cell H. pylori cagA-																									
	Epithelial cell H. pylori cagE-																									
	Epithelial cell H. pylori PAI-																									

|0.135 |0.225 |0.201 |0.262

Figure 4-9: Summary of expression programs (EPs) 51–75 discovered by GeneProgram++ in the infection data. See Figure 4-7 containing EPs 1–25 for an explanation of the matrix; see also Figures 4-8 and 4-10 for the remaining EPs.

Programs with intermediate generality scores were used by experiments involving host-cells exposed to a wider variety of agents

Several intermediate generality programs appeared to represent coordinated down-regulation of proteolytic and antigen presentation pathways. For example, EP 68 (generality = 0.227, 14 experiments) was used by experiments involving exposure of primary macrophages to a variety of interleukins/interferons, pathogens or their components. This program was enriched for genes involved in proteasome function (KEGG: hsa03050), and was generally repressed early in the time-series. As another example, EP 77 (generality = 0.298, 6 experiments) was used by several experiments involving PBMCs or cell-culture derived macrophages exposed to different bacteria or immune modulating chemicals. This program was enriched for genes involved in both MHC I and MHC II antigen processing and presentation pathways (KEGG: hsa04612), and was used with a middle repression modifier by all associated experiments. Downregulation of proteasome and antigen presentation pathways subsequent to infection may reflect commitment of phagocytic cells to presentation of antigens from a pathogen that has just been encountered [33].

Several intermediate generality expression programs revealed differences in temporal phasing of the response of host cells exposed to different classes of pathogenic organisms. For example, EP 88 (generality = 0.386, 13 experiments) was enriched for genes involved in ribosomal structure or function (KEGG: hsa03010). Induction of ribosomal genes may be a prelude to production of critical signaling and defensive proteins, such as chemokines and cytokines. EP 88 was induced early in macrophages or dendritic cells exposed to several varieties of Gram-negative bacteria, but induced in the middle of the time-series in host cells exposed to Gram-positive bacteria. As another example, EP 91 (generality = 0.402, 13 experiments), was enriched for genes involved in mRNA splicing (GO: 0000398) and oxidative phosphorylation (KEGG: hsa00190). This program was induced early in time-courses in dendritic cells exposed to bacteria or viruses, but not induced until the middle phase in experiments involving dendritic cells exposed to live fungi or fungal cell components. Interestingly, the program was repressed early in macrophages exposed to various interferons/interleukins. We hypothesize that the phasing differences observed in induction of EPs 88 and 91 may be due to the lesser ability of Gram-positive or fungal organisms to induce critical signaling pathways in innate immune cells. We also note that both programs were significantly enriched for genes bound by NF-κB family members. Interestingly, a number of the NF-κB targets in EP 88 were ribosomal genes, suggesting a direct role for this transcription factor in ribosome activity induction.

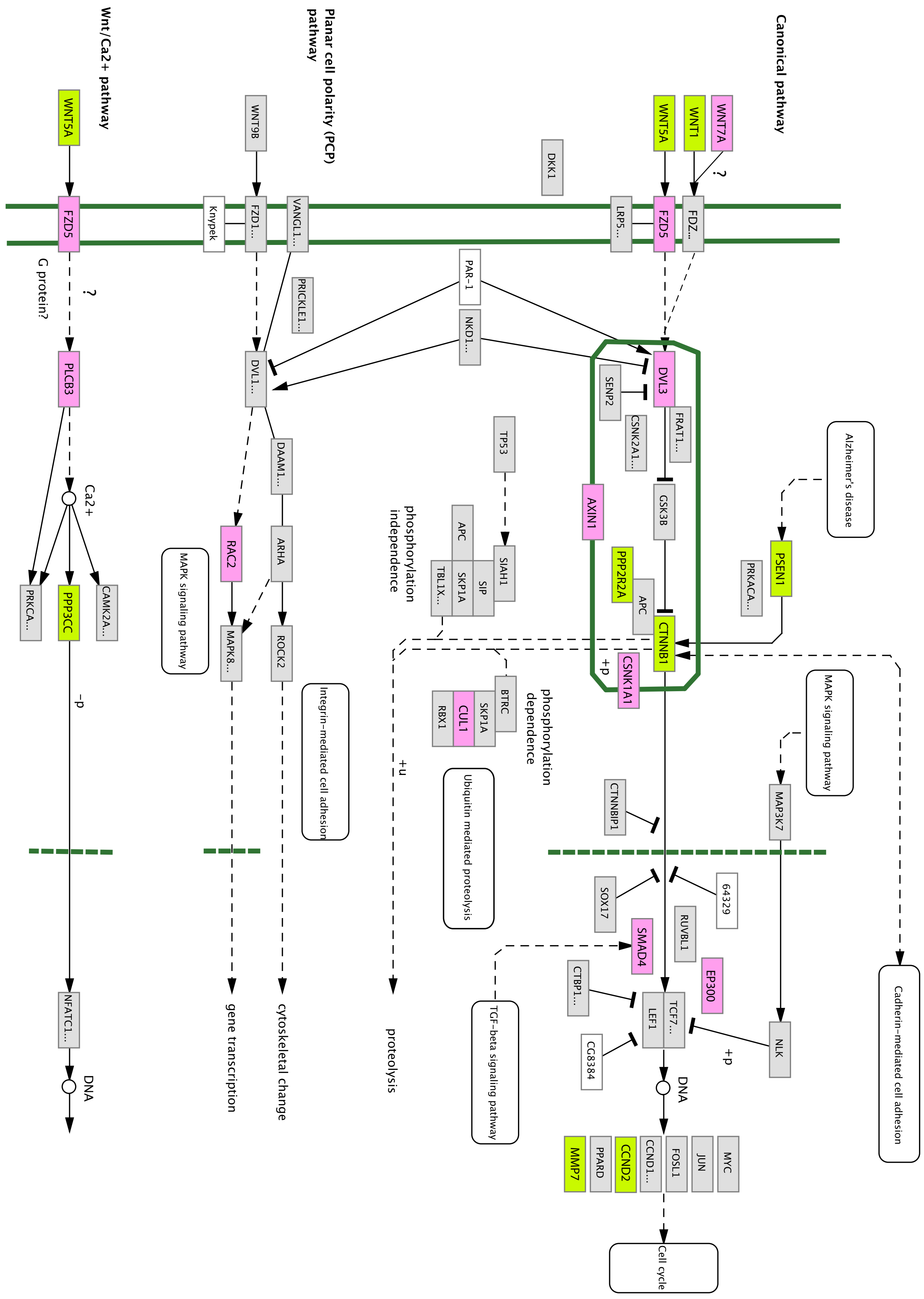
Several intermediate generality programs were significantly enriched for surprising signaling pathways or host-cell receptor types.

		76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104
A	m0 control1			M-								M+			M+															
	m0 control2											M+																		
	m0 control3	E+			E+			E-						E+		E-										E-				
	m0 control4																													
	m0 Latex											M+		M-	E+															
	m0 E. coli	M-					M-			M+		M+													E+			M+	M+	M+
	m0 EHEC						M-	M-		M+				E+								M-			E+			M+	M+	
	m0 S. typhi	E-					M-	M-		M+												M-			E+		M-	M+	E+	M+
	m0 S. typhimirium	M-					M-	M-		M+						E-						M-			E+		M-	M+		M+
	m0 S. aureus			E-			M-	M-		M+				M+								M-			E+				M+	M+
	m0 L. monocytogenes						M-	M-		M+				M+								M-			E+				M+	
	m0 M. tuberculosis						M-																		E+				M+	
	m0 M. bovis BCG			M-			M-			M+		M+			M+							M-			E+				M+	M+
	m0 LPS E	M-					M-																		E+			M+	M+	
	m0 LPS S						M-			M+															E+			M+	M+	
	m0 LTA	M-		M+													E+	E+							E+				E+	
	m0 MDP																								E+					
	m0 TB hsp70				E+		M-	E-		M+				E+		E-						M-			E+					
	m0 BCG hsp65				E+		M-	E-						E+											E+	E-			M+	
	m0 MPA																													
	m0 f-MLP																													
	m0 Protein A																													
	m0 Mannose	E+																												
	m0 IFN-alpha	E-		E+	E+		E-		E+							E-	E-	E-						M-				M+	E+	
	m0 IFN-beta	E-		E+	E+		E-		E+							M-	E-	M-						E-	E+			M+		
	m0 IFN-gamma	E-		E+	E+		E-		E+				M+			E-	E-	E-						E-						
	m0 IL-10	M-		E+	E+		E-		E+							E-	E-	E-						E-						
	m0 IL-12			E-												M-	E-	E-												
	DC control1	E-					M-		E-						E-						E-				E+	M-	M-			
	DC control2	E+		E+										E+		E+	E-	M-	M-	M+	L-			M-		L-	E+			
	DC control3			E+		L-			L-		L+			E+						M+										
B	m0 (U937) control1																													
	m0 (U937) B. pertussis 338		M-																				E+							M+
	m0 (U937) B. pertussis 537 (avirulent)																						E+							M+
	m0 (U937) B. pertussis A2-6 (AC-)																						E+							M+
	m0 (U937) B. pertussis Tox6 (PT-)																						E+							M+
	PBMC control1																													
	PBMC B. pertussis LPS																						E+							
	PBMC B. pertussis 338		M-																				E+							
	PBMC B. pertussis Minnesota1		M-																				E+							
	PBMC E. coli		M-																				E+							
	PBMC S. aureus (1)																						E+							
	PBMC S. aureus (2)		M-																				E+							
	PBMC Ionomycin+PMA		M-																				E+							
C	DC E. coli	M-		E-	L-			M-	L+	M+	L-	L+		E+		L+	E+	M+	L+	M-	L+	M-		E+	E+	E+	M-	M+	E+	M+
	DC LPS	L+		E-						M+				M+		L+	E+	M+	L+	M-	L+			M+	E+	M+	L+	M+	E+	
	DC Influenza			E-					L+		L-			E+		M+	E+	M+	L+	M-	L+			E+	E+	E+	M-	M+	M+	
	DC C. albicans										L-	L+					M+					L-	M+	M+				M+	M+	M+
	DC Mannan						M-	M-		M+							M+	M+	M+	M-		M-		M+	E+		M-	M+	E+	M+
	DC PolyIC			M-					E+					E+	E+	M+	M+	M+			E+			E+	M+	E+	E+	M+	M+	
D	PBMC B. pertussis 338 Killed																								E+			M+	M+	
	PBMC B. pertussis 338 Live					E-																			E+				M+	
	WB control NP					M+																			M+					
	WB control SW					M+																			M+					
	WB N. meningitidis NP																								M+				M+	
	WB N. meningitidis SW																								E+				M+	
E	Epithelial cell control												E+																	
	Epithelial cell H. pylori G27												M+																	
	Epithelial cell H. pylori cagN-																													
	Epithelial cell H. pylori cagA-																													
	Epithelial cell H. pylori cagE-												M+																	
	Epithelial cell H. pylori PAI-												E+																	
		0.294							0.435							0.401							0.535							0.727

Figure 4-10: Summary of expression programs (EPs) 76–104 discovered by GeneProgram++ in the infection data. See Figure 4-7 containing EPs 1–25 for an explanation of the matrix; see also Figures 4-8 and 4-9 for the remaining EPs.

For instance, EP 50 (generality = 0.134, 13 experiments) and EP 84 (generality = 0.331, 12 experiments) were significantly enriched for genes involved in the Wnt signaling pathway (KEGG: hsa04310), including WNT7A and FZD5 in EP 50, and WNT1, WNT5A and MMP7 in EP 84 (see Figure 4-11). Wnt signaling pathways have been traditionally implicated in developmental processes [93], and have only recently been shown to be involved in immune system functions [32, 129, 168]. For instance, Lobov *et al.* demonstrated that macrophages can secrete WNT7B, which induces apoptosis in vascular endothelial target cells via the canonical Wnt signaling pathway [168]. Signaling via the WNT5A receptor FZD5 has been implicated in stimulation of pro-inflammatory molecules (e.g., MMP7, TNF-α, IL-12) in macrophages, possibly via both canonical and non-canonical pathways [32, 168]. Consistent with these reports of Wnt activity in macrophages, EP 50 was used by macrophages exposed to a variety of bacteria and stimulatory molecules. Interestingly, the program was repressed in macrophages infected with bacteria and induced in cells treated with interleukins or interferons. This difference in program usage may reflect the ability of bacteria to downregulate pro-inflammatory Wnt pathways. In contrast, EP 84, which was mostly used by macrophages and dendritic cells exposed to bacteria or microbial components, was uniformly induced in the middle of the infection time-series. Because the two programs contain different Wnt pathway genes, they may be involved in different inflammatory functions. Further, we note that only some of the Wnt pathway associated genes in EPs 50 and 84 have previously been implicated in macrophage function, making these attractive candidates for future experimental biology work.

EP 55 (generality = 0.161, 12 experiments), which was significantly enriched for genes coding for neurotransmitter or hormonal receptors (KEGG: hsa04080), was another surprising finding. This program was induced in macrophages treated with various interleukins or interferons, and repressed in macrophages exposed to various microbial components. The program contained genes coding for a variety of receptors including those for acetylcholine (CHRM5), cannabinoids (CNR2), dopamine (DRD2) and histamine (HRH1). Although such receptor types are typically found on neurons, they have also been found on macrophages and T-cells, and recent studies suggest they may have important pro- and anti-inflammatory properties [69, 156, 208, 213]. The different use of this program by macrophages exposed to interleukins/interferons (induction) versus that of those exposed to bacterial components (repression) may reflect bias toward pro- or anti-inflammatory states mediated by different neuroactive receptor signaling pathways.



Canonical pathway
Planar cell polarity (PCP) pathway
Wnt/Ca2+ pathway
WNT7A
WNT1
WNT5A
?
DKK1
FDZ...
FZD5
LRP5...
VANGL1...
FZD1...
Knypek
WNT9B
FZD5
PAR-1
NKD1...
PRICKLE1...
G protein?
?
DVL3
SENP2
CSNK2A1...
FRAT1...
GSK3B
AXIN1
PPP2R2A
APC
CTNNB1
CSNK1A1
+p
DVL1...
PLCB3
Ca2+
DAAM1...
ARHA
ROCK2
RAC2
MAPK8...
MAPK signaling pathway
PRKCA...
PPP3CC
CAMK2A...
-p
NFATC1...
DNA
TP53
SIAH1
SIP
SKP1A
APC
TBL1X...
phosphorylation independence
BTRC
SKP1A
CUL1
RBX1
phosphorylation dependence
+u
Ubiquitin mediated proteolysis
Integrin-mediated cell adhesion
Alzheimer's disease
PSEN1
PRKACA...
MAPK signaling pathway
MAP3K7
CTNNBIP1
SOX17
64329
RUVBL1
SMAD4
EP300
CTBP1...
TCF7...
LEF1
CG8384
NLK
+p
DNA
TGF-beta signaling pathway
Cadherin-mediated cell adhesion
MYC
JUN
FOSL1
CCND1...
CCND2
PPARD
MMP7
Cell cycle
proteolysis
gene transcription
cytoskeletal change

Figure 4-11: The Wnt signaling pathway (KEGG: hsa04310) and overlapping genes in EP 50 (shaded in pink) and EP 84 (shaded in green). The pathway graphic was adapted from the KEGG database [109]. EP 50 was used by macrophages exposed to a variety of bacteria and stimulatory molecules. The program was repressed in macrophages infected with bacteria and induced in cells treated with interleukins or interferons. EP 84 was mostly used by macrophages and dendritic cells exposed to bacteria or microbial components, and was uniformly induced in the middle of the infection time-series.

Programs with high generality scores were used by the full spectrum of experiments involving exposure of different host cell types to a wide variety of agents

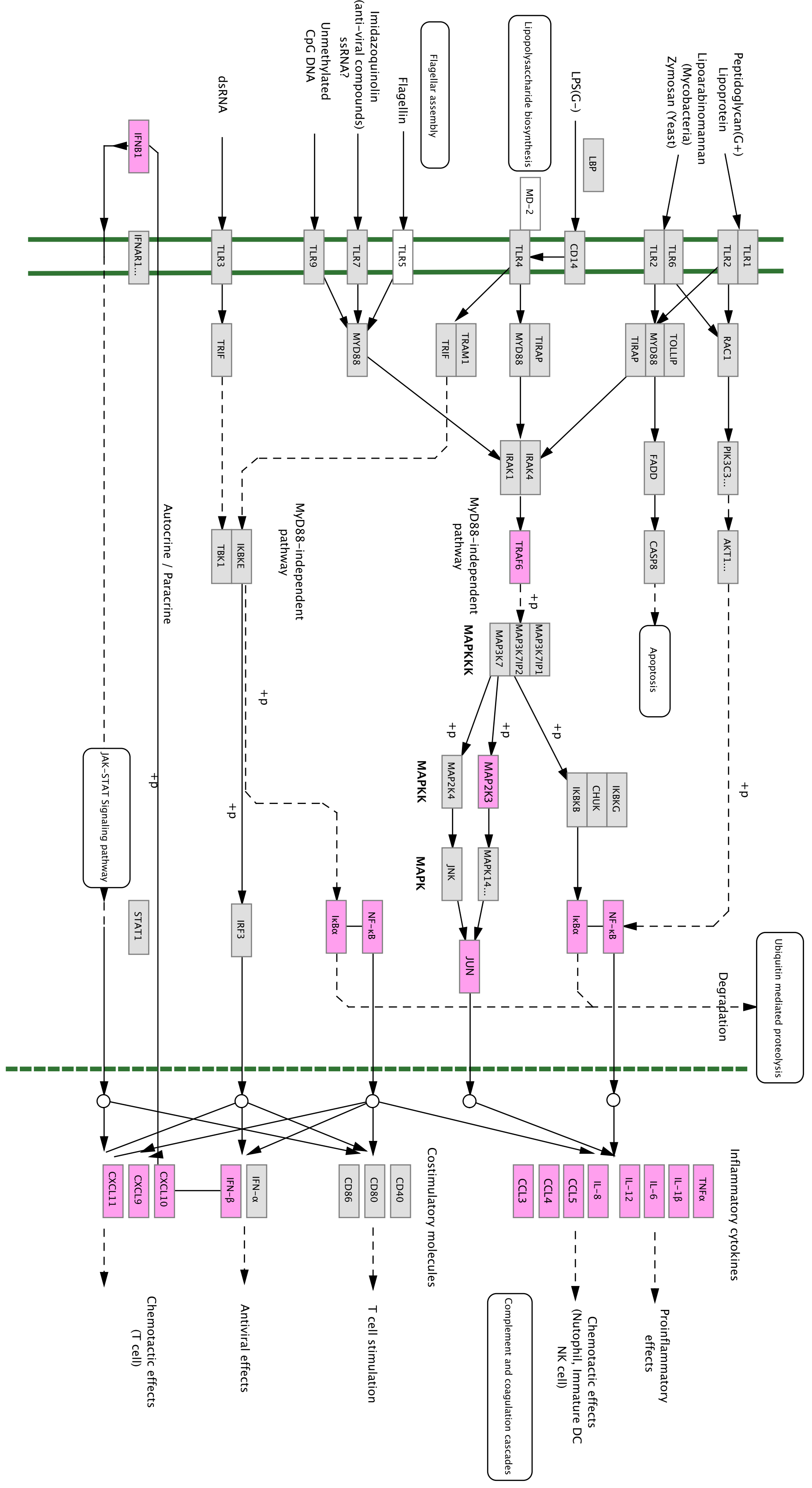
Most programs with high generality scores were enriched for genes bound by NF-κB family members and involved in a range of signaling pathways.

For instance, EP 99 (generality = 0.585, 27 experiments) appeared to represent a "common infection response" program, which was used by experiments from every tissue group and almost always induced early. This EP contained 203 genes, 15% of which code for cytokines or cytokine receptors, and was also significantly enriched for a number of signaling pathways. Of particular interest, it contained many genes involved in the Toll-like receptor signaling pathway (KEGG: hsa04620, see Figure 4-12). As expected, many of the genes in the overlap between EP 99 and this pathway code for cytokines, but a number also code for various downstream signaling molecules, including TRAF6, AP-1, NF-κB (p105) and IκBα. Further, EP 99 was significantly enriched for genes bound by NF-κB family members RELB, p65, p50, p52 or c-REL in ChIP-chip experiments.

In contrast, EP 102 (generality = 0.634, 15 experiments) was also used by a wide variety of experiments, but was induced in the middle of the time-series of all associated experiments, and was not significantly enriched for binding of any NF-κB family members. This program contained a large number of genes coding for interferons or interferon induced chemokines [107]. Interestingly, many interferon sensitive genes (ISGs) are activated by transcription factors other than NF-κB, and a delay in production of ISGs has previously been noted, presumably due to the time needed to establish critical autocrine and paracrine signaling loops [107].

4.4 Conclusion and discussion

In this chapter, we presented GeneProgram++, an extension of our original algorithm that explicitly models general patterns of expression changes, such as induction and repression or temporal dynamics. We achieved this through the innovation of program *usage modifiers*, which are variables that alter the context in which each tissue uses an expression program. We first developed a simplified model without tissue groups that uses modifiers, and then presented the full GeneProgram++ model and described an



Peptidoglycan(G+)
Lipoprotein
Lipoarabinomannan
(Mycobacteria)
Zymosan (Yeast)
LPS(G−)
LBP
Lipopolysaccharide biosynthesis
MD−2
Flagellar assembly
Flagellin
Imidazoquinolin
(anti−viral compounds)
ssRNA?
Unmethylated
CpG DNA
dsRNA
IFNB1
TLR1
TLR2
TLR6
CD14
TLR4
TLR5
TLR7
TLR9
TLR3
IFNAR1...
RAC1
TOLLIP
MYD88
TIRAP
TRAM1
TRIF
PIK3C3...
AKT1...
FADD
CASP8
Apoptosis
IRAK4
IRAK1
TRAF6
MyD88−independent pathway
MAPKKK
MAP3K7IP1
MAP3K7IP2
MAP3K7
+p
IKBKG
CHUK
IKBKB
MAP2K3
MAP2K4
MAPKK
MAPK14...
JNK
MAPK
IκBα
NF−κB
JUN
IKBKE
TBK1
MyD88−independent pathway
Autocrine / Paracrine
JAK−STAT Signaling pathway
STAT1
IRF3
Ubiquitin mediated proteolysis
Degradation
Inflammatory cytokines
TNFα
IL−1β
IL−6
IL−12
IL−8
CCL5
CCL4
CCL3
Costimulatory molecules
CD40
CD80
CD86
IFN−α
IFN−β
CXCL10
CXCL9
CXCL11
Proinflammatory effects
Chemotactic effects (Nutophil, Immature DC NK cell)
Complement and coagulation cascades
T cell stimulation
Antiviral effects
Chemotactic effects (T cell)

Figure 4-12: The Toll-like receptor signaling pathway (KEGG: hsa04620) and overlapping genes in EP 99 (shaded in pink). The pathway graphic was adapted from the KEGG database [109]. EP 99 appeared to represent a “common infection response” program, which was used by experiments from every tissue group and almost always induced early. This EP contained 203 genes, 15% of which code for cytokines or cytokine receptors, and was also significantly enriched for a number of signaling pathways including the Toll-like receptor pathway depicted here. As expected, many of the genes in the overlap between EP 99 and this pathway code for cytokines, but a number also code for various downstream signaling molecules, including TRAF6, AP-1, NF-κB (p105) and IκBα. Further, EP 99 was significantly enriched for genes bound by NF-κB family members RELB, p65, p50, p52 or c-REL in ChIP-chip experiments.

efficient MCMC sampling method for approximate model inference. We also described additional improvements relating to the interpretability and efficiency of posterior distribution summary methods. We used two large compendia of expression data to show that GeneProgram++ outperformed popular biclustering algorithms to an even greater extent than did the original GeneProgram algorithm.

We then took advantage of GeneProgram++'s ability to find coherent gene sets used with different temporal dynamics by each tissue sample—a capability traditional biclustering algorithms do not have. We applied GeneProgram++ to a compendium of short time-series gene expression data sets exploring the responses of human host-cells to infectious agents and immune-modulating molecules. Using this data set, GeneProgram++ discovered 5 tissue groups and 104 expression programs, a substantial number of which were significantly enriched for genes involved in key signaling pathways and/or bound by NF-κB transcription factor family members in ChIP-chip experiments. We used the generality score to characterize the functional spectrum of discovered expression programs—from gene sets involved in response to specific pathogens in one host cell type, to those mediating common inflammatory pathways.

GeneProgram++ automatically discovered many expression programs involved in key pathways related to the response to infection and uncovered temporal phasing differences in program use by some experiments. For instance, programs enriched for genes involved in ribosomal function or energy production were induced earlier in host-cells exposed to Gram-negative organisms than in those exposed to Gram-positive organisms or fungi. Some of the discovered programs overlapped with previously described gene sets derived from the same expression data, such as EP 99 and the “common host response” genes discussed by Jenner and Young [107]. However, previous meta-analyses of the data compendium relied on extensive prior biological knowledge and manual inspection of data [107]. In contrast, our method was automatic, discovering expression programs, associating them with consistent temporal patterns, and finding significantly overlapping biological pathways and NF-κB binding from ChIP-chip data.

Some of the gene sets discovered by GeneProgram++ implicated surprising signaling pathways or host-cell receptor types in the response to infection. In particular, EPs 50 and 84 were significantly enriched for genes involved in the Wnt signaling pathway, and EP 55 was significantly enriched for genes coding for neurotransmitter or hormonal receptors. Wnt signaling pathways [32, 129, 168] and neurotransmitter receptors [69, 156, 208, 213] have only recently been implicated in the response to infection. To our knowledge, our work is the first to uncover the activity of these pathways in the data sets analyzed, and to characterize the different temporal behaviors of these pathways in response to a variety of infectious agents and immunomodulatory molecules. We believe that the genes in the above-mentioned expression programs constitute particularly attractive candidates for further biological characterization.

GeneProgram++'s handling of temporal data by collapsing time-series into predefined, discrete patterns differs from the approach taken by many previous analysis methods, and so merits some further discussion. Overall, we believe that our approach is very useful for finding interpretable gene expression programs, particularly when analyzing short time-series experiments in which there are a limited number of clearly meaningful temporal patterns. Further, our method allows us to extract features present in a compendium of time-series, even when the series have different durations, sampling rates, and numbers of samples. In the case of the infection time-series data analyzed in this work, we defined relevant temporal patterns manually, based on prior biological knowledge [33, 91, 149, 150]. However, in future work, temporal patterns could be derived in more automated ways, such as through pre-processing steps that apply time-series clustering algorithms [17, 59] to individual series in the data compendium.

CHAPTER 5

Conclusion

"Before reaching the final line, however, he had already understood that he would never leave the room, for it was foreseen that the city of mirrors (or mirages) would be wiped out by the wind and exiled from the memory of men at the precise moment when Aureliano Babilonia would finish deciphering the parchments, and that everything written on them was unrepeatable since time immemorial and forever more, because races condemned to one hundred years of solitude did not have a second opportunity on earth."

—*Gabriel Garcia Marquez, One Hundred Years of Solitude*

We have presented three novel computational approaches and have shown that each finds biologically meaningful sets of genes in large collections of high-throughput molecular data. In the remainder of this thesis, we summarize the main contributions made in each chapter and then conclude with a discussion of directions for future work.

5.1 The GRAM algorithm

5.1.1 Summary of results

In Chapter 2, we presented the GRAM (Genetic RegulAtory Modules) algorithm, a method for fusing information from genome-wide expression and *in vivo* transcription factor-DNA binding data sets to discover regulatory networks of gene modules, which are sets of genes that are both co-expressed and bound by the same set of transcription factors. We used the GRAM algorithm to discover a genome-wide regulatory network using binding information for 106 transcription factors in *Saccharomyces cerevisiae* in rich media conditions and over 500 expression experiments. We validated the quality of these results by performing analyses using four independent data sources. We then used the discovered modules to label transcription factors as activators or repressors and identified patterns of combinatorial regulation. Further, we analyzed a new genome-wide protein-DNA binding data set profiling transcription factors in yeast cells treated with rapamycin, and used the GRAM algorithm to provide biological insights in this regulatory network. Finally, we presented a method for using modules to build automatically genetic regulatory sub-networks for specific biological processes, and used this to reconstruct accurately key elements of the cell-cycle in yeast.

5.1.2 Specific computational and biological contributions

The GRAM algorithm was the first published method for fusing genome-wide expression data and large collections of transcription factor binding data. Unlike previous approaches [58, 162, 99, 26, 191] that relied primarily on *functional information* from expression data, the GRAM algorithm explicitly links genes to the factors that regulate them by using DNA binding data to incorporate *direct* physical evidence of regulatory interactions.

Fusion of genome-wide expression and binding data is computationally challenging for several reasons. First, expression data is high-dimensional, and the GRAM algorithm must handle continuous data for thousands of genes measured in hundreds of experiments. Second, each module is potentially regulated by any combination of over one hundred transcription factors. Finally, both genome-wide binding and expression data are measured experimentally using DNA microarrays, which produce notoriously noisy output.

To address these challenges, we made several specific *computational contributions* with the GRAM algorithm:

- **Efficient, robust determination of nearby genes in the high-dimensional expression space.** The GRAM algorithm uses a theoretically justified approximation technique inspired by computational geometry methods for finding the largest set of “nearby” genes in expression space. This method is robust in the sense that it is insensitive to co-bound genes with outlying expression measurements. To our knowledge, the GRAM algorithm is the first use of such a method in the context of expression data analysis.

- **Efficient search over combinations of transcription factors.** As discussed, there is a potentially exponential number of combinations of transcription factors to be considered as regulators for each gene module. However, the GRAM algorithm efficiently restrict its search to combinations “confidently” implied by the data, rendering the number of subsets to be searched over much smaller. This innovation enabled the algorithm to operate on a data set containing genome-wide binding data for over 100 transcription factors.

- **Improvement of true-positive rates without significant increases in false-positive rates in genome-wide binding data.** A key feature of the GRAM algorithm is its ability to compensate for technical limitations in genome-wide binding data through the integration of expression data. To determine binding events in data, previous methods [118] used a statistical model and chose a relatively stringent p-value threshold with the intention of reducing false-positives at the expense of true-positives. The GRAM algorithm presents a powerful alternative to using a single p-value threshold to predict binding events, because our method allows the p-value cutoff to be relaxed if there is sufficient supporting evidence from expression data. Approximately 40% of binding events discovered by the GRAM algorithm would not have been detected using only genome-wide binding data and the stringent p-value cutoff. We used four independent sources of biological information, including gene-specific chromatin-immunoprecipitation experiments, to demonstrate that the newly predicted binding events did not substantially increase the false-positive rate.

We made several *biological contributions* with our applications of the GRAM algorithm to genome-wide yeast gene expression and transcription factor binding data:

- **The first genome-wide regulatory network in yeast derived from direct physical information.** Using genome-wide binding data for over 100 yeast transcription regulators profiled in rich media conditions, the GRAM algorithm discovered 106 gene modules, containing 655 distinct genes and regulated by 68 of the transcription factors. Although previous studies derived regulatory networks from large-scale data sources in yeast, these methods did not incorporate large compendia of genome-wide binding data [162, 99]. We demonstrated that our results represent a refinement of the modules from these other studies. In particular, our method identified not only the genes that participate in a certain module, but also provided evidence as to the factors that are used to activate these genes.

- **Biological insights from construction of a regulatory network from novel genome-wide transcription factor binding data profiling yeast exposure to rapamycin.** Rapamycin is a small macrolide that mimics nutrient starvation and has recently been investigated for the treatment of malignancies and heart disease [104, 49, 172]. Prior to our work, there was little information about the transcriptional regulatory network involved and how this transcriptional network contributed to the overall response to rapamycin treatment. Our biologist collaborators identified 14 key transcriptional regulators involved in the yeast response to rapamycin treatment and performed genome-wide binding experiments for the regulators. Our analysis suggested several unexpected interactions in which regulators typically assigned to a particular biological response also appear to bind genes acting in different biological pathways. In addition, our analysis suggested more complex regulatory interactions in which there was communication among gene modules. Such complicated network topologies may be important for facilitating rapid and flexible responses to changing environmental conditions. These results provide suggestions for future directed biological experiments.

- **Automatic, accurate recovery of key elements of the yeast cell-cycle regulatory network.** We developed a novel sub-network discovery algorithm that drew on our previous work involving continuous representations of time-series data [18] and applied it to analysis of the yeast cell-cycle. Our method correctly identified cell-cycle associated transcriptional regulators, and assigned all to stages of the cell-cycle in which they had been described to function in previous studies [199]. Significantly, this accurate reconstruction of the regulatory architecture was automatic and required no prior knowledge of the regulators that control transcription during the cell-cycle. Further, our analysis suggested that combinatorial factor interactions may provide control that allows for subdividing cell cycle phases into different biological functions.

5.2 The GeneProgram algorithm

5.2.1 Summary of results

In Chapter 3, we presented a new computational methodology, GeneProgram, specifically designed for analyzing large compendia of mammalian expression data. Through synthetic data experiments, we showed that GeneProgram was able to correctly recover gene sets that other popular analysis methods could not. We then applied our method to a large compendium of human and mouse body-wide gene expression data from representative normal tissue samples, and demonstrated that GeneProgram outperformed other methods in the discovery of biologically interpretable gene sets. We further showed that allowing the GeneProgram model to infer tissue groups automatically significantly improved performance. Using the data compendium, GeneProgram discovered 19 tissue groups and 100 expression programs active in mammalian tissues. We introduced an expression program generality score that exploits the tissue groupings automatically learned by GeneProgram, and showed that this score characterizes the functional spectrum of discovered expression programs.

5.2.2 Specific computational and biological contributions

GeneProgram addresses an important research problem in computational biology: the identification of *expression programs*, sets of co-activated genes orchestrating physiological processes, and the characterization of the functional breadth of these programs. The use of mammalian expression data compendia for discovery of such programs presents several challenges, including cellular inhomogeneity within samples, genetic and environmental variation across samples, and uncertainty in the numbers of programs and sample populations.

Specific *computational contributions* from our GeneProgram work include:

- **First application of Hierarchical Dirichlet Process-based topic models to expression data analysis.** GeneProgram's probability model addresses the above-mentioned challenges associated with finding expression programs in large, complex compendia of mammalian expression data. A topic model-based framework allows GeneProgram to deal with tissue inhomogeneity by allocating the total mRNA recovered from each tissue to different gene expression programs, which may be shared across tissues. A hierarchical model allows GeneProgram to deal with tissue samples coming from different individuals, by explicitly modeling each tissue as a sample from a population of related tissues. Uncertainty in the numbers of tissue groups and expression programs is handled in an integrated, principled manner by using Dirichlet Processes, which provide prior distributions over numbers of sets. To our knowledge, GeneProgram is the first model to incorporate all these features.

- **Extension of the Hierarchical Dirichlet Process model for automatic learning of tissue groups.** In the original formulation of Hierarchical Dirichlet Processes (HDPs), the number of data groups was fixed and needed to be specified *a priori* [211]. GeneProgram extends the HDP model, allowing the number of groups and membership in them to be learned automatically. This is accomplished by employing another Dirichlet Process to generate tissue groups. We developed efficient MCMC update steps to make approximate inference feasible for this extended HDP model. Further, we used cross-validation to compare the full GeneProgram model to one without automatic inference of tissue groups, and demonstrated that the full model had significant performance improvements.

- **A novel method for summarization of the model posterior probability distribution.** The posterior distributions of Dirichlet Process mixture models are particularly challenging to summarize because the number of mixture components may differ for each sample. Previous approaches for summarizing Dirichlet Process mixture model components have used pair-wise co-clustering probabilities as a similarity measure for input into an agglomerative clustering algorithm [139]. This method is not feasible for summarizing expression programs in large data sets because of the number of pair-wise probabilities that would need to be calculated for each sample. We developed a novel method for summarization of the model posterior distribution, which discovers recurrent expression programs by combining information from similar expression programs that reoccur across posterior samples.

- **Ability to accurately recover coherent gene sets in noisy synthetic data that other algorithms could not.** We created synthetic data that simulated important features of real microarray data profiling mammalian tissues, and evaluated the ability of our algorithm and several others to recover coherent gene sets from the data. We chose for evaluation several popular algorithms used for microarray data analysis including hierarchical clustering [58], Samba (a biclustering algorithm) [209, 194], singular value decomposition [8, 9] and a non-negative matrix factorization implementation [36]. We also evaluated a simplified version of GeneProgram without the capability for automatic tissue group discovery. Only the full version of GeneProgram was capable of accurately recovering all the gene sets present in the synthetic data.

We applied GeneProgram to a large compendium of data, consisting of genome-wide expression measurements for 79 human and 61 mouse tissues. Specific *biological contributions* from this application of GeneProgram include:

- **Automatic assignment of tissues to biologically relevant groups.** GeneProgram discovered 19 tissue groups, 79% of which were significantly enriched for tissues belonging to one of ten manually derived, broad physiological categories. For instance, a tissue group significantly enriched for the "hematological/immune" category consisted exclusively of human immune cells such as natural killer cells, and CD4+ and CD8+ T-cells. As another example, a tissue significantly enriched only for the "neural" category, consisted exclusively of neural tissues from both species. GeneProgram discovered these groups in a wholly unsupervised manner, and many of the groups clearly represent a more refined picture of the data than the ten manually derived categories.

- **Outperformance of biclustering algorithms in the discovery of biologically relevant gene sets.** Because expression programs characterize both genes and tissues, we used both Gene Ontology (GO) categories [11] and 10 manually derived tissue categories to assess GeneProgram's ability to recover biologically relevant gene sets and to compare this performance to that of two biclustering algorithms, Samba [209, 194] and a non-negative matrix factorization (NMF) implementation [36]. GeneProgram clearly outperformed the other two algorithms in the tissue dimension (60% of expression programs significantly enriched for tissue categories, versus 10% for Samba and 20% for NMF). GeneProgram outperformed NMF and had equivalent performance to Samba in the gene dimension (61% of expression programs significantly enriched for GO categories, versus 62% for Samba and 27% for NMF). Presumably, our algorithm's clear dominance of both Samba and NMF method in the tissue dimension was partly attributable to GeneProgram's hierarchical model. Both of the other algorithms lack such a model, so the assignment of tissues to biclusters was not guided by global relationships among tissues.

- **Outperformance of cross-species versus single-species expression programs in terms of biological relevance.** Seventy-nine percent of cross-species expression programs were significantly enriched for GO categories versus 52% of single-species programs, and 82% of cross-species programs were significantly enriched for the manually derived tissue categories versus 51% of single-species programs. These results suggest that combining data from both species was valuable for discovery of biologically relevant expression programs.

- **Introduction of a novel *generality score* that automatically quantified the functional specificity of expression programs.** We developed a score for assessing the functional generality of expression programs, and demonstrated its utility for automatically characterizing the spectrum of discovered programs. Based on the generality score, we divided expression programs into three broad categories: 1) general body-wide physiology, 2) specialized organ physiology, and 3) tissue specific. Evaluation of the weighted average of generality scores across all expression programs for each tissue uncovered several trends relating to tissue function and anatomic location. For instance, some tissues types, including neural, testicular and thyroid samples, had very low average generality scores, presumably reflecting the highly specialized functions of these tissues. In contrast, a number of other tissue types, including embryologic, hematologic progenitors, immune, malignant, epithelial and adipose samples, had very high average generality scores. The generality score requires a global organization of tissues into groups, rather than just the local associations of subsets of tissues with individual gene sets provided by biclustering algorithms. Because there is uncertainty in the number of tissue groups, GeneProgram's Dirichlet Process-based model provides a natural framework for computing the generality score.

- **Automatic discovery of a comprehensive, body-wide map of expression programs active in mammalian physiology.** By simultaneously using information across 140 tissue samples, GeneProgram was able to finely dissect the data, automatically splitting mRNA expressed in tissues among both general and specific programs. Because our model is fully Bayesian, providing a global penalty for model complexity including for the number of tissue groups and expression programs, the generated map represents a mathematically principled compression of gene expression information throughout the entire organism. Although such a large, comprehensive map is inherently complicated, we believe that GeneProgram's automatic grouping of tissues and the associated expression program generality score will make it particularly useful for guiding future biological experiments. Tissue-specific expression programs can provide candidate genes for diagnostic markers or drug targets that exhibit minimal "cross-talk" with unintended organs. General expression programs may be useful for identifying genes important in regulating and maintaining general physiological responses, which may go awry in disease states such as sepsis and malignancy. Both general and tissue-specific discovered programs contained many functionally unannotated genes, and in some cases the programs were shared among unexpected sets of tissues. Additionally, some such unannotated genes appear in cross-species expression programs, making them particularly attractive candidates for further biological characterization.

5.3 The GeneProgram++ algorithm

5.3.1 Summary of results

In Chapter 4, we presented GeneProgram++, an extension of the GeneProgram algorithm that explicitly models general patterns of expression changes, such as induction and repression or temporal dynamics. We described an efficient approximate inference scheme for the GeneProgram++ model as well as additional improvements relating to the interpretability and efficiency of posterior distribution summary methods. We used two large compendia of expression data to show that GeneProgram++ outperformed popular biclustering algorithms to an even greater extent than did the original GeneProgram algorithm. We then applied GeneProgram++ to a compendium of short time-series gene expression data sets exploring the responses of human host-cells to infectious agents and immune-modulating molecules. Using this data set, GeneProgram++ discovered 5 tissue groups and 104 expression programs, a substantial number of which were significantly enriched for genes involved in key signaling pathways and/or bound by NF-κB transcription factor family members in ChIP-chip experiments. We used the generality score to characterize the functional spectrum of discovered expression programs—from gene sets involved in response to specific pathogens in one host cell type, to those mediating common inflammatory pathways. GeneProgram++ automatically discovered many expression programs involved in key pathways related to the response to infection and uncovered temporal phasing differences in program use by some experiments. Of particular interest, some of the gene sets discovered by GeneProgram++ implicated surprising signaling pathways or host-cell receptor types in the response to infection.

5.3.2 Specific computational and biological contributions

We made several *computational contributions* in our work on GeneProgram++:

- **Introduction of the novel concept of program *usage modifiers* that explicitly model general patterns of expression changes including temporal dynamics.** In many microarray gene expression experiments, we are interested in genes' behavior relative to some baseline condition. For instance, we may be interested in the extent of induction or repression of gene expression after cells are exposed to environmental stresses [71], infected with microorganisms [202, 149, 150, 107, 33, 91, 160, 80], or observed throughout development [220]. In analyzing patterns of gene expression change, we would like to discover sets of genes that behave coherently. A limitation of the GeneProgram algorithm is that it does not explicitly model patterns of gene expression change. GeneProgram++ extends the algorithm with *usage modifiers*. A usage modifier is a variable that is specific to a tissue-expression program pair and describes how a tissue uses the program. For instance, usage modifiers can specify the temporal phase and direction (induction or repression) of expression. Thus, the genes used by a tissue from a program and the manner in which they are expressed (e.g., early induction versus late repression) are chosen probabilistically and influenced by the behavior of similar tissues. Further, usage is by definition consistent across a program for a particular tissue, which facilitates biological interpretation. We fully incorporated usage modifier variables into a hierarchical model based on Dirichlet Processes. We then developed efficient MCMC update steps that make approximate inference feasible for this model.

- **Improvements in recurrent expression program computation that increase algorithm performance and model interpretability.** GeneProgram's method for deriving recurrent expression programs (REPs) had two limitations. First, the original tissue REP usage score was difficult to compare across tissues and did not take into account expression probabilities for genes in a program. GeneProgram++ improves upon the original REP usage score by weighting by the probabilities of genes in the expression program. Second, the method used by GeneProgram to track REPs could lead to the creation of many REPs with essentially the same gene probabilities, but different "signatures" of significant tissues. Although the REPs could be merged in subsequent post-processing steps, having to track a large number of REPs substantially reduced the algorithm's performance and became infeasible for very large data sets. To deal with these issues, GeneProgram++ does not track REPs based on significant tissues, but instead saves all expression programs at each iteration and then sequentially merges similar programs based on how similar the gene expression probabilities are for programs. GeneProgram++ also uses spaced samples from the MCMC sampler, which better approximate independent samples from the posterior, and can thus result in more accurate results [72]. The improved method for merging REPs as well as the use of spaced samples allows us to use fewer overall samples than we did for the original GeneProgram applications. We used two large compendia of expression data to show that GeneProgram++ outperformed popular biclustering algorithms to an even greater extent than did the original GeneProgram algorithm in terms of the biological relevance of gene sets.

We applied GeneProgram++ to a compendium of 62 short time-series gene expression experiments in which various human cell types had been exposed to different infectious agents or immune-modulating substances. Specific *biological contributions* from this application of GeneProgram++ include:

- **Extensive overlap of automatically discovered expression programs with key human signaling pathways and biological processes.** GeneProgram++ discovered 104 expression programs in the infection time-series data. A large number of expression programs were significantly enriched for GO categories (50%) or KEGG pathways (59%). As expected, many significant GO categories and KEGG pathways were specifically involved with response to infection. Interestingly, a substantial number of the significantly enriched GO categories or KEGG pathways corresponded to signaling cascades with genes in expression programs spanning the levels of cascades, from genes coding for cell-membrane associated proteins to end effectors such as transcriptional regulators. Further, there were also a number of significantly enriched biological processes or pathways not directly labeled as being infection-related, but that are involved with significant changes in cellular physiology consistent with infection. To our knowledge, this application of GeneProgram++ represents the first large-scale, automated analysis using signaling pathways of a compendium of expression data profiling response of human cells to infection.

- **Surprising number of expression programs containing many genes bound by NF-κB transcription factor family members.** NF-κB family members are critical transcription factors in many immune cell types. We evaluated the biological relevance of discovered expression programs using genome-wide ChIP-chip data profiling static binding of NF-κB family transcription factors in human cell-culture derived macrophages [185]. Fifteen of the 104 expression programs discovered by GeneProgram++ were significantly enriched for sets of genes bound by at least one NF-κB family member. This overlap with the binding data was quite large, considering that only 348 genes were bound by NF-κB family members in the ChIP-chip experiments [185]. Overall, the fifteen significantly enriched programs tended to be those used by a diverse array of host-cell types exposed to a range of pathogens and their components. This suggests that these expression programs may represent common processes that are strongly induced or repressed during infection via NF-κB family member regulatory activity.

- **Automatic categorization using the generality score of discovered expression programs into a spectrum of responses to infection.** Programs with low generality scores were used by experiments spanning a limited number of host cell and infection types. For instance, experiments involving exposure of gastric epithelial cells to *H. pylori* used several low generality programs, some of which were clearly involved in specific responses to this infection (such as actin cytoskeleton reorganization). Programs with intermediate generality scores tended to be used by experiments involving host cells exposed to a wider variety of agents. For instance, several programs with intermediate generality scores appeared to represent coordinated downregulation of cellular degradative and antigen presentation pathways in response to infection with different pathogens. Programs with high generality scores were used by the full spectrum of experiments involving exposure of different host cell types to a wide variety of agents. For instance, one program with a high generality score was used by essentially all the experiments in the data set and appeared to represent a "common infection response." This program contained 203 genes, 15% of which code for cytokines or cytokine receptors and was enriched for a number of pathways and biological processes, including the important Toll-like receptor signaling pathway. These results improved on previous meta-analyses of the infection time-series data compendium that relied on extensive prior biological knowledge and manual inspection of data [107]. In contrast to previous analyses, our method was automatic, discovering expression programs, associating them with consistent temporal patterns, and finding significantly overlapping biological pathways and NF-κB binding from ChIP-chip data.

- **Discovered expression programs revealed differences in temporal phasing.** Several expression programs revealed differences in temporal phasing of the response of host cells exposed to different classes of pathogenic organisms. For example, one expression program was enriched for genes involved in ribosomal structure or function. Induction of ribosomal genes may be a prelude to production of critical signaling and defensive proteins, such as chemokines and cytokines. The expression program was induced early in macrophages or dendritic cells exposed to several varieties of Gram-negative bacteria, but induced in the middle of the time-series in host cells exposed to Gram-positive bacteria. As another example, a second expression program was enriched for genes involved in mRNA splicing and oxidative phosphorylation. This program was induced early in time-courses in dendritic cells exposed to bacteria or viruses, but not induced until the middle phase in experiments involving dendritic cells exposed to live fungi or fungal cell components. We hypothesize that the phasing differences observed in induction of the above-mentioned programs may be due to the lesser ability of Gram-positive or fungal organisms to induce critical signaling pathways in innate immune cells.

- **Implication of surprising signaling pathways or host-cell receptor types in the human response to infection.** Some of the gene sets discovered by GeneProgram++ implicated surprising signaling pathways or host-cell receptor types in the response to infection. In particular, two programs were significantly enriched for genes involved in the Wnt signaling pathway, and a third program was significantly enriched for genes coding for neurotransmitter or hormonal receptors. Wnt signaling pathways [32, 129, 168] and neurotransmitter receptors [69, 156, 208, 213] have only recently been implicated in the response to infection. To our knowledge, our work is the first to uncover the activity of these pathways in the data sets analyzed, and to characterize the different temporal behaviors of these pathways in response to a variety of infectious agents and immunomodulatory molecules. We believe that the genes in the above-mentioned expression programs constitute particularly attractive candidates for further biological characterization.

5.4 Directions for future work

5.4.1 The GRAM algorithm

Extensions to handle multi-condition and temporal transcription factor binding data

Although the GRAM algorithm was applied to data from two conditions—rich media and rapamycin exposure—the conditions were considered separately in our analysis. In future work, the algorithm could be extended to model binding data across time or different conditions explicitly. One method for doing this would be to introduce multi-condition binding patterns associated with each transcription factor and each gene. Because the GRAM algorithm considers transcription factor binding as a discrete event, there would be a limited number of possible binding patterns across the different conditions (or time-points). Instead of searching over combinations of transcription factors directly, the extended algorithm would search over combinations of multi-condition binding patterns associated with transcription factors. Assuming that transcription factors bind to a limited number of genes, the extended algorithm could exploit the spareness of the data in the same manner as the original algorithm.

The utility of such an extension is dependent on large compendia of multi-condition (or temporal) binding data. Harbison *et al.* [82] measured binding of yeast transcription factors across several conditions. However, only a small number of factors were profiled in most conditions, and very few factors were profiled in more than two conditions. To our knowledge, there is currently no ChIP-chip data measuring the temporal binding of transcription factors. However, more comprehensive and varied genome-wide transcription factor binding data is likely to become available soon.

Application to nucleosome state and other ChIP-chip data sources

The GRAM algorithm can be readily applied to new types of ChIP-chip data measuring genome-wide protein-DNA interactions. In this thesis, we applied the algorithm to genome-wide transcription factor binding data. Recently, the ChIP-chip technique has also been used to measure genome-wide nucleosome occupancy and histone modification state [28, 153, 179, 144, 113, 27, 223]. Such data, particularly if it measures changes in nucleosome state across multiple conditions, can provide important new insights into cellular processes such as development. The current version of the GRAM algorithm could be applied to single condition nucleosome state data; the extended version described above would be appropriate for multi-condition or temporal data.

Models for larger data sets and coherence across subsets of expression data

As described in Section 2.6.2, the GRAM algorithm is inefficient on extremely large data sets—particularly those in which genes are bound by large numbers of transcription factors—due to the algorithm's exhaustive search step. An additional limitation of the algorithm is that it requires coherence across all expression experiments. Both these limitations could be overcome by resorting to probabilistic search techniques (i.e., choosing random subsets of transcription factors and expression experiments). Although various heuristic scoring schemes could be developed for such probabilistic search methods, an attractive alternative would be to construct a model based on Dirichlet Processes. We note that such a model would differ from that used for GeneProgram, because a GRAM gene module consists of genes that are all co-expressed and bound by the same set of transcription factors. A model capturing GRAM-like modules could be constructed using a Dirichlet Process mixture model, in which each mixture component consists of random variables for each gene in each experiment, indicating whether the gene is bound (or differentially expressed) and the degree of binding (or expression level). Thus, each mixture component would map to a module, specifying a subset of transcription factor and expression experiments and relevant levels.

5.4.2 GeneProgram and GeneProgram++

Improved run-time for model inference

Although GeneProgram is able to analyze large data sets in a reasonable amount of time (on the order of days), faster run-times will be necessary for feasible analyses of truly huge data compendia.

As described in Section 3.3.3, GeneProgram uses Markov Chain Monte Carlo (MCMC) methods for approximate model inference. MCMC is a relatively efficient inference method for GeneProgram, because many of the distributions used by the model are conjugate to one another. The general efficiency of GeneProgram's inference method is attested to by the fact that it ran faster than an implementation of a competing biclustering algorithm (non-negative matrix factorization), which uses a more "traditional" objective maximization algorithm to search for the appropriate number of biclusters deterministically (see Section 3.5.4). However, such relatively good performance does not mean that GeneProgram's run-time cannot be improved.

A "split-merge" MCMC method, recently introduced by Jain and Neal [105, 106], has been shown to improve mixing and convergence times for some types of Dirichlet Process mixture models. The basic idea behind the improved sampling scheme is to probabilistically split or merge mixture component, thereby potentially reassigning multiple data points to components simultaneously. This method can in principle avoid the problem of the MCMC sampler becoming trapped in local modes that cannot be readily escaped through single reassignments of data points. However, even with these improvements to the MCMC method, a general problem remains: there are no clear theoretical guidelines indicating when to stop sampling, and in practice we must collect a very large number of samples in an attempt to find a good approximation to the model posterior. This difficulty encourages the development of alternative inference methods for GeneProgram.

Mean-field variational methods are another alternative for approximate inference for Dirichlet Process mixture models [29]. Variational methods involve deriving a lower bound on the log likelihood for the model using a tractable distribution over the model's hidden variables (i.e., an exponential family distribution). The lower bound is then deterministically optimized. In recent work, variational methods have been shown to have faster run-times while producing results similar to those obtained using MCMC methods for some types of Dirichlet Process mixture models [29].

Expectation propagation methods are yet another alternative for approximate inference for Dirichlet Process mixture models [141]. This inference technique iteratively attempts to find an optimal approximation of a complex distribution using products of simpler functions. Minka worked out the update equations for a simple Dirichlet Process mixture model [141]. However, even for this case, the algorithm apparently produces worse results than does MCMC sampling.

None of the above alternative approximate inference methods (including split-merge MCMC sampling) have been applied to Hierarchical Dirichlet Processes (HDPs) so far. Such applications are not straightforward, because of the dependencies introduced among mixture components in HDP models. GeneProgram and GeneProgram++ further extend the HDP model, making inference even more difficult. Additionally, none of the above-mentioned alternative inference methods provide direct means for updating of concentration parameters or other model hyperparameters. However, despite these complications, it is likely that one or more of the alternative approximate inference methods discussed above can be tailored to work with GeneProgram and GeneProgram++ to improve our algorithms' run-times.

Applications to novel data types

As discussed in Section 3.6, GeneProgram is a general method, making it suitable for analyzing any large expression data compendium, including those relating to developmental or disease processes. Our framework is also flexible, and could accommodate other genome-wide sources of biological data in future work, such as DNA-protein binding or DNA sequence motif information. As a specific example, compendia of ChIP-chip data could readily be analyzed by discretizing binding data (either continuous ratios or binding p-values as the GRAM algorithm does). In such an application, mixture components in the GeneProgram model would correspond to sets of genes that are generally bound by the same factors. Groups would correspond to transcription factors that tend to act together (e.g., complexes or co-factors). Overall, GeneProgram's ability to discover tissue groups and expression programs *de novo* using a principled probabilistic method, as well as its use of data in a semi-quantitative manner, makes it especially valuable for novel "meta-analysis" applications involving large data sets of unknown complexity in which direct fully quantitative comparisons are difficult.

Dependencies among gene expression events

As discussed in Section 3.6, an unrealistic assumption of the GeneProgram model is that gene expression "events" are independently generated by expression programs. There are several approaches by which this assumption could be relaxed.

The n-gram methods developed for text analysis applications of topic models could be applied. The assumption of independent expression events in GeneProgram is equivalent to the "bag-of-words" assumption in topic models applied to text analysis. Despite the success of the "bag-of-words" assumption [64, 79, 31], word order in documents can clearly provide important information. In n-gram models, words are no longer generated independently, but are assumed to depend on n other words. Bigram models ($n = 2$) are relatively efficient, because they assume that the probability for generating a word in a document is dependent only on the previous word generated [216]. Thus, instead of parameterizing the word probabilities for each topic with a multinomial distribution, the topic uses a conditional probability table that specifies the probability of generating each word conditioned on any other word. If gene expression data for each experiment were converted to a ranking of genes, an n-gram model would be meaningful for expression programs. That is, the model would capture the information that "gene X's expression is likely to be greater than gene Y's in expression program Z." The use of rankings rather than quantitative expression values could be advantageous, due to the reasons discussed in Section 3.6.

Another approach would be to use two variables to model observed expression data for a gene: a binary variable indicating whether a gene is "used" by the program and a continuous variable modeling the level of expression. An expression program would then consist of a binomial distribution and a continuous level distribution (e.g., an exponential distribution) for each gene. A potential difficulty with this approach is that it is not obvious how discrete indicator variables and continuous expression levels should be weighted in the model likelihood function. Additionally, as discussed in Section 3.6, it is unclear whether observed continuous expression data has a meaningful biological interpretation.

Yet another alternative would be to deal with continuous expression data directly and not use topic models. For instance, expression data for a tissue could be modeled as a linear combination over expression programs. Battle *et al.* developed such a model, but assumed the number of programs was fixed and that experiments were independent [23]. Such a model could in principle be extended to use a Hierarchical Dirichlet Process framework, although efficient inference methods would need to be developed. Additionally, as mentioned previously, it is unclear the extent to which observed continuous expression data has a meaningful biological interpretation.

Explicit dependencies among expression programs

In the GeneProgram model, the fact that tissues in the same group are more likely to share expression programs induces implicit dependencies among programs.

An alternate approach would be to model hierarchical dependencies among programs explicitly. For example, with such a model, we might discover a hierarchical structure among programs involved in metabolism, with general programs near the root and very specific programs at the leaves. Li and McCallum introduced a model in which topic dependencies are specified using a directed acyclic graph structure [120]. Although their method is based on a fixed number of topics, the extension to an infinite mixture model using Dirichlet Processes seems quite feasible.

Another interesting explicit dependency to model would be expression program relationships over time. Blei and Lafferty introduced the *dynamic topic model*, in which both the content of topics (probabilities over words) and the prior distribution for choosing topics (mixture weights) evolve over time [30]. They applied their model to a corpus of over 100 years of articles from the journal *Science*, and demonstrated how topics relating to scientific disciplines such as atomic physics were automatically discovered in late nineteenth century articles, and evolved to be used by modern articles on atomic physics. One could imagine applying a version of such a model to an extensive compendium of developmental time-series or cross-species gene expression data, and discovering how programs change during growth of an organism or throughout evolution. At present, data is still too limited to be useful in such a model, but prospects for future applications of this sort are very exciting.

Bibliography

[1] J. Aach and G. M. Church. Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17:495–508, 2001.

[2] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, 1972.

[3] Affymetrix, Inc., corporate website. *www.affymetrix.com.*

[4] Affymetrix, Inc. *GeneChip Expression Analysis Technical Manual*, 2007.

[5] Agilent Technologies, Inc., corporate website. *www.chem.agilent.com.*

[6] A. Alibes, P. Yankilevich, A. Canada, and R. Diaz-Uriarte. IDconverter and IDClight: conversion and annotation of gene and protein IDs. *BMC Bioinformatics*, 8:9, 2007.

[7] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, and L. M. Staudt. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Science*, 403:503–510, 2000.

[8] O. Alter, P. O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proc Natl Acad Sci U S A*, 97(18):10101–6, Aug 29 2000.

[9] O. Alter, P. O. Brown, and D. Botstein. Generalized singular value decomposition for comparative analysis of genome-scale expression data sets of two different organisms. *Proc Natl Acad Sci U S A*, 100(6):3351–6, Mar 18 2003.

[10] R. Anderson. Myth-Quoted: Words of Chief Seattle Were Eloquent - But Not His. *The Seattle Times*, page A1, July 1991.

[11] Anonymous. The Gene Ontology (GO) project in 2006. *Nucleic Acids Res*, 34(Database issue):D322–6, Jan 1 2006.

[12] C. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Statistics*, 2(6):1152–1174, 1974.

[13] A. Arkin, J. Ross, and H. H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected *Escherichia coli* cells. *Genetics*, 149(4):1633–48, 1998.

[14] M. Badoiu and K. Clarkson. Smaller core-sets for balls. In *14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2003.

[15] H. V. Baker. Glycolytic gene expression in *Saccharomyces cerevisiae*: nucleotide sequence of GCR1, null mutants, and evidence for expression. *Mol. Cell Biol*, 6:3774–3784, 1986.

[16] Z. Bar-Joseph. *Inferring interactions, expression programs and regulatory networks from high throughput biological data*. PhD thesis, MIT EECS, 2003.

[17] Z. Bar-Joseph. Analyzing time series gene expression data. *Bioinformatics*, 20:2493–503, 2004.

[18] Z. Bar-Joseph, G. Gerber, T. S. Jaakkola, D. K. Gifford, and I. Simon. A new approach to analyzing gene expression time series data. In *Proceedings of The Sixth Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 39–48, 2002.

[19] Z. Bar-Joseph, G. Gerber, T. S. Jaakkola, D. K. Gifford, and I. Simon. Continuous representations of time series gene expression data. *Journal of Computational Biology*, 3-4:341–356, 2003.

[20] Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford. *Supplemental website*. www.psrg.lcs.mit.edu/ zivbj/modules/modules.html.

[21] Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford. Computational discovery of gene modules and regulatory networks. *Nature Biotechnology*, 21:1337–1342, 2003.

[22] A. Basu, I. R. Harris, and S. Basu. Minimum distance estimation: The approach using density-based distances. In G. S. Maddala and C. R. Rao, editors, *Handbook of Statistics*, volume 15, pages 21–48. North-Holland, 1997.

[23] A. Battle, E. Segal, and D. Koller. Probabilistic discovery of overlapping cellular processes and their regulation. *J Comput Biol*, 12(7):909–27, Sep 2005.

[24] Y. Benjamini and Y. Hochberg. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society, Series B (Methodological)*, 57(1):289–300, 1995.

[25] S. Bergmann, J. Ihmels, and N. Barkai. Similarities and differences in genome-wide expression data of six organisms. *PLoS Biol*, 2(1):E9, Jan 2004.

[26] B. P. Berman, Y. Nibu, B. D. Pfeiffer, P. Tomancak, S. E. Celniker, M. Levine, G. M. Rubin, and M. B. Eisen. Exploiting transcription factor binding site clustering to identify *cis*-regulatory modules involved in pattern formation in the *Drosophila* genome. *Proc Natl Acad Sci U S A*, 99(2):757–62, Jan 22 2002.

[27] B. E. Bernstein, M. Kamal, K. Lindblad-Toh, S. Bekiranov, D. K. Bailey, D. J. Huebert, S. McMahon, E. K. Karlsson, E. J. Kulbokas, T. R. Gingeras, S. L. Schreiber, and E. S. Lander. Genomic maps and comparative analysis of histone modifications in human and mouse. *Cell*, 128(2):169–181, January 2005.

[28] E. L. Bernstein, R. L. Humphrey, R. Erlich, R. Schneider, P. Bouman, J. S. Liu, T. Kouzarides, and S. L. Schreiber. Methylation of histone h3 lys 4 in coding regions of active genes. *PNAS*, 99:8695–8700, 2002.

[29] D. M. Blei and M. I. Jordan. Variational inference for Dirichlet process mixtures. *Journal of Bayesian Analysis*, 1:121–144, 2005.

[30] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 113–120, New York, NY, USA, 2006. ACM Press.

[31] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[32] A. Blumenthal, S. Ehlers, J. Lauber, J. Buer, C. Lange, T. Goldmann, H. Heine, E. Brandt, and N. Reiling. The Wingless homolog WNT5A and its receptor Frizzled-5 regulate inflammatory responses of human mononuclear cells induced by microbial stimulation. *Blood*, 108:965–973, 2006.

[33] 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. *Proc Natl Acad Sci U S A*, 99(2):972–7, 2002.

[34] M. I. Borelli, F. E. Estivariz, and J. J. Gagliardino. Evidence for the paracrine action of islet-derived corticotropin-like peptides on the regulation of insulin release. *Metabolism*, 45(5):565–70, May 1996.

[35] E. Boy-Marcotte, M. Perrot, F. Bussereau, H. Boucherie, and M. Jacquet. Msn2p and Msn4p control a large number of genes induced at the diauxic transition which are repressed by cyclic AMP in *Saccharomyces cerevisiae*. *J. Bacteriol.*, 180:1044–1052, 1998.

[36] J. P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proc Natl Acad Sci U S A*, 101(12):4164–9, Mar 23 2004.

[37] W. Callebaut and D. Rasskin-Gutman, editors. *Modularity.* The MIT Press, 2005.

[38] M. E. Cardenas, N. S. Cutler, M. C. Lorenz, C. J. Di Como, and J. Heitman. The TOR signaling cascade regulates gene expression in response to nutrients. *Genes Dev.*, 13:3271–3279, 1999.

[39] P. Carmona-Saez, R. D. Pascual-Marqui, F. Tirado, J. M. Carazo, and A. Pascual-Montano. Biclustering of gene expression data by non-smooth non-negative matrix factorization. *BMC Bioinformatics*, 7:78, 2006.

[40] S. Cawley, S. Bekiranov, H. H. Ng, P. Kapranov, E. A. Sekinger, D. Kampa, A. Piccolboni, V. Sementchenko, J. Cheng, A. J. Williams, R. Wheeler, B. Wong, J. Drenkow, M. Yamanaka, S. Patel, S. Brubaker, H. Tammana, G. Helt, K. Struhl, and T. R. Gingeras. Unbiased Mapping of Transcription Factor Binding Sites along Human Chromosomes 21 and 22 Points to Widespread Regulation of Noncoding RNAs. *Cell*, 116:499–509, February 2004.

[41] L. F. Chen and W. C. Greene. Shaping the nuclear action of NF-kappaB. *Nat Rev Mol Cell Biol*, 5(5):392–401, 2003.

[42] Y. Cheng and G. M. Church. Biclustering of expression data. In *Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 93–103. AAAI Press, 2000.

[43] V. G. Cheung, M. Morley, F. Aguilar, A. Massimi, R. Kucherlapati, and G. Childs. Making and reading microarrays. *Nature Genetics Supplement*, 21:15–19, 1999.

[44] J. A. Coffman, R. Rai, T. Cunningham, V. Svetlov, and T. G. Cooper. Gat1p, a GATA family protein whose production is sensitive to nitrogen catabolite repression, participates in transcriptional activation of nitrogen-catabolic genes in *Saccharomyces cerevisiae*. *Mol. Cell Biol.*, 16:847–858, 1996.

[45] N. E. Cooke, D. Coit, J. Shine, J. D. Baxter, and J. A. Martial. Human prolactin: cDNA structural analysis and evolutionary comparisons. *J Biol Chem*, 256(8):4007–16, Apr 25 1981.

[46] T. G. Cooper. Transmitting the signal of excess nitrogen in *Saccharomyces cerevisiae* from the Tor proteins to the GATA factors: connecting the dots. *FEMS Microbiol. Rev.*, 26:223–238, 2002.

[47] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley-Interscience, 1991.

[48] J. L. Crespo and M. N. Hall. Elucidating TOR signaling and rapamycin action: lessons from *Saccharomyces cerevisiae*. *Microbiol. Mol. Biol. Rev*, 66:579–591, 2002.

[49] J. L. Crespo, T. Powers, B. Fowler, and M. N. Hall. The TOR-controlled transcription activators GLN3, RTG1, and RTG3 are regulated in response to intracellular levels of glutamine. *Proc. Natl. Acad. Sci. U S A*, 99:6784–6789, 2002.

[50] V. D. Dang, C. Bohn, M. Bolotin-Fukuhara, and B. Daignan-Fornier. The CCAAT box-binding factor stimulates ammonium assimilation in *Saccharomyces cerevisiae*, defining a new cross-pathway regulation between nitrogen and carbon metabolisms. *J. Bacteriol.*, 178:1842–1849, 1996.

[51] V. D. Dang, M. Valens, M. Bolotin-Fukuhara, and B. Daignan-Fornier. Cloning of the ASN1 and ASN2 genes encoding asparagine synthetases in *Saccharomyces cerevisiae*: differential regulation by the CCAAT-box-binding factor. *Mol. Microbiol.*, 22:681–692, 1996.

[52] A. J. de Bold. Atrial natriuretic factor: a hormone produced by the heart. *Science*, 230(4727):767–70, Nov 15 1985.

[53] R. O. Dror, J. G. Murnick, N. J. Rinaldi, V. D. Marinescu, R. M. Rifkin, and R. A. Young. A Bayesian approach to transcript estimation from gene array data: the Beam technique. In *Proceedings of the Sixth Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, 2002.

[54] A. M. Dudley, J. Aach, M. A. Steffen, and G. M. Church. Measuring absolute expression with microarrays with calibrated reference sample and an extended signal intensity range. *PNAS*, 99:7554–7559, 2002.

[55] D. Dueck, Q. D. Morris, and B. J. Frey. Multi-way clustering of microarray data using probabilistic sparse matrix factorization. *Bioinformatics*, 21 Suppl 1:i144–51, Jun 2005.

[56] D. J. Duggan, M. Bittner, Y. Chen, P. Meltzer, and J. M. Trent. Expression profiling using cDNA microarrays. *Nature Genetics Supplement*, 21:10–14, 1999.

[57] Y. H. Edwards, S. Sakoda, E. Schon, and S. Povey. The gene for human muscle-specific phosphoglycerate mutase, PGAM2, mapped to chromosome 7 by polymerase chain reaction. *Genomics*, 5(4):948–51, Nov 1989.

[58] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A*, 95(25):14863–8, Dec 8 1998.

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

[60] M. D. Escobar and M. West. Bayesian Density Estimation and Inference Using Mixtures. *Journal of the American Statistical Association*, 90(430):577–588, 1995.

[61] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230, 1973.

[62] T. S. Ferguson. Prior distributions on spaces of probability measures. *Annals of Statistics*, 2:615–629, 1974.

[63] M. A. Fisher, B. B. Plikaytis, and T. M. Shinnick. Microarray analysis of the *Mycobacterium tuberculosis* transcriptional response to the acidic conditions found in phagosomes. *J. Bacteriology*, 184:4025–32, 2002.

[64] P. Flaherty, G. Giaever, J. Kumm, M. I. Jordan, and A. P. Arkin. A latent variable model for chemogenomic profiling. *Bioinformatics*, 21(15):3286–93, Aug 1 2005.

[65] M. C. Fonseca. The contribution of nuclear compartmentalization to gene regulation. *Cell*, 108:513–521, 2002.

[66] S. L. Forsburg and L. Guarente. Identification and characterization of HAP4: a third component of the CCAAT-bound HAP2/HAP3 heteromer. *Genes Dev.*, 3:1166–1178, 1989.

[67] D. A. Freedman. On the asymptotic behavior of Bayes estimates in the discrete case. *Annals of Mathematical Statistics*, 34:1386–1403, 1963.

[68] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian Networks to Analyze Expression Data. *Journal of Computational Biology*, 7:601–620, 2000.

[69] G. Galvis, K. S. Lips, and W. Kummer. Expression of nicotinic acetylcholine receptors on murine alveolar macrophages. *J Mol Neurosci*, 30:107–108, 2006.

[70] D. J. Garry, G. A. Ordway, J. N. Lorenz, N. B. Radford, E. R. Chin, R. W. Grange, R. Bassel-Duby, and R. S. Williams. Mice without myoglobin. *Nature*, 395(6705):905–8, Oct 29 1998.

[71] A. P. Gasch, P. T. Spellman, C. M. Kao, O. Carmel-Harel, M. B. Eisen, G. Storz, D. Botstein, and P. O. Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Mol Biol Cell*, 11:4241–4257, 2004.

[72] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman & Hall, 2004.

[73] G. K. Gerber. Do the time wrap: Continuous alignment of gene expression time series data. Master's thesis, MIT EECS, 2002.

[74] G. K. Gerber, R. D. Dowell, T. S. Jaakkola, and D. K. Gifford. *Table 1 (Supplemental): Summary of expression programs discovered by GeneProgram from Novartis Tissue Atlas v2 data, CSAIL Digial Work Product Archive*. http://hdl.handle.net/1721.1/37602.

[75] G. K. Gerber, R. D. Dowell, T. S. Jaakkola, and D. K. Gifford. *Table 2 (Supplemental): Complete data for all 100 expression programs discovered by GeneProgram from the Novartis Gene Atlas v2, CSAIL Digial Work Product Archive*. http://hdl.handle.net/1721.1/37603.

[76] G. K. Gerber, R. D. Dowell, T. S. Jaakkola, and D. K. Gifford. Automated discovery of functional generality of human gene expression programs. *PLoS Computational Biology*, 3(8):e148, 2007.

[77] J. L. Gerton, J. DeRisi, R. Shroff, M. Lichten, P. O. Brown, and T. D. Petes. Inaugural article: global mapping of meiotic recombination hotspots and coldspots in the yeast *Saccharomyces cerevisiae*. *PNAS*, 97:1138311390, 2000.

[78] A. Gotthelf, editor. *Aristotle:* Historia Animalium*, Volume I, Books I-X*. Cambridge University Press, 2002.

[79] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc Natl Acad Sci U S A*, 101 Suppl 1:5228–35, Apr 6 2004.

[80] K. Guillemin, N. R. Salama, L. S. Tompkins, and S. Falkow. Cag pathogenicity island-specific responses of gastric epithelial cells to *Helicobacter pylori* infection. *Proc Natl Acad Sci U S A*, 99(23):15136–41, 2002.

[81] P. Gunning, P. Ponte, L. Kedes, R. Eddy, and T. Shows. Chromosomal location of the co-expressed human skeletal and cardiac actin genes. *Proc Natl Acad Sci U S A*, 81(6):1813–7, Mar 1984.

[82] C. T. Harbison, D. B. Gordon, T. I. Lee, N. J. Rinaldi, K. D. MacIsaac, T. W. Danford, N. M. Hannett, J-B Tagne, D. B. Reynolds, J. Yoo, E. G. Jennings, J. Zeitlinger, D. K. Pokholok, M. Kellis, P. A. Rolfe, K. T. Takusagawa, E. S. Lander, D. K. Gifford, E. Fraenkel, and R. A. Young. Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431:99–104, September 2004.

[83] J. S. Hardwick, F. G. Kuruvilla, J. K. Tong, A. F. Shamji, and S. L. Schreiber. Rapamycin-modulated transcription defines the subset of nutrient-sensitive signaling pathways directly controlled by the Tor proteins. *Proc. Natl. Acad. Sci. U S A*, 96:14866–14870, 1999.

[84] A. J. Hartemink, D. K. Gifford, T. S. Jaakkola, and R. A. Young. Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In *Pac. Symp. on Biocomputing (PSB)*, pages 422–433, 2001.

[85] A. J. Hartemink, D. K. Gifford, T. S. Jaakkola, and R. A. Young. Combining location and expression data for principled discovery of genetic regulatory network models. In *Pac. Symp. on Biocomputing (PSB)*, pages 437–449, 2002.

[86] R. Hasan, C. Leroy, A. D. Isnard, J. Labarre, E. Boy-Marcotte, and M. B. Toledano. The control of the yeast H_2O_2 response by the Msn2/4 transcription factors. *Mol. Microbiol.*, 45:233–241, 2002.

[87] S. Hermann-Le Denmat, M. Werner, and A. Sentenac. Suppression of yeast RNA polymerase III mutations by FHL1, a gene coding for a fork head protein involved in rRNA processing. *Mol Cell Biol.*, 14:2905–2913, 1994.

[88] P. Hieter and M. Boguski. Functional genomics: It's all how you read it. *Science*, 278:601–602, 1997.

[89] M. J. Holland, T. Yokoi, J.P. Holland, K. Myambo, and M. A. Innis. The GCR1 gene encodes a positive transcriptional regulator of the enolase and glyceraldehyde-3-phosphate dehydrogenase gene families in *Saccharomyces cerevisiae*. *Mol. Cell Biol*, 7:813–820, 1989.

[90] C. E. Horak, M. C. Mahajan, N. M. Luscombe, M. Gerstein, S. M. Weissman, and M. Snyder. GATA-1 binding sites mapped in the betaglobin locus by using mammalian ChIP-chip analysis. *PNAS*, 99:29242929, 2002.

[91] 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):870–5, 2001.

[92] E. Hubbell, W. M. Liu, and R. Mei. Robust estimators for expression analysis. *Bioinformatics*, 18(12):1585–92, Dec 2002.

[93] J. Huelsken and W. Birchmeier. New aspects of Wnt signaling pathways in higher vertebrates. *Curr Opin Genet Dev*, 11:547–553, 2001.

[94] D. A. Hume. Probability in transcriptional regulation and its implications for leukocyte differentiation and inducible gene expression. *Blood*, 96:2323–2328, 2000.

[95] A. Hummel, U. Lendeckel, H. Hahn von Dorsche, and H. Zuhlke. Presence and regulation of a truncated proopiomelanocortin gene transcript in rat pancreatic islets. *Biol Chem Hoppe Seyler*, 373(10):1039–44, Oct 1992.

[96] T. Ideker, O. Ozier, B. Schwikowski, and A. F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, Suppl 1:S233–40, 2002.

[97] T. Ideker, V. Thorsson, S. F. Siegel, and L. E. Hood. Testing for differentially-expressed genes by maximum-likelihood analysis of microarray data. *Journal of Computational Biology*, 7:805–817, 2000.

[98] A. M. Idicula, G. L. Blatch, T. G. Cooper, and R. A. Dorrington. Binding and activation of the zinc cluster transcription factors of *Saccharomyces cerevisiae*. *J. Biol. Chem.*, 277:45977–45983, 2002.

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

[100] S. Imbeaud and C. Auffray. 'The 39 steps' in gene expression profiling: critical issues and proposed best practices for microarray experiments. *Drug Discovery Today*, 10:1175–1182, 2005.

[101] J. P. A. Ioannidis. Microarrays and molecular research: noise discovery? *The Lancet*, 365:454–455, 2005.

[102] H. Ishwaran and L. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, 2001.

[103] V. R. Iyer, C. E. Horak, C. S. Scafe, D. Botstein, M. Snyder, and P. O. Brown. Genomic binding sites of the yeast cell-cycle transcription factors SBF and MBF. *Nature*, 409:533–538, 2001.

[104] E. Jacinto and M. N. Hall. Tor signalling in bugs, brain and brawn. *Nat. Rev. Mol. Cell Biol.*, 4:117–126, 2003.

[105] S. Jain and R. M. Neal. A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13:158–182, 2004.

[106] S. Jain and R. M. Neal. Splitting and merging components of a nonconjugate Dirichlet process mixture model. *Bayesian Analysis*, 1(5):1–38, 2007.

[107] R. G. Jenner and R. A. Young. Insights into host responses against pathogens from transcriptional profiling. *Nat Rev Microbiol*, 3(4):281–94, Apr 2005.

[108] M. D. Kane, T. A. Jatkoe, C. R. Stumpf, J. Lu, J. D. Thomas, and S. J. Madore. Assessment of the sensitivity and specificity of oligonucleotide (50mer) microarrays. *Nucleic Acids Research*, 28:4552–4557, 2000.

[109] M. Kanehisa, S. Goto, M. Hattori, K. F. Aoki-Kinoshita, M. Itoh, S. Kawashima, T. Katayama, M. Araki, and M. Hirakawa. From genomics to chemical genomics: new developments in KEGG. *Nucleic Acids Res.*, 34:D354–357, 2006.

[110] R. E. Kass and A. E. Raftery. Bayes Factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

[111] P. M. Kim and B. Tidor. Subsystem identification through dimensionality reduction of large-scale gene expression data. *Genome Res*, 13(7):1706–18, Jul 2003.

[112] A. Komeili, K. P. Wedaman, E. K. O'Shea, and T. Powers. Mechanism of metabolic control: Target of rapamycin signaling links nitrogen quality to the activity of the Rtg1 and Rtg3 transcription factors. *J. Cell Biol.*, 151:863–878, 2000.

[113] S. K. Kurdistani, S. Tavazoie, and M. Grunstein. Mapping global histone acetylation patterns to gene expression. *Cell*, 117:721–733, June 2004.

[114] K. Kurihara, M. Welling, and N. Vlassis. Accelerated variational Dirichlet process mixtures. In *Neural Information Processing Systems (NIPS)*, Vancouver, B.C., 2006.

[115] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–91, Oct 21 1999.

[116] J. Lee, C. Godon, G. Lagniel, D. Spector, J. Garin, J. Labarre, and M. B. Toledano. Yap1 and Skn7 control two specialized oxidative stress response regulons in yeast. *J Biol Chem*, 274(23):16040–16046, 1999.

[117] T. I. Lee, R. G. Jenner, L. A. Boyer, M. G. Guenther, S. S. Levine, R. M. Kumar, B. Chevalier, S. E. Johnstone, M. F. Cole, K. Isono, H. Koseki, T. Fuchikami, K. Abe, H. L. Murray, J. P. Zucker, B. Yuan, G. W. Bell, E. Herbolsheimer, N. M. Hannett, K. Sun, D. T. Odom, A. P. Otte, T. L. Volkert, D. P. Bartel, D. A. Melton, D. K. Gifford, R. Jaenisch, and R. A. Young. Control of developmental regulators by Polycomb in human embryonic stem cells. *Cell*, 125:301–313, 2006.

[118] T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, N. M. Hannett, C. R. Harbison, C. M. Thompson, I. Simon, J. Zeitlinger, E. G. Jennings, H. L. Murray, D. B. Gordon, B. Ren, J. J. Wyrick, J-B Tagne, T. L. Volkert, E. Fraenkel, D. K. Gifford, and R. A. Young. Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, 798:799–804, 2002.

[119] T. Lehmeier, V. Raker, H. Hermann, and R. Luhrmann. cDNA cloning of the Sm proteins D2 and D3 from human small nuclear ribonucleoproteins: evidence for a direct D1-D2 interaction. *Proc Natl Acad Sci U S A*, 91(25):12317–21, Dec 6 1994.

[120] W. Li and A. McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. In *ICML '06: Proceedings of the 23rd International Conference on Machine Learning*, pages 577–584, New York, NY, USA, 2006. ACM Press.

[121] X. Li and W. H. Wong. Sampling motifs on phylogenetic trees. *PNAS*, 102(27):9481–6, 2005.

[122] Z. Li, S. Van Calcar, C. Qu, W. K. Cavenee, M. Q. Zhang, and B. Ren. A global transcriptional regulatory role for c-Myc in Burkitts lymphoma cells. *PNAS*, 100:8164–8169, 2003.

[123] B. Y. Liao and J. Zhang. Evolutionary conservation of expression profiles between human and mouse orthologous genes. *Mol Biol Evol*, 23(3):530–40, Mar 2006.

[124] X. Liao and R. A. Butow. RTG1 and RTG2: two yeast genes required for a novel path of communication from mitochondria to the nucleus. *Cell*, 72:61–71, 1993.

[125] J. D. Lieb, X. Liu, D. Botstein, and P. O. Brown. Promoter-specific binding of Rap1 revealed by genome-wide maps of protein-DNA association. *Nature Genetics*, 28:327–334, 2001.

[126] R. J. Lipshutz, S. P. A. Fodor, T. R. Gingeras, and D. J. Lockhart. High density synthetic oligonucleotide arrays. *Nature Genetics Supplement*, 21:20–24, 1999.

[127] J. W. Little. Threshold effects in gene regulation: when some is not enough. *Proc Natl Acad Sci U S A*, 102(15):5310–1, 2005.

[128] M. Liu, S. J. Popper, K. H. Rubins, and D. A. Relman. Early days: genomics and human responses to infection. *Curr Opin Microbiol*, 9(3):312–9, Jun 2006.

[129] I. B. Lobov, S. Rao, T. J. Carroll, J. E. Vallance, M. Ito, J. K. Ondr, S. Kurup, D. A. Glass, M. S. Patel, W. Shu, E. E. Morrisey, A. P. McMahon, G. Karsenty, and R. A. Lang. WNT7b mediates macrophage-induced programmed cell death in patterning of the vasculature. *Nature*, 437:417–21, 2005.

[130] H. Lodish, A. Berk, P. Matsudaira, C. A. Kaiser, M. Krieger, M. P. Scott, L. Zipursky, and J. Darnell. *Molecular Cell Biology*. W.H. Freeman, 2007.

[131] N. M. Luscombe, M. M. Babu, H. Yu, M. Snyder, S. A. Teichmann, and M. Gerstein. Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, 431(7006):308–12, 2004.

[132] R. E. MacLaury. Prototypes revisited. *Annual Review of Anthropology*, 20:55–74, 1991.

[133] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[134] R. M. Marion, A. Regev, E. Segal, Y. Barash, D. Koller, N. Friedman, and E. K. O'Shea. Sfp1 is a stress- and nutrient-sensitive regulator of ribosomal protein gene expression. *PNAS*, 101(40):14315–14322, 2004.

[135] The Mathworks, Natick, MA. *Matlab*, r14 edition, 2006.

[136] V. Matys, E. Fricke, R. Geffers, E. Gossling, M. Haubrock, R. Hehl, K. Hornischer, D. Karas, A. E. Kel, O. V. Kel-Margoulis, D. U. Kloos, S. Land, B. Lewicki-Potapov, H. Michael, R. Munch, I. Reuter, S. Rotert, H. Saxel, M. Scheer, S. Thiele, and E. Wingender. TRANSFAC: transcriptional regulation, from patterns to profiles. *Nucleic Acids Res.*, 31:374–378, 2003.

[137] N. J. McKenna and B. W. O'Malley. Combinatorial control of gene expression by nuclear receptors and coregulators. *Cell*, 108:465–474, 2002.

[138] G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley & Sons, 2000.

[139] M. Medvedovic, K. Y. Yeung, and R. E. Bumgarner. Bayesian mixture model based clustering of replicated microarray data. *Bioinformatics*, 20(8):1222–32, May 22 2004.

[140] H. W. 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 Res.*, 30:31–34, 2002.

[141] T. Minka and Z. Ghahramani. Expectation propagation for infinite mixtures. In *Neural Information Processing Systems (NIPS)*, Vancouver, B.C., 2003.

[142] K. L. Moore and A. F. Dalley. *Clinically Oriented Anatomy*. Lippincott Williams & Wilkins, fourth edition, 1999.

[143] M. Mueller, L. Martens, and R. Apweiler. Annotating the human proteome: beyond establishing a parts list. *Biochimica et Biophysica Acta*, 2006.

[144] P. L. Nagy, M. L. Cleary, P. O. Brown, and J. D. Lieb. Genomewide demarcation of RNA polymerase II transcription units revealed by physical fractionation of chromatin. *PNAS*, 100:6364–6369, 2003.

[145] G. J. Narlikar, H. Y. Fan, and R. E. Kingston. Cooperation between complexes that regulate chromatin structure and transcription. *Cell*, 108:475–487, 2002.

[146] K. Natarajan, M. R. Meyer, B. M. Jackson, D. Slade, C. Roberts, A. G. Hinnebusch, and M. J. Marton. Transcriptional profiling shows that Gcn4p is a master regulator of gene expression during amino acid starvation in yeast. *Mol Cell Biol.*, 21:4347–4368, 2001.

[147] National Center for Biotechnology Information, website. *www.ncbi.nlm.nih.gov*.

[148] National Institutes of Health Biomedical Information Science and Technology Initiative, website. *www.bisti.nih.gov*.

[149] G. J. Nau, J. F. Richmond, A. Schlesinger, E. G. Jennings, E. S. Lander, and R. A. Young. Human macrophage activation programs induced by bacterial pathogens. *Proc Natl Acad Sci U S A*, 99(3):1503–8, 2002.

[150] G. J. Nau, A. Schlesinger, J. F. Richmond, and R. A. Young. Cumulative Toll-like receptor activation in human macrophages treated with whole bacteria. *J Immunol*, 170(10):5203–9, 2003.

[151] D. J. Navarro, T. L. Griffiths, M. Steyvers, and M. D. Lee. Modeling individual differences using Dirichlet processes. *Journal of Mathematical Psychology*, 50:101–122, 2006.

[152] R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:249–265, 2000.

[153] H. H. Ng, F. Robert, R. A. Young, and K. Struhl. Genome-wide location and regulated recruitment of the RSC nucleosome-remodeling complex. *Genes Dev.*, 16:806–819, 2002.

[154] NimbleGen Systems, Inc., corporate website. *www.nimblegen.com*.

[155] M. Nykter, T. Aho, M. Ahdesmaki, P. Ruusuvuori, A. Lehmussola, and O. Yli-Harja. Simulation of microarray data with realistic characteristics. *BMC Bioinformatics*, 7:349, 2006.

[156] O. Ofek, M. Karsak, N. Leclerc, M. Fogel, B. Frenkel, K. Wright, J. Tam, M. Attar-Namdar, V. Kram, E. Shohami, R. Mechoulam, A. Zimmer, and I. Bab. Peripheral cannabinoid receptor, CB2, regulates bone mass. *Proc Natl Acad Sci U S A*, 103:696–701, 2006.

[157] K. Ohta, Y. Nobukuni, H. Mitsubuchi, T. Ohta, T. Tohma, Y. Jinno, F. Endo, and I. Matsuda. Characterization of the gene encoding human pituitary-specific transcription factor, Pit-1. *Gene*, 122(2):387–8, Dec 15 1992.

[158] S. H. Orkin and L. I. Zon. Genetics of erythropoiesis: induced mutations in mice and zebrafish. *Annu Rev Genet*, 31:33–60, 1997.

[159] G. Orphanides and D. Reinberg. A unified theory of gene expression. *Cell*, 108:439–451, 2002.

[160] N. Pathan, C. A. Hemingway, A. A. Alizadeh, A. C. Stephens, J. C. Boldrick, E. E. Oragui, C. McCabe, S. B. Welch, A. Whitney, P. O'Gara, S. Nadel, D. A. Relman, S. E. Harding, and M. Levin. Role of interleukin 6 in myocardial dysfunction of meningococcal septic shock. *Lancet*, 363(9404):203–9, 2004.

[161] T. Pieler and Y. Chen. Forgotten and novel aspects in pancreas development. *Biol Cell*, 98(2):79–88, Feb 2006.

[162] Y. Pilpel, P. Sudarsanam, and G. M. Church. Identifying regulatory networks by combinatorial analysis of promoter elements. *Nature Genetics*, 29:153–159, 2001.

[163] J. L. Pinkham and L. Guarente. Cloning and molecular analysis of the HAP2 locus: a global regulator of respiratory genes in *Saccharomyces cerevisiae*. *Mol. Cell Biol.*, 5:3410–3416, 1985.

[164] D. K. Pokholok, C. T. Harbison, S. Levine, M. Cole, N. M. Hannett, T. I. Lee, G. W. Bell, K. Walker, P. A. Rolfe, E. Herbolsheimer, J. Zeitlinger, F. Lewitter, D. K. Gifford, and R. A. Young. Genome-wide map of nucleosome acetylation and methylation in yeast. *Cell*, 122:517–527, 2005.

[165] K. V. Prasanth and D. L. Spector. Eukaryotic regulatory RNAs: an answer to the genome complexity conundrum. *Genes & Dev.*, 21:11–42, 2007.

[166] SGD Project. *Saccharomyces Genome Database*, 2003. www.yeastgenome.org.

[167] M. Ptashne and E. Ganna. *Genes and Signals*. Cold Spring Habor Laboratory Press, 2002.

[168] T. Pukrop, F. Klemm, T. Hagemann, D. Gradl, M. Schulz, S. Siemes, L. Trumper, and C. Binder. Wnt 5a signaling is critical for macrophage-induced invasion of breast cancer cell lines. *Proc Natl Acad Sci U S A*, 103:5454–9, 2006.

[169] Y. Qi and H. Ge. Modularity and Dynamics of Cellular Networks. *PLoS Computational Biology*, 2:1502–1510, 2006.

[170] Y. Qi, A. Rolfe, K. D. MacIsaac, G. K. Gerber, D. Pokholok, J. Zeitlinger, T. Danford, R. D. Dowell, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford. High-resolution Computational Models of Genome Binding Events. *Nature Biotechnology*, 24:963–970, August 2006.

[171] C. Rasmussen. The Infinite Gaussian Mixture Model. In *Neural Information Processing Systems (NIPS)*, 2000.

[172] B. Raught, A. C. Gingras, and N. Sonenberg. The target of rapamycin (TOR) proteins. *Proc. Natl. Acad. Sci. U S A*, 98:7037–7044, 2001.

[173] J. S. Reis-Filho, C. Westbury, and J-Y Pierga. The impact of expression profiling on prognostic and predictive testing in breast cancer. *J. Clin. Pathol.*, 59:225–231, 2006.

[174] B. Ren, F. Robert, J. Wyrick, O. Aparicio, E. G. Jennings, I. Simon, J. Zeitlinger, J. Schreiber, N. Hannett, E. Kanin, T. L. Volkert, C. J. Wilson, S. P. Bell, and R. A. Young. Genome-wide location and function of DNA binding proteins. *Science*, 290:2306–2309, December 2000.

[175] M. Rep, M. Krantz, J. M. Thevelein, and S. Hohmann. The transcriptional response of *Saccharomyces cerevisiae* to osmotic shock. Hot1p and Msn2p/Msn4p are required for the induction of subsets of high osmolarity glycerol pathway-dependent genes. *J. Biol. Chem.*, 275:8290–8300, 2000.

[176] E. J. Richards and S. C. R. Elgin. Epigenetic codes for heterochromatin formation and silencing: Rounding up the usual suspects. *Cell*, 108:489–500, 2002.

[177] F. Robert, D. K. Pokholok, N. M. Hannet, N. J. Rinaldi, M. Chandy, A. Rolfe, J. Workman, D. K. Gifford, and R. A. Young. Global position and recruitment of HATs and HDACs in the yeast genome. *Molecular Cell*, 16:199–209, 2004.

[178] N. G. Robertson, L. Lu, S. Heller, S. N. Merchant, R. D. Eavey, M. McKenna, Jr. Nadol J. B., R. T. Miyamoto, Jr. Linthicum F. H., J. F. Lubianca Neto, A. J. Hudspeth, C. E. Seidman, C. C. Morton, and J. G. Seidman. Mutations in a novel cochlear gene cause DFNA9, a human nonsyndromic deafness with vestibular dysfunction. *Nat Genet*, 20(3):299–303, Nov 1998.

[179] D. Robyr, Y. Suka, I. Xenarios, S.K. Kurdistani, A. Wang, N. Suka, and M. Grunstein. Microarray deacetylation maps determine genome-wide functions for yeast histone deacetylases. *Cell*, 109:437446, 2002.

[180] E. H. Rosch. Natural categories. *Cognitive Psychology*, 4:328–350, 1973.

[181] E. H. Rosch. Cognitive reference points. *Cognitive Psychology*, 7:532–547, 1975.

[182] E. H. Rosch, C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyes-Braem. Basic objects in natural categories. *Cognitive Psychology*, 8:382–439, 1976.

[183] R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8), 2000.

[184] M. Schena. *Microarray analysis*. Wiley-Liss, 2003.

[185] J. Schreiber, R. G. Jenner, H. L. Murray, G. K. Gerber, D. K. Gifford, and R. A. Young. Coordinated binding of NF-κB family members in the response of human cells to lipopolysaccharide. *Proc Natl Acad Sci U S A*, 103(15):5899–5904, Apr 11 2006.

[186] H. J. Schuller. Transcriptional control of nonfermentative metabolism in the yeast *Saccharomyces cerevisiae*. *Curr. Genet.*, 43:139–160, 2003.

[187] A. Schulze and J. Downward. Navigating gene expression using microarrays – a technology review. *Nature Cell Biology*, 3:190–195, 2001.

[188] S. Scott, A. T. Abul-Hamd, and T. G. Cooper. Roles of the Dal82p domains in allophanate/oxalurate-dependent gene expression in *Saccharomyces cerevisiae*. *J. Biol. Chem.*, 275:30886–30893, 2000.

[189] E. Segal, N. Friedman, N. Kaminski, A. Regev, and D. Koller. From signatures to models: understanding cancer using microarrays. *Nat Genet*, 37 Suppl:S38–45, Jun 2005.

[190] E. Segal, N. Friedman, D. Koller, and A. Regev. A module map showing conditional activity of expression modules in cancer. *Nat Genet*, 36(10):1090–8, Oct 2004.

[191] E. Segal, M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genetics*, 34(2):166–76, 2003.

[192] A. J. Sehnert, A. Huq, B. M. Weinstein, C. Walker, M. Fishman, and D. Y. Stainier. Cardiac troponin T is essential in sarcomere assembly and cardiac contractility. *Nat Genet*, 31(1):106–10, may 2002.

[193] J. Sethuraman. A Constructive Definition of Dirichlet Priors. *Statistica Sinica*, 4:639–650, 1994.

[194] R. Shamir, A. Maron-Katz, A. Tanay, C. Linhart, I. Steinfeld, R. Sharan, Y. Shiloh, and R. Elkon. EXPANDER–an integrative program suite for microarray data analysis. *BMC Bioinformatics*, 6:232, 2005.

[195] A. F. Shamji, F. G. Kuruvilla, and S. L. Schreiber. Partitioning the transcriptional program induced by rapamycin among the effectors of the Tor proteins. *Curr Biol.*, 10:1574–1581, 2000.

[196] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nat. Genet.*, 31:64–68, 2002.

[197] Q. Sheng, Y. Moreau, and B. De Moor. Biclustering microarray data by Gibbs sampling. *Bioinformatics*, 19 Suppl 2:II196–II205, Oct 2003.

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

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

[200] S. T. Smale. Core promoters: active contributors to combinatorial gene regulation. *Genes Dev.*, 15:2503–2508, 2001.

[201] P. H. A. Sneath. The application of computers to taxonomy. *J. Gen. Microbiol.*, 17:201–226, 1957.

[202] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisia* by microarray hybridization. *Mol. Biol. of the Cell*, 9:3273–3297, 1998.

[203] T. Strachan and A. P. Read. *Human Molecular Genetics.* John Wiley & Sons, second edition, 1999.

[204] 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, Oct 10 2003.

[205] A. I. Su, T. Wiltshire, S. Batalov, H. Lapp, K. A. Ching, D. Block, J. Zhang, R. Soden, M. Hayakawa, G. Kreiman, M. P. Cooke, J. R. Walker, and J. B. Hogenesch. A gene atlas of the mouse and human protein-encoding transcriptomes. *Proc Natl Acad Sci U S A*, 101(16):6062–7, Apr 20 2004.

[206] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Describing Visual Scenes using Transformed Dirichlet Processes. In *Neural Information Processing Systems (NIPS)*, Vancouver, B.C., 2005.

[207] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Wilsky. Learning Hierarchical Models of Scences, Objects, and Parts. In *International Conf. on Computer Vision*, Beijing, China, 2005.

[208] K. S. Tan, A. G. Nackley, K. Satterfield, W. Maixner, L. Diatchenko, and P. M. Flood. Beta2 adrenergic receptor activation stimulates pro-inflammatory cytokine production in macrophages via PKA- and NF-kappaB-independent mechanisms. *Cell Signal*, 19:251–60, 2007.

[209] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18 Suppl 1:S136–44, 2002.

[210] A. Tanay, R. Sharan, and R. Shamir. *Biclustering Algorithms: A Survey.* Computer and Information Science Series. Chapman & Hall/CRC, 2005.

[211] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 2006.

[212] N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[213] M. Triggiani, A. Petraroli, S. Loffredo, A. Frattini, F. Granata, P. Morabito, R. I. Staiano, A. Secondo, L. Annunziato, and G. Marone. Differentiation of monocytes into macrophages induces the upregulation of histamine H1 receptor. *J Allergy Clin Immunol*, 119:472–81, 2007.

[214] G. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *PNAS*, 98(9):5162–5121, 2001.

[215] R. Wadlow and S. Ramaswamy. DNA microarrays in clinical cancer research. *Curr Mol Med*, 5(1):111–20, Feb 2005.

[216] H. M. Wallach. Topic modeling: beyond bag-of-words. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 977–984, New York, NY, USA, 2006. ACM Press.

[217] A. S. Weinmann, P. S. Yan, M. J. Oberley, T. H. Huang, and P. J. Farnham. Isolating human transcription factor targets by coupling chromatin immunoprecipitation and CpG island microarray analysis. *Genes Dev.*, 16:235–244, 2002.

[218] J. Wells, P. S. Yan, M. Cechvala, T. Huang, and P. J. Farnham. Identification of novel pRb binding sites using CpG microarrays suggests that E2F recruits pRb to specific genomic sites during S phase. *Oncogene*, 22:1445–1460, 2003.

[219] D. L. Wheeler, T. Barrett, D. A. Benson, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. DiCuccio, R. Edgar, S. Federhen, L. Y. Geer, W. Helmberg, Y. Kapustin, D. L. Kenton, O. Khovayko, D. J. Lipman, T. L. Madden, D. R. Maglott, J. Ostell, K. D. Pruitt, G. D. Schuler, L. M. Schriml, E. Sequeira, S. T. Sherry, K. Sirotkin, A. Souvorov, G. Starchenko, T. O. Suzek, R. Tatusov, T. A. Tatusova, L. Wagner, and E. Yaschenko. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res*, 34(Database issue):D173–80, Jan 1 2006.

[220] K. P. White, S. A. Rifkin, P. Hurban, and D. S. Hogness. Microarray analysis of *Drosophila* development during metamorphosis. *Science*, 286:2179–2184, 1999.

[221] Z. Wu and R. A. Irizarry. Stochastic models inspired by hybridization theory for short oligonucleotide arrays. *J Comput Biol*, 12(6):882–93, Jul-Aug 2005.

[222] J. J. Wyrick, J. G. Aparicio, T. Chen, J. D. Barnett, E. G. Jennings, R. A. Young, S. P. Bell, and O. M. Aparicio. Genome-Wide Distribution of ORC and MCM Proteins in *S. cerevisiae*: High-Resolution Mapping of Replication Origins. *Science*, 294:2357–2360, 2001.

[223] G. C. Yuan, Y. J. Liu, M. F. Dion, M. D. Slack, L. F. Wu, S. J. Altschuler, and O. J. Rando. Genome-scale identification of nucleosome positions in *S. cerevisiae*. *Science*, 309:626–630, July 22 2005.

[224] Q. Zhang, M. E. Andersen, and R. B. Conolly. Binary gene induction and protein expression in individual cells. *Theor Biol Med Model*, 3, 2006.

[225] X. J. Zhou and G. Gibson. Cross-species comparison of genome-wide expression patterns. *Genome Biol*, 5(7):232, 2004.

[226] X. Zhu, A. Zhou, A. Dey, C. Norrbom, R. Carroll, C. Zhang, V. Laurent, I. Lindberg, R. Ugleholdt, J. J. Holst, and D. F. Steiner. Disruption of PC1/3 expression in mice causes dwarfism and multiple neuroendocrine peptide processing defects. *Proc Natl Acad Sci U S A*, 99(16):10293–8, Aug 6 2002.