Reverse Engineering Gene Networks Using Genomic Time-Course Data

Andrea Rau

Graduate School ETD Form 9
(Revised 12/07)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Andrea Rau

Entitled Reverse Engineering Gene Networks Using Genomic Time-Course Data

For the degree of Doctor of Philosophy

Is approved by the final examining committee:

Rebecca W. Doerge
Chair

Bruce A. Craig

Florence Jaffrézic

Jayanta K. Ghosh

Yuan (Alan) Qi

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): Rebecca W. Doerge

Approved by: Jun Xie 7/13/2010
Head of the Graduate Program Date

Graduate School Form 20
(Revised 1/10)

PURDUE UNIVERSITY
GRADUATE SCHOOL

Research Integrity and Copyright Disclaimer

Title of Thesis/Dissertation:
Reverse Engineering Gene Networks Using Genomic Time-Course Data

For the degree of Doctor of Philosophy

I certify that in the preparation of this thesis, I have observed the provisions of *Purdue University Teaching, Research, and Outreach Policy on Research Misconduct (VIII.3.1)*, October 1, 2008.*

Further, I certify that this work is free of plagiarism and all materials appearing in this thesis/dissertation have been properly quoted and attributed.

I certify that all copyrighted material incorporated into this thesis/dissertation is in compliance with the United States' copyright law and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and save harmless Purdue University from any and all claims that may be asserted or that may arise from any copyright violation.

Andrea Rau
Printed Name and Signature of Candidate

7/14/2010
Date (month/day/year)

*Located at http://www.purdue.edu/policies/pages/teach_res_outreach/viii_3_1.html

REVERSE ENGINEERING GENE NETWORKS USING GENOMIC TIME-COURSE DATA

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Andrea Rau

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2010

Purdue University

West Lafayette, Indiana

To my parents, for always encouraging me to do my very best,
et à Grégoire, qui a cru en moi depuis le début.

ACKNOWLEDGMENTS

Many people have been a part of my graduate education as advisors, friends, mentors, teachers, and colleagues. Rebecca W. Doerge, first and foremost, has been all of these. Her overwhelming generosity and support over the past five years, and her unwavering commitment to her students, have made it a pleasure to work with her. Thank you for pushing me above and beyond my own expectations. My thanks and appreciation also go to my committee members, Professor Alan Qi, Professor Bruce Craig, Professor Jayanta Ghosh, and Dr. Florence Jaffrézic, for their support and generosity.

J'éprouve un profond respect à l'égard de Jean-Louis Foulley pour ses connaissances de la statistique bayésienne et pour les suggestions substantielles qu'il a formulées tout au long de ce travail. Sans lui, cette thèse n'aurait pas sa forme actuelle. Je tiens aussi à remercier Florence et Jean-Louis de m'avoir accueillie si chaleureusement au sein du bâtiment 211 pendant mes séjours à l'INRA. Merci à tous les deux pour le temps que vous m'avez consacrée et pour le financement que vous m'avez accordée. Je remercie également mes collègues et mes amis de l'INRA qui ont su m'accueillir et me faire sentir chez-moi très rapidement en France, notamment Btissam Salmi, Guillemette Marot, et les autres membres de GABI.

I have had the great fortune of being a member of the RWD research group, with whom I had many productive scientific discussions and more than a few beers. It has been an extraordinary privilege to work with the current and past RWD groups members: Tilman Achberger, Lingling An, Paul Livermore Auer, Doug Baumann, Riyan Cheng, Brian Denton, Alex Lipka, Cheri Ochsenfeld, Gayla Olbricht (and the UWN), Sanvesh Srivastava, Amy Wozniak, and Suk Young Yoo. My very best wishes to all of you in your future endeavors.

The Department of Statistics at Purdue University, headed by Professor Mary Ellen Bock, has provided a friendly and encouraging environment for my graduate studies. My time at Purdue has taught me the pleasure of learning, and the atmosphere of respect and friendship among students and professors alike has made my time here very rewarding. I am especially grateful to the many friends I have made in the Statistics department over the last few years, including Tim Clough, Alexandra Chronopoulou, Meghan Honerlaw, Ryan Martin, Shraddha Mehta, Rochelle Remke, and Patricia Yoshida. Thanks to Regina Becker for her enthusiastic management of the Statistical Consulting Service and Technical Assistance Program, and for the much-needed Wednesday night tennis breaks. I also gratefully acknowledge the staff members in our department for their assistance and kindness. Without the computational resources provided by Doug Crabill and My Truong, as well as their patient help and advice during my stays in France, my graduate work would not have been possible. Thanks also to My for knowing when a spin in the side-car with Pucca would cheer me up.

I would like to express my most heartfelt gratitude to those closest to me for their love and support. Thanks especially to Kelly Underkofler, my best friend for the past nine years, Jenny Kochsiek, whose phone calls got me through the first few years of graduate school, Leah Johnson Ellis, Rachel Kabukala, Beth Raecker, Katie Collins, and Kristen Buchta. I am also thankful to Dean Charles for always being just a phone call away when I needed a friendly graduate student ear. Thanks to my brother Paul, who has a knack for cheering me up and planning surprise visits, and to my sister Kris, my travel companion, closest friend, and mirror image. To my parents, your brain waves got me psyched more than you know. Thank you from the bottom of my heart for your support and encouragement.

Enfin, un remerciement spécial pour Grégoire. Ce travail te doit beaucoup (même si ça n'a pas toujours marché de multiplier par 4!). Merci pour ton soutien quotidien, malgré la grande distance qui nous a séparés jusque-là. Maintenant, une nouvelle aventure commence!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

ABSTRACT

Rau, Andrea Ph.D., Purdue University, August 2010. Reverse Engineering Gene Networks Using Genomic Time-Course Data. Major Professor: R. W. Doerge.

Gene regulatory networks are collections of genes that interact, whether directly or indirectly, with each other and with other substances in the cell. Such gene-to-gene interactions play an important role in a variety of biological processes, as they regulate the rate and degree to which genes are transcribed and proteins are created. By measuring gene expression over time, it may be possible to reverse engineer, or infer, the structure of the gene network involved in a particular cellular process. With the development of microarray and next-generation sequencing technologies, it has become possible to conduct longitudinal experiments to measure the expression of thousands of genes simultaneously over time. However, due to the high dimensionality of gene expression data, the limited number of biological replicates and time points typically measured, and the complexity of biological systems themselves, the problem of reverse engineering networks from transcriptomic data demands a specialized suite of appropriate statistical tools and methodologies.

Two methods are proposed that use directed graphical models of stochastic processes, known as dynamic Bayesian networks, and first-order linear models to represent gene regulatory networks. In the first method, an algorithm is developed based on a hierarchical Bayesian framework for a Gaussian state space model. Hyperparameters are estimated using an empirical Bayes procedure, and parameter posterior distributions determine the presence or absence of gene-to-gene interactions. In the second method, a simulation-based approach known as Approximate Bayesian Computing based on Markov Chain Monte Carlo sampling is modified to the context of gene regulatory networks. Because no likelihood calculation is required, this method

permits inference even for networks where no distributional assumptions are made. The performance of the proposed approaches is investigated via simulations, and both methods are applied to real longitudinal expression data. The two methods, while not comparable, are complementary, and help illustrate the need for a variety of network inference methods adapted for different contexts.

1. INTRODUCTION

The discovery of the molecular structure of deoxyribonucleic acid (DNA) in 1953 by Watson and Crick (1953) was a pivotal moment in the history of molecular biology and modern biotechnology. Since that time, an extraordinary amount of progress has been made in discovering the information encoded by these molecules and how this genetic code relates to phenotypic traits, such as physical characteristics and disease status. In particular, the development of "high-throughput" technologies over the past twenty years has led to an explosive growth of wide-scale studies done with an aim to examine the complete genetic information (the genome) of a variety of organisms. For instance, the introduction of microarrays in the 1990's (Schena et al., 1995, 1996; Lipschutz et al., 1999) enabled scientists to simultaneously assay the expression level of thousands of genes. Sequence-based methods, including serial analysis of gene expression (SAGE) (Velculescu et al., 1995, 1997) and next-generation sequencing (NGS) (Mardis, 2008), have opened the door to sequencing entire genomes, complete sets of transcripts (the transcriptome), and the mineral nutrient composition of an organism (the ionome).

High-throughput platforms are often used to study the behavior of genes during specific biological processes, such as the cell cycle or a response to an external input. However, in spite of the abundance of data output from these technologies, it can be very difficult to unravel the complexity of the chemical dynamics that occur within a cell. One reason for this is that cell development is regulated by well-orchestrated patterns of expression among groups of genes, often referred to as gene regulatory networks (Friedman, 2004; Wilkinson, 2009). These networks are generally believed to govern the rate at which genes in the network are expressed, and often play a critical role in the control of complicated cellular functions. Identifying and understanding

the components of gene regulatory networks is essential to improving our knowledge of how complex biological systems work.

In studies of gene expression, the number of samples (e.g., biological replicates or time points) collected is typically far outweighed by the number of genes observed. For this reason, standard statistical techniques cannot be used to infer gene regulatory networks from transcriptomic data, and a specialized suite of appropriate statistical tools and methodologies is required. In this dissertation, two algorithmic approaches are proposed. They are based in approximate Bayesian methodology (Carlin and Louis, 2000; Beaumont et al., 2002) to infer gene regulatory networks from longitudinal expression data. Because both methods incorporate the joint behavior of a set of genes over time, rather than examining each time point independently, the correlation structure of gene expression between two adjacent time points is maintained and elucidates important information about the network. The first approach develops a method to conduct larger, exploratory analyses of gene regulatory networks where no *a priori* biological information is known (Rau et al., 2010). The second approach focuses on the analysis of small, well-characterized pathways. The methods, while not comparable, are complementary, and help illustrate the need for a variety of methods adapted for different contexts.

1.1 Basics of Genetics: DNA and RNA

DNA is a nucleic acid that acts as the blueprint for the development and functioning of living organisms, and serves as the foundation from which other cellular components are constructed. DNA consists of two long polymers made up of smaller units called nucleotides, and are stored in bundles of varying lengths, known as chromosomes, which vary in number between different organisms. Each DNA molecule has a double-stranded structure in the form of a double helix (Watson and Crick, 1953), where the polymers run anti-parallel to one another. Specifically, the "head" of one strand, known as the 3' end, binds to the "tail", or 5' end, of the other. The

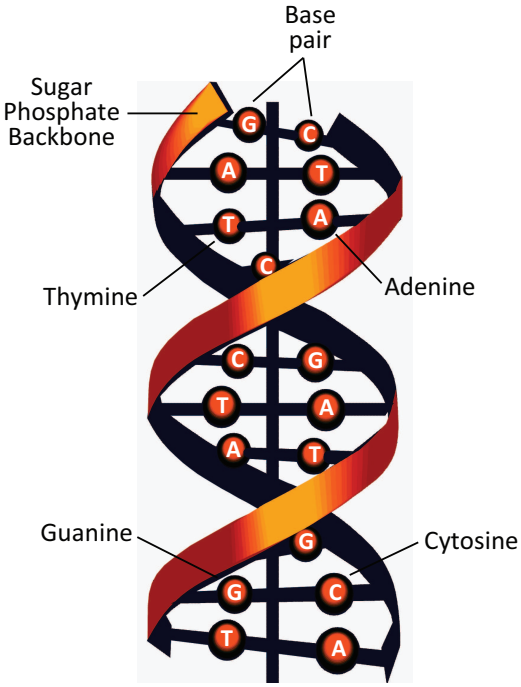


Figure 1.1. The double-stranded structure of a DNA molecule in the form of a double helix. Each strand is made up of a sugar phosphate backbone, where one of four nucleotides (adenine, cytosine, guanine, and thymine) are attached to each sugar. Similar image shown at the National Human Genome Research Institute (2010).

nucleotides forming a DNA strand are made up of a backbone of sugar and phosphate groups joined by ester bonds (Figure 1.1). One of four bases are attached to each sugar: the purines adenine (A) and cytosine (C), and the pyrimidines guanine (G) and thymine (T). Adenine pairs with thymine, and cytosine pairs with guanine. Thus, the two types of possible base pairings are A with T (two hydrogen bonds) and G with C (three hydrogen bonds).

One of the most important functions of DNA is to encode genes that in turn produce proteins, which are compounds of chains of amino acids. Proteins are essential elements in the majority of cellular functions, including biochemical reactions, metabolism, cell signalling, immune responses, and the cell cycle. There are twenty

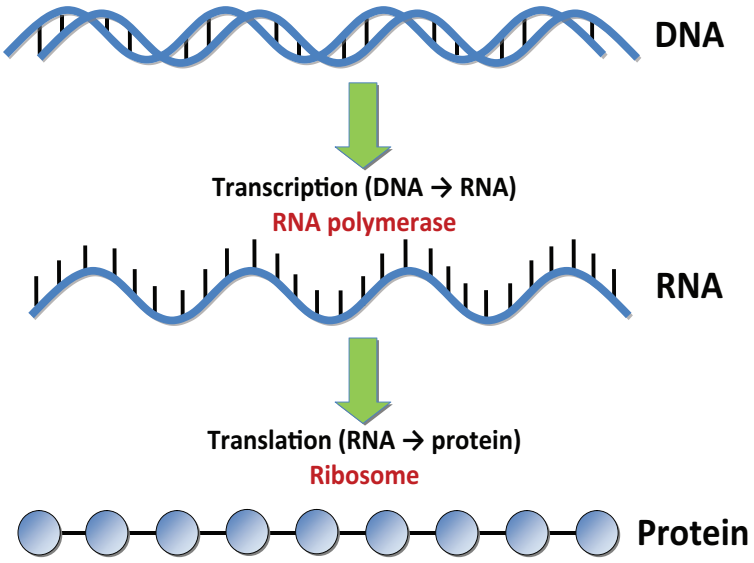


Figure 1.2. The central dogma of molecular biology. Biological information contained in a double-stranded DNA molecule is transcribed into single-stranded RNA by a polymerase, and subsequently translated into proteins by ribosomes.

naturally occurring amino acids (Griffiths et al., 2008) that are encoded by triplets of nucleotides, or codons, in a DNA strand. The transfer of biological information from DNA to proteins is described by the central dogma of molecular biology (Crick, 1970). First, the information encoded in DNA is transcribed by a polymerase into ribonucleic acid (RNA), which in turn is translated into proteins by ribosomes (Figure 1.2). The molecular structure of RNA is similar to that of DNA, with three exceptions: (1) RNA strands are generally single-stranded, and much shorter than DNA, (2) RNA nucleotides contain ribose rather than deoxyribose, and (3) the thymine base present in DNA is replaced by the base uracil (U) in RNA. In total, RNA is comprised of a cap, a start codon to initiate translation, a coding sequence made up of nucleotides arranged into codons, a stop codon to terminate translation, and a trail sequence.

Several types of RNA molecules are active during the process of transcription. After the double strand of DNA is split by the RNA polymerase in the transcription step, a complementary strand of RNA called messenger RNA (mRNA) is formed

to copy information from the DNA and carry it out of the nucleus. The mRNA chain is read by ribosomes made up of proteins and ribosomal RNA (rRNA), which form the machine that can read the information carried in the mRNA and translate it into amino acids. As codons are read off, transfer RNA (tRNA) transfers the corresponding amino acids to a growing polypeptide chain. Once the stop codon is read and a protein has been fully assembled, the mRNA detaches from the ribosome and remains in the cell until it degrades.

1.2 Measuring Gene Expression

A gene is a region of the genome that contains both "coding" sequences that determine function, as well as "non-coding" regions that determine when the gene is active, or expressed. Gene expression refers to the transcription of genes to mRNA, and eventually translation into functional products. Overlooking factors such as RNA degradation and post-translational modifications, the mRNA content of a cell roughly corresponds to the amount of gene expression. As such, the abundance of mRNA is often used as a proxy for a measure of gene expression.

1.2.1 Microarrays

Microarrays are a multiplex technology based on the affinity of single-stranded DNA sequences to bind to complementary sequences of nucleotides. Thousands of "spots", known as features, made up of specific probe sequences are attached to a physical surface, such as a slide, which allows for a genome-wide assay of gene expression. For many organisms and technologies, these probes can be created artificially, based on known gene sequences. Depending on cost, number of genes being studied, and the specific research question, there are a variety of ways that microarrays can be fabricated. In addition to custom-made microarrays prepared and designed by the user, several companies also produce and sell microarrays commerically, including Affymetrix (2010), Illumina (2010), NimbleGen (2010), and Agilent Technologies

Figure 1.3. Examples of commercially-produced microarrays. Clockwise, starting from top left: Affymetrix GeneChip, NimbleGen array, Agilent array, Illumina Microarray Core. In order, images courtesy of the JIC Genome Laboratory (2010), the Leiden Genome Technology Center (2010), the Agilent Technologies Newsroom (2010), and the Microarray Core Facility (2010).

(2010) (see Figure 1.3). Typically, the most substantial distinction among these microarrays lies in the process by which probes are created and affixed to the slide.

Although the fabrication of commercially-produced microarrays varies by company, the basic production steps are fairly standard. First, a glass or silicone slide is prepared, and genetic material, known as probes, is attached in an array of spots to the slide. Because one slide can potentially hold many thousands of spots, a single microarray has the potential to represent the coding regions of an entire genome. After obtaining a tissue sample under the experimental condition of interest, the mRNA in a sample of biological material is extracted and prepared. Complementary DNA (cDNA) is produced from the extracted mRNA through reverse-transcription, and the cDNA is subsequently amplified using a technique known as polymerase chain reaction (PCR) (Mullis et al., 1994). After cleaning the PCR products, the amplified cDNA, known as targets, is marked with fluorescent dye and allowed to bind to its partner on the array via complementation of the previously explained base pairing.

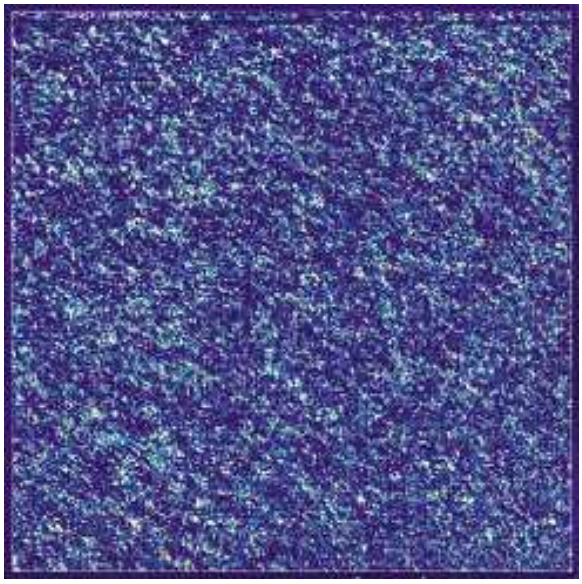


Figure 1.4. An Affymetrix GeneChip probe array. Image courtesy of Giessen Research Center in Infectious Diseases (2010).

The unattached genetic material is then washed off, and the slide is dried. Finally, a laser is used to excite the fluorescent dye attached to the targets, and the fluorescence level of a spot is used as a measure of relative transcript abundance (see Figure 1.4 for an example from an Affymetrix GeneChip).

Once the fluorescence levels of the spots on a microarray have been quantified, an image analysis is used to scan the microarray, recognize spots, remove poor-quality probes, and quantify spot intensities. The raw data must typically undergo several pre-processing steps prior to analysis, including background correction, normalization, and log-transformation. These steps are largely dependent on the type of microarray and the particular experimental design. For instance, each gene on an Affymetrix chip is typically represented by a unique combination of 11 target sequences made up of 25 base pairs, known as a probe set (Affymetrix, 2010). This means that the fluorescence level of probes in a probe set must be combined in a statistically sound way to obtain meaningful estimates of gene expression levels. There is a vast body of literature available on the subject of pre-processing and analyzing microarray data;

see, for example, Kerr and Churchill (2001), Yang et al. (2001), Shaw and Tollett (2001), Kerr (2003), Bolstad et al. (2003), and Smyth and Speed (2003).

Other statistical issues arise when biological or technical replication is included in an experiment. The former refers to the case when tissue samples from different individuals in the same treatment group are hybridized to identical arrays, and gives a measure of the biological variation present in gene expression. The latter refers to the case when identical samples (from the same individual) are hybridized to identical probes several times, whether on the same slide or on different identical slides. This helps measure the technical variation present in a particular experiment, which could arise from the physical preparation of the microarray or the samples.

1.2.2 Serial Analysis of Gene Expression

At roughly the same time that microarrays were introduced, a sequencing-based method for measuring gene expression, known as serial analysis of gene expression (SAGE), was developed (Velculescu et al., 1995, 1997). Although it also measures mRNA abundance, the SAGE protocols and resulting data are quite different from that used to produce microarrays (Serial Analysis of Gene Expression, 2010). The premise of SAGE is that a short transcript fragment (on the order of 10 to 14 base pairs), known as a sequence tag, contains sufficient information to uniquely identify the transcript, given that the tag is from a unique position in the genome. By linking several short sequence tags together to form long molecules, known as concatamers, sequencing machines that can read the nucleotides in DNA may be used to sequence the tags.

SAGE data consist of a list of observed short sequence tags and the number of times each is observed (see Figure 1.5). Typically, sequence databases (e.g., EMBL Nucleotide Sequence Database (2010); Saccharomyces Genome Database (2010); National Center for Biotechnology Information Entrez Genome (2010)), are then used to align the tag to a reference genome, i.e., determine the genomic location for each

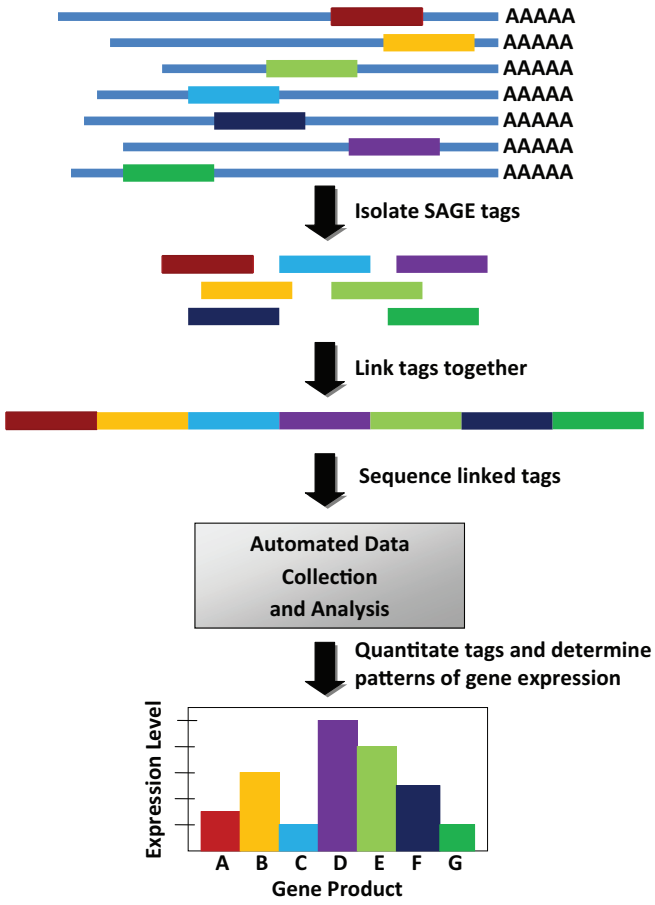


Figure 1.5. Schematic of Serial Analysis of Gene Expression (SAGE) method. After reverse-transcribing a population of mRNA molecules into cDNA, enzymes are used to cut off short segments, known as sequence tags. By isolating the tags and linking them together into long molecules, the tags may be sequenced by machines that can read the nucelotides in DNA. By counting the number of tags that come from a given gene, gene expression is quantified in terms of digital values (counts). Similar image found at the Serial Analysis of Gene Expression (2010) website.

tag. In addition, because the method is sequence-based, mRNA does not need to be known *a priori*, so unknown genes may be studied. SAGE yields digital values (counts) for gene expression, which typically exhibit less background noise than microarray intensity values (which often occur due to the effects of PCR and non-optimal

hybridization). Because the SAGE protocol measures gene expression as counts, and not continuous measurements as with microarray data, these data require an alternative set of statistical methods (Man et al., 2000; Romualdi et al., 2001; Ruijter et al., 2002) based on discrete distributions. For example, Baggerly et al. (2004) suggest an overdispersed logistic regression for SAGE data, and Tĭno (2009) suggests using an Audic-Claverie statistic based on an underlying Poisson distribution.

1.2.3 Next-Generation Sequencing

In the past two years, the rise of next-generation sequencing (NGS) (Mardis, 2008) has revolutionized the fields of genetics, genomics, and epigenomics. One application of NGS, known as RNA sequencing (RNA-Seq) offers an unprecedented improvement in throughput and relative cost compared to the technologies of the previous decade (Cloonan et al., 2008). Like SAGE, the RNA-Seq methodology uses sequencing technology to quantify and map a population of transcripts from entire transcriptomes. However, RNA-Seq is capable of doing so with far greater coverage than SAGE (Morozova et al., 2009; Mortazavi et al., 2008).

The three leading commercially available NGS platforms for RNA-Seq are the Genome Analyzer of Illumina (2010), SOLiD of Applied Biosystems (2010), and the Genome Sequencer FLX of 454 Life Sciences (2010). Although there are differences in how each system works, the basic approach is the same. First, mRNA is isolated from a biological sample, fragmented at random positions, and reverse-transcribed into cDNA. Only fragments within a specified range (e.g., 200-500 bases long for the Genome Analyzer) are retained for PCR amplification. The amplified cDNA is then sequenced using NGS technology, and the reads are mapped back (aligned) to a reference genome. Like SAGE, the data obtained via RNA-Seq is made up of gene counts, or digital gene expression (DGE) measurements. However, RNA-Seq generates data several orders of magnitude larger than SAGE, typically with millions of short reads from a single library of sequences.

As the cost of sequencing continues to fall, the RNA-Seq methodology is expected to replace microarrays for many applications, including measuring gene expression. Because they share similar sampling schemes, methods developed for SAGE data can be adapted for RNA-Seq data when analyzing differential expression (Man et al., 2000; Romualdi et al., 2001; Ruijter et al., 2002; Tĭno, 2009). However, the experimental design and subsequent analysis of RNA-Seq data remains an active area of research (Auer and Doerge, 2010).

1.2.4 Time-Course Gene Expression

It has become increasingly feasible to study gene expression profiles by collecting tissue samples over several time points. In this work, the terms longitudinal, time-course, and time series are used interchangeably to refer to data collected in this fashion. All three of the NGS technologies have potential to assay time-course expression data, even though microarrays are presently the most prominent (and least expensive) platform for such studies. The decreasing cost and refinement of NGS technology suggests that longitudinal expression profiles will likely be studied using RNA-Seq methodology in the near future.

For time-course gene expression data, researchers often study the log-transformed ratio of gene expression at a given time point to that measured at a zero-time point, known as a log-fold change. In typical studies of differential expression, each time point is tested individually, thus losing any information about correlation between time points. However, some analyses do attempt to model the longitudinal nature of such data by examining differentially expressed genes over time (Park et al., 2003; Bar-Joseph et al., 2003), patterns of co-expression among genes (Spellman et al., 1998), clusters of expression profiles (Eisen et al., 1998), or the underlying networks of gene activity over time (Friedman, 2004).

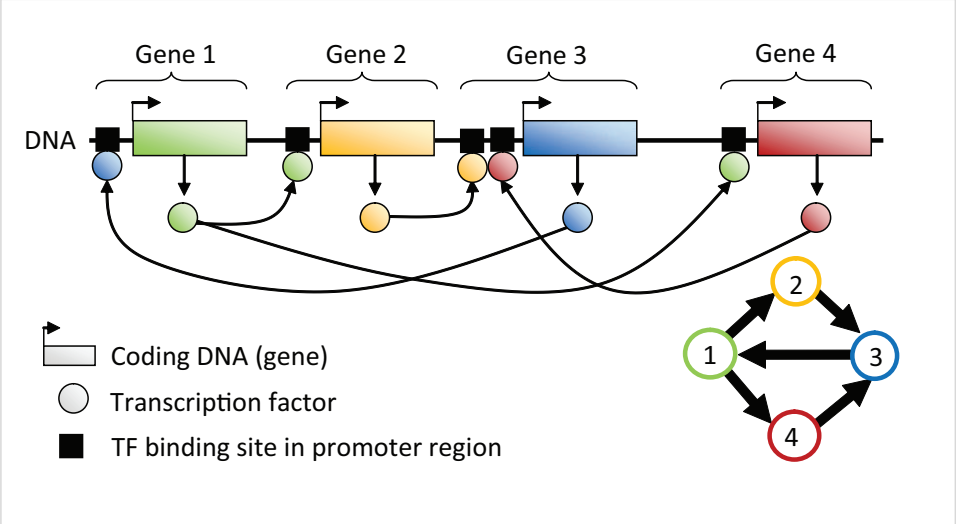


Figure 1.6. A simple gene regulatory network made up of four genes. Each gene is transcribed and translated into a transcription factor protein, which in turn regulates the expression of other genes in the network by binding to their respective promoter regions. The gene regulatory network may be represented using the graph in lower right corner, made up of four nodes (genes) and five edges (interactions among the genes). A similar figure is shown in Schlitt and Brazma (2007).

1.3 Gene Regulatory Networks

Just as gene expression affects the amount of protein in a cell, proteins can in turn regulate gene expression. The proteins that regulate gene expression are known as transcription factors (TF). By binding to the promoter regions of genes, TF control the transfer of information from DNA to mRNA by promoting (activating) or blocking (repressing) RNA polymerase during transcription, which in turn affects the level of gene expression (Figure 1.6). TF are key players in the complex relationships that occur among genes and other cellular products. However, because the abundance of TF in a cell can be difficult to measure experimentally, the expression levels of their corresponding genes are often used as a proxy measure. In this context, a gene regulatory network can thus be described as the interactions that occur (indirectly through mRNA and TF) among a collection of interconnected genes. Gene regulatory

networks are known to be crucial components in cell development, cell maintenance, and cell response for a variety of organisms, including *Saccharomyces cervisae* (yeast) (Spellman et al., 1998), humans (Rangel et al., 2004), and mice (Yaragatti et al., 2009).

Graphs are often used as an abstraction to visualize these networks, where nodes represent genes and edges represent interactions among the genes (Figure 1.6, bottom right). Edges may be directed (Figure 1.6) or undirected. In the former case, an edge from gene A to gene B indicates that A is a regulator of B (that is, that a protein encoded by gene A binds to a promoter region of gene B, thus regulating its expression level). In the latter case, an edge from gene A to gene B is the same as that from gene B to gene A, representing an interaction of a more ambiguous nature. Structurally, gene regulatory networks tend to have several properties in common (Figure 1.7). First, most genes are regulated just one step away from their regulator in a spoke-and-hub type structure, and long regulatory cascades are rare (Alon, 2007). In addition, gene networks tend to be sparse, where genes are regulated by a limited number of other genes (Leclerc, 2008). This characteristic is referred to as the fan-in or in-degree of a particular gene (i.e., the number of regulators for that gene). Finally, sophisticated regulatory circuits, such as positive and negative feedback loops (self-regulating processes) are common motifs in the structure of gene regulatory networks (Brandman and Meyer, 2008).

Identifying the gene-to-gene interactions that are present during a particular biological process can lead to a better understanding of the topology of gene regulatory networks and, ultimately, of the molecular function of each gene. These improved descriptions of the regulatory mechanisms in a cellular system can in turn lead to targeted in silico experiments to investigate and predict the behavior of the system under different conditions. As such, understanding how genes interact with one another, and how these genetic interactions affect changes at the phenotypic level, is currently a major goal in the systems biology community. Two basic types of approaches are often used to this end (Tegnér et al., 2003). At one end of the spectrum

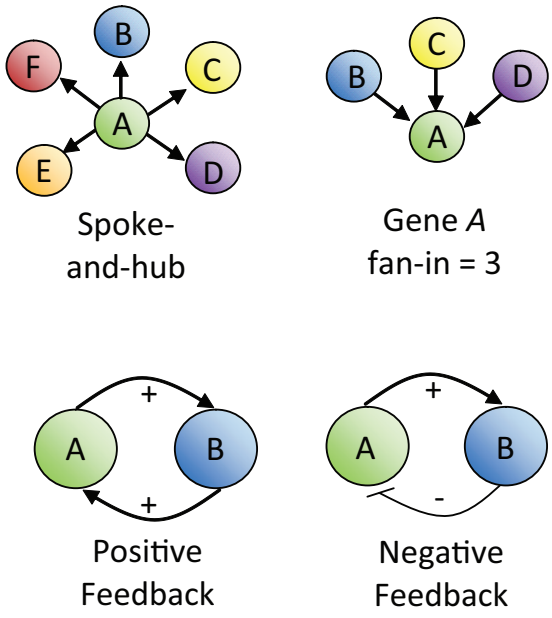


Figure 1.7. Illustration of characteristics of gene regulatory networks. (Top) A spoke-and-hub type structure is displayed on the left, with gene A acting as a central regulator gene. On the right, the fan-in (number of regulators) for gene A is 3. (Bottom) Feedback loops among genes are common motifs in real biological networks. On the left is a positive feedback loop, where gene A activates gene B (represented by an edge with an arrowhead), and gene B in turn activates gene A. On the right, a negative feedback loop exists, as gene A activates gene B while gene B represses gene A (represented by an edge ending in a bar).

is the forward engineering approach, which aims to quantify fundamental equations of gene regulation based on the underlying principles of biochemistry. At the other end of the spectrum is the reverse engineering approach, which attempts to discover the architecture (i.e., the connectivity) of a gene regulatory network from a massive set of gene expression data. For this purpose, gene expression is typically measured in one of two ways: after a specific perturbation (e.g., stress conditions, temperature shifts, and chemical treatments) or within the same organism across time.

Reverse engineering (also referred to as inferring or reconstructing) gene regulatory networks makes use of statistical methods to deduce the architecture of the network

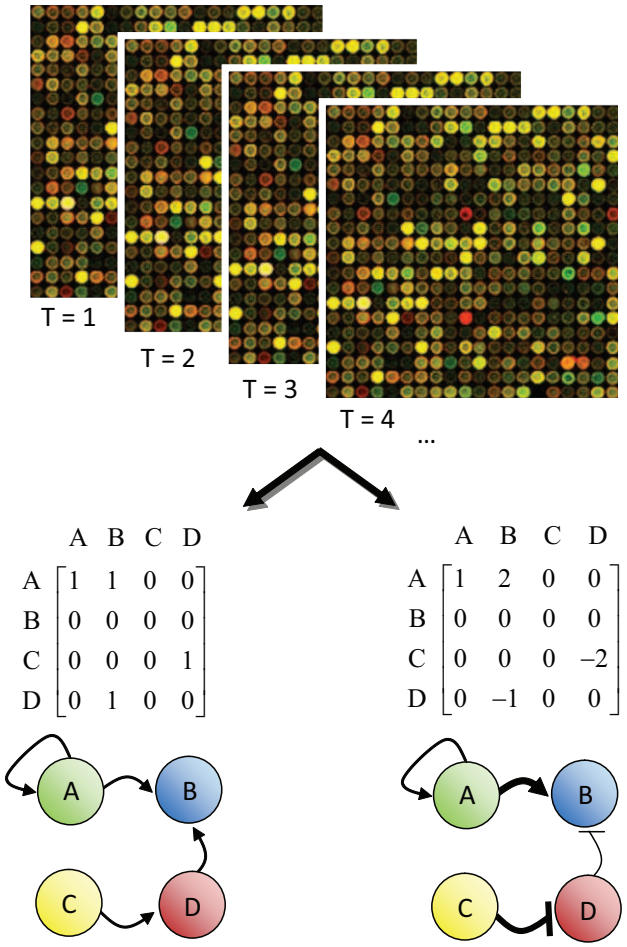


Figure 1.8. An illustration of the process of reverse engineering a gene regulatory network from longitudinal data. High throughput technologies (e.g., microarrays) are used to measure the gene expression in biological samples taken across several time points. Statistical methods may attempt to infer a network adjacency matrix (bottom left), where ones and zeroes indicate the presence or absence of an edge in the graph (i.e., gene-to-gene interaction in the network), respectively. Alternatively, other approaches also include more detailed descriptions of network structure through a parameter matrix (bottom right), where non-zeroes indicate the magnitude and type (activation or repression) of interactions present in the network, and zeroes represent the absence of an interaction. In this representation, thick edges in the graph represent stronger effects, arrowheads activations, and barred lines repressions.

from expression data, whether in terms of an adjacency matrix or a parameter matrix (Figure 1.8). The former consists of a matrix where ones and zeroes indicate the presence or absence of a gene-to-gene interaction in the network (i.e., the presence or absence of an edge between two nodes in the graph). The latter contains additional detailed information about the degree and type (i.e., activation or repression) of each gene-to-gene interaction, based on the magnitude and sign of non-zero elements. That is, the larger the magnitude of a particular element of the parameter matrix, the larger the regulatory effect of a gene-to-gene interaction (and consequently, the thicker the edge in the graph).

Generally, studies of gene expression using high-throughput technologies (microarrays, SAGE, and RNA-Seq) measure information on hundreds or thousands of genes simultaneously over a very limited number of time points and biological replicates. As such, longitudinal studies of gene expression actually amplify the "large p small n" paradigm typical of genomic studies, which renders even more challenging the task of reverse engineering networks from such data. In addition, because genes are interacting with one another as well as reacting to the cellular environment in very sophisticated ways, the structure of a gene regulatory network can inherently be very complex itself. A direct consequence of this network structure is that the resulting expression data often exhibit high multicollinearity. To deal with these challenges, specialized statistical tools and methodologies have been developed to extract relevant information about the relationships among genes from longitudinal gene expression data.

2. METHODS FOR INFERRING GENE REGULATORY NETWORKS

As high-throughput technologies have become increasingly accessible and affordable, the goal of understanding gene regulatory networks has received growing interest in the systems biology community. Forward engineering approaches have made wide use of ordinary differential equations (Zak et al., 2003; Quach et al., 2007; Cao and Zhao, 2008) to estimate the parameters of dynamic models from gene expression data for very small, known networks. Much attention has also been directed to developing methods to reverse engineer network structure from transcriptome data. These methods can be broadly grouped into four classes (Basso et al., 2005): regression techniques, integrative bioinformatics approaches, statistical methods, and optimization methods.

2.1 Current State of Network Inference Methods

2.1.1 Regression Techniques

Several authors (Yeung et al., 2002; Gardner et al., 2003; Rogers and Girolami, 2005) have adapted regression techniques, which have the benefit of allowing rapid, easily scalable calculations, to the task of inferring gene regulatory networks. These methods often make use of data arising from steady-state transcriptional perturbations, such as gene knock-out experiments, rather than time-course expression data (Gardner et al., 2003). Due to both the high dimensionality and multicollinearity of the data, specialized algorithms have been developed to enable parameter estimation and to take advantage of the fact that biological networks are known to be sparse. One such approach (Yeung et al., 2002) applied the singular value decomposition

(SVD) (Eckart and Young, 1936) to approximate the matrix of gene expression measurements by another matrix of lower rank. After producing a condensed description of the data in this way, Yeung et al. (2002) applied a robust regression to identify the sparsest network structure. Rogers and Girolami (2005) proposed a Bayesian regression approach that is naturally sparse, which circumvents the need to pre-define thresholds to determine inclusion of an edge in the graph (i.e., a gene-to-gene interaction in the network). In contrast to these methods, which use a linear model to define the dynamics of particular network Imoto et al. (2002) made use of nonparametric regression to capture nonlinear relationships among genes. Although this approach can more realistically approximate the dynamics of biological systems, it relies on a restrictive assumption of Gaussian distributions, and can be very sensitive to outliers in the data.

Most regression-based methods have been developed for perturbation data, although a few do exist to analyze time-series expression measurements. Opgen-Rhein and Strimmer (2007) estimated the coefficients of a vector autoregressive (VAR) model using an analytic shrinkage approach, and model selection was performed by testing the partial correlations of each gene-to-gene interaction. This approach is very computationally efficient and well-suited to the small sample sizes typical of genomic data, but in practice, it tends to be overly strict in terms of edge selection. More recently Ahmed and Xing (2009) proposed a method to infer time-varying networks based on decoupled regularized regressions for each node in the graph, using a lasso penalty to ensure sparsity and a fused lasso penalty to enforce smoothly changing graphs over time. Both of these proposed approaches lack a characterization of uncertainty in selected edges and overall network structure.

2.1.2 Integrative Bioinformatics Approaches

To gain additional insight into regulatory mechanisms, computational integrated approaches have been suggested to simultaneously assimilate diverse data types into

a single biological model. For example, Niida et al. (2008) proposed an analysis which integrated regulatory sequences, known transcription factor binding motifs, and expression profiles in a study of breast tumor progression. Although such integrative approaches are able to simultaneously make use of a wide range of data types, they are typically implemented using a multi-step process, rather than as a single, statistical model. In addition, this type of approach often relies on mining known information about well-characterized pathways from biological literature or curated databases, both of which may suffer from publishing biases. The result of this is that uncertainty surrounding published gene interactions is not quantified as a measure of confidence in a particular network. Finally, such approaches are limited to biological systems in organisms for which large amounts of data are available.

2.1.3 Statistical Methods

Early statistical approaches for inferring gene regulatory networks from transcriptome data typically relied on exploratory analyses to group together similar genes based on measures of pairwise gene co-expression or correlation (D'haeseleer et al., 2000). Clustering algorithms, which assume the existence of pre-defined groups of genes, were among the first techniques applied to gene expression data. Clustering methods assume that genes sharing similar expression patterns are likely to be involved in the same regulatory network (Fuhrman et al., 2000; Spellman et al., 1998). A further supposition often made is that genes with similar expression profiles share a common biological function. Most clustering algorithms use a matrix of pairwise distance measures as inputs based on correlation (Pan, 2006; Huang and Pan, 2006; Tseng, 2007), mutual information (also referred to as relevance networks) (Butte and Kohane, 2000; Luo et al., 2008) or entropy (Basso et al., 2005; Meyer et al., 2008). Based on these distance measures, genes are categorized into groups such that the distances between observations within a group are small compared to those between observations in different groups.

Clustering methods suffer from several shortcomings. First, distance measures for high-dimensional, highly correlated longitudinal datasets can be difficult to define and are sensitive to pre-processing procedures like normalization and background correction. In addition, random noise in the gene expression measurements is ignored in clustering procedures. Perhaps most importantly, genes that share similar expression profiles over time cannot always be assumed to share biological function, and conversely, genes that share biological function cannot be assumed to share similar expression profiles (Lockhart and Winzeler, 2000). Because such co-expression methods identify pairs of genes with similar profiles rather than those involved in physical interactions, they often exhibit high rates of false positives.

More recently, a simple class of undirected graphical models known as Graphical Gaussian models (GGM) (Schäfer and Strimmer, 2005; Keller et al., 2008; Chiquet et al., 2009), also referred to as gene association networks, have been used to detect conditionally dependent genes. A GGM is defined as follows: the observed data matrix Y with P rows (genes) and N columns (samples) is assumed to be drawn from a multivariate Normal distribution, $N_G(\mu, \Sigma)$ with mean vector $\mu = (\mu_1, \ldots, \mu_P)^T$ and positive definite covariance matrix $\Sigma = (\sigma_{ij})$, where $1 \leq i, j \leq P$. As $\sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$, the covariance matrix can be decomposed into the variance components and Bravais-Pearson correlation matrix $P = (\rho_{ij})$. Under the GGM framework, the inverse of P yields the partial correlation matrix, which characterizes the direct pairwise correlations between genes.

To reconstruct a GGM network the partial correlation matrix is computed by inverting the correlation matrix P, which is typically estimated using the unbiased sample covariance matrix. However, when the sample covariance matrix is not positive definite (which occurs when the number of samples is smaller than the number of variables, as is the case with gene expression data), the standard GGM algorithm is not applicable. In this case, GGM approaches must implement regularized, small-sample estimates of partial correlation and specialized edge inclusion tests, as proposed by Schäfer and Strimmer (2005). Although GGM are conceptually simple,

their applicability in the inference of gene regulatory networks is limited by the fact that the edges in the graph are assumed to follow a Gaussian distribution and are undirected. In addition, GGM assume samples to be independent over time, rather than as a series of correlated data.

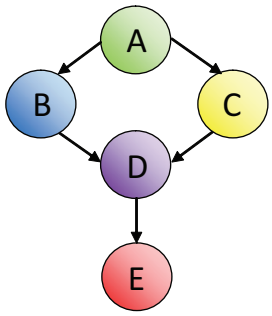
2.1.4 Optimization Methods

Over the past ten years, Bayesian networks (BN) (Pearl, 1988) have become a popular statistical tool used for the inference of gene regulatory networks, due in part to their flexibility and intuitive interpretation. The seminal works of Murphy and Mian (1999) and Friedman (2000) motivated a large body of work dedicated to using Bayesian networks for biological network inference. BN fall in the intersection of graph theory and probability theory, as they use directed graphical models to represent the conditional probabilistic relationships among a set of random variables.

More formally, BN are defined by a graphical structure $\mathcal{G} = \{V, E\}$ made up of a set of random variables (referred to as nodes or vertices) and edges, and a family of conditional probability distributions $\mathcal{F}$ parameterized by Θ (Husmeier et al., 2005). If an edge exists pointing from node V_1 to node V_2 in $\mathcal{G}$, then V_1 is referred to as a parent (or ancestor) of V_2, and V_2 is referred to as a child (or descendent) of V_1 (see Figure 2.1). In a graph with n vertices, let $\mathrm{Pa}^{\mathcal{G}}(V_i)$ denote the set of parents of node V_i in $\mathcal{G}$, for $i = 1, \ldots, n$. The graph $\mathcal{G}$ encodes the Markov assumption, i.e., each node is independent of its non-descendents, given its parents in $\mathcal{G}$. The joint distribution of the graph $\mathcal{G}$ can be written as the product of the conditional distributions as follows:

$$P(V_1, \ldots, V_n) = \prod_{i=1}^{n} P(V_i | \mathrm{Pa}^{\mathcal{G}}(V_i)). \tag{2.1}$$

In a BN, each of the conditional distributions $P(V_i|\mathrm{Pa}^{\mathcal{G}}(V_i))$, or local probability models, must also be defined. The two classes of models typically used for this purpose are a multinomial model or a linear Gaussian model. In the former model, variables must be discretized (e.g., under-expressed, no change, and over-expressed as compared to a control) and the probability of each possible state of the child



$$P(A,B,C,D,E) = P(A)P(B|A)P(C|A)P(D|B,C)P(E|D)$$

Figure 2.1. Example of a Bayesian network $\mathcal{G}$ with five vertices (A, B, C, D, and E) and five edges. The parents of a given vertex V are the set of vertices pointing directly to V via a single edge (e.g., A is a parent of vertices B and C). The descendents of a given vertex V are the set of vertices pointed to by a single edge emanating from V (e.g., B and C are descendents of A). Edges represent conditional relationships among the vertices. For example, the edge from A to B indicates that B is conditionally independent of its non-descendents, given the value of A. The full joint distribution of the graph $\mathcal{G}$ can be written as the product of the independent conditional distributions. Similar image shown in (Husmeier et al., 2005).

variable is calculated, given the state of its parents (Friedman, 2000). Although this model can be quite flexible and can effectively capture non-linear dependencies, the discretization procedure often incurs a substantial loss of information. In the latter model, continuous measurements are used, and linear regression models are learned for each child, given its parents. However, in some cases, the assumption of Gaussian distributions can be overly restrictive for gene expression measurements.

Regardless of the pre-defined local probability model, reverse engineering a BN from data Y generally consists of finding a network $B = \langle \mathcal{G}, \Theta \rangle$ that best corresponds to the data. One common approach is to use a score function to search for an optimal network. Friedman (2000) proposed a local heuristic search based on a greedy, hill-climbing algorithm and Hartemink et al. (2001) used a Bayesian scoring metric to compare proposed structures. As the number of genes (and in turn, the number of

possible interactions) in a particular network increases, these search algorithms are encumbered by the exponential increase in the search space of the model. In addition, artificially choosing a single "optimal" network structure may not be straightforward if several network structures yield similarly high scores.

2.2 Dynamic Bayesian Networks

Although BN are a powerful and flexible tool for inferring network structure, there are three major issues that limit their applicability to inferring networks from time-course data. First, continuous gene expression data must typically be discretized, which incurs a substantial loss of information. In addition, defining threshold levels to use for discretizing data is not straightforward and may penalize genes with naturally small ranges of variation (Friedman, 2000). Second, because a BN must be a directed acyclic graph (DAG), the graphical structure cannot contain any directed cycles (Husmeier et al., 2005). In other words, the network cannot include cycles where all edges point in the same direction (as in the loop on node 3 in Figure 2.3). This is problematic, as sophisticated regulatory circuits, such as feedback loops, are common motifs in biological networks (Brandman and Meyer, 2008). Third, it is possible that expanding the joint probability of two different BN can yield the same factorization if they show alternative ways of describing the same set of independence relationships (Husmeier et al., 2005). This is referred to as equivalence classes (Husmeier et al., 2005), and occurs only if two graphs share the same skeleton (i.e., two different networks differ only in the the direction of an edge) as shown in Figure 2.2.

To deal with these issues, we focus on an extension of BN known as Dynamic Bayesian Networks (DBN) that have been widely applied in the context of gene regulatory networks, e.g., Beal et al. (2005), Husmeier (2003), Ong et al. (2002), Perrin et al. (2003), Rangel et al. (2004), and Zou and Conzen (2005). A DBN uses time-series measurements on a set of random variables to unfold a BN over time, as shown in Figure 2.3, which implies that interactions among random variables occur with a

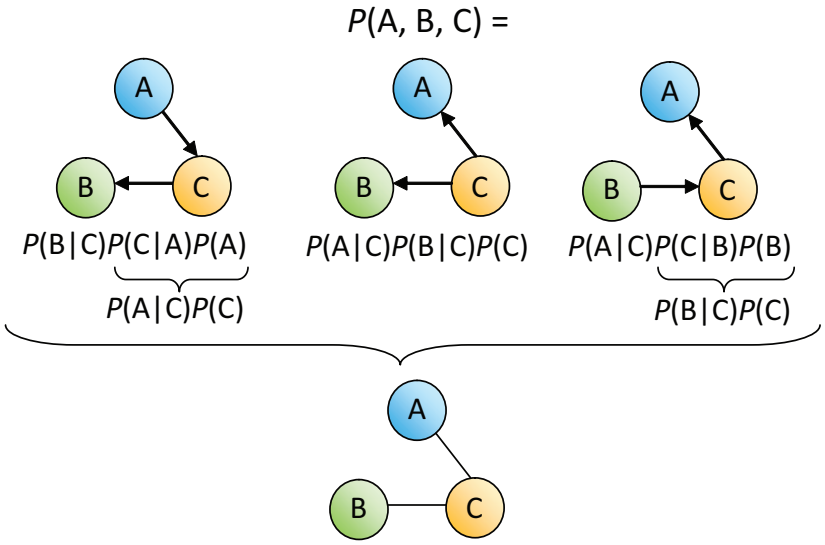


Figure 2.2. The top row shows three distinct Bayesian networks that belong to the same equivalence class, as the expanded probabilities all lead to the same factorization. All three graphs share the same skeleton (represented by the undirected graph in the bottom row) as they differ only in the directions of edges in the network. As before, directed edges represent the conditional relationships among vertices. Similar image shown in Husmeier et al. (2005).

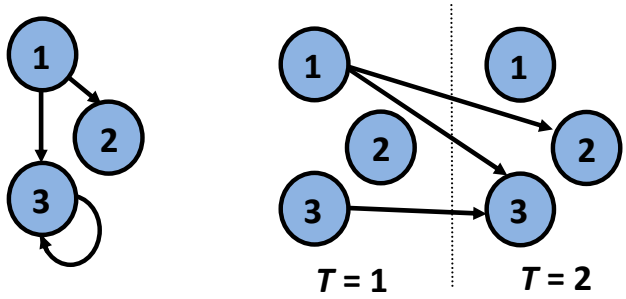


Figure 2.3. (Left) A network with three nodes and three edges, including one feedback loop for node 3. Due to the presence of this feedback loop, this network does not meet the acyclicity constraint of Bayesian networks. (Right) The same network, unrolled over time as a dynamic Bayesian network. By directing arrows with respect to the flow of time, the network shown on the left can be fully represented without violating the acyclicity constraint, despite the presence of a feedback loop (Husmeier et al., 2005).

time delay. To avoid an explosion in model complexity in this setting, the parameters are typically set such that the transition probabilities between time points $t-1$ and t are the same for all t (i.e., a homogeneous Markov model) (Husmeier et al., 2005). As such, gene-to-gene interactions are assumed to be constant across time. In addition, this assumption either implies that biological samples are taken at equidistant time points, or that the dynamics of a biological system are equal across differing time intervals (e.g., the intensity of a biological reaction may decrease over time and be measured over increasing intervals of time). Because edges are directed with respect to the flow of time, the acyclicity graph constraint can be met without eliminating feedback loops, and any ambiguity in the direction of the arrows (i.e., equivalence classes) is resolved (Figure 2.3). In addition, continuous observations may be used in a DBN without the need for discretization. In this work, we focus on the application of two subclasses of DBN for the inference of gene regulatory networks: state space models and autoregressive models.

2.2.1 State Space Models

One special case of DBN is the linear Gaussian state space model (SSM), also referred to as a linear dynamical system (LDS). In some cases, SSM have proven to be well-suited for dealing with time-course gene expression data, as they are able to handle continuous, noisy data and can model the effect of hidden variables (e.g., unmeasured epigenetic factors or genes) on the network dynamics (Rangel et al., 2004; Beal et al., 2005; Rau et al., 2010). Under the SSM framework, a pair of linear equations, known as the state and dynamic equations, is used to relate the expression of genes and a set of hidden states from one time point to the next. In general, these equations can be time-variant (Liang and Kelemen, 2007) or nonlinear (Quach et al., 2007), but we restrict our attention to the linear, time-invariant model.

For time-course gene expression data with P genes, K hidden states, M potential inputs (e.g., known transcription factors), T time points, and R biological replicates,

let $\mathbf{y}_{tr}$, $\mathbf{x}_{tr}$, and $\mathbf{u}_{tr}$ represent the expression of the sets of genes, hidden states, and inputs, respectively, in replicate r at time t. The state and dynamic equations for the classic input SSM are

$$\begin{aligned} \mathbf{x}_{tr} &= A\mathbf{x}_{t-1,r} + B\mathbf{u}_{tr} + \mathbf{w}_{tr} \\ \mathbf{y}_{tr} &= C\mathbf{x}_{tr} + \Theta\mathbf{u}_{tr} + \mathbf{z}_{tr} \end{aligned} \tag{2.2}$$

where $\mathbf{w}_{tr} \sim N(0, I)$ and $\mathbf{z}_{tr} \sim N(0, V^{-1} = \text{diag}(\mathbf{v}^{-1}))$, with $\mathbf{v}$ being a P-dimensional vector of gene precisions, for $t = 1, ..., T$ and $r = 1, ..., R$. Alternatively, to investigate the role of feedback loops in a gene regulatory network, the expression level of genes from a previous time point (i.e., feedback) can be explicitly incorporated as inputs by setting $\mathbf{u}_{1r} = 0$ and $\mathbf{u}_{tr} = \mathbf{y}_{t-1,r}$ for $t = 2, \ldots, T$ (Figure 2.4). In this case, the state and dynamic equations for the feedback SSM are

$$\begin{aligned} \mathbf{x}_{tr} &= A\mathbf{x}_{t-1,r} + B\mathbf{y}_{t-1,r} + \mathbf{w}_{tr} \\ \mathbf{y}_{tr} &= C\mathbf{x}_{tr} + \Theta\mathbf{y}_{t-1,r} + \mathbf{z}_{tr}. \end{aligned} \tag{2.3}$$

For both the input SSM (Equation (2.2)) and the feedback SSM (Equation (2.3)), the primary parameter set of interest in the context of gene regulatory networks is typically contained in the matrix Θ, which encodes the direct TF-to-gene interactions at a given time point (Equation 2.2) or the gene-to-gene interactions from one time to the next (Equation 2.3). Note that in general, state space models are unidentifiable, as the hidden state can be re-scaled and the matrices of the state and dynamic equations adapted accordingly. This implies that two different representations can have equivalent matrices Θ, but different values for the hidden states and matrices A, B, and C. However, since the matrices Θ and $CB + \Theta$ are sub-identifiable (Rangel et al., 2004), inference on the structure of the network based on these matrices is possible. When the matrices of the state and dynamic equations and hidden state dimension K are known, the Kalman filter and smoother (Kalman, 1960) may be used to estimate the values of the hidden states (Bremer and Doerge, 2009). The Kalman filter and smoother are essentially a set of recursive calculations, where the

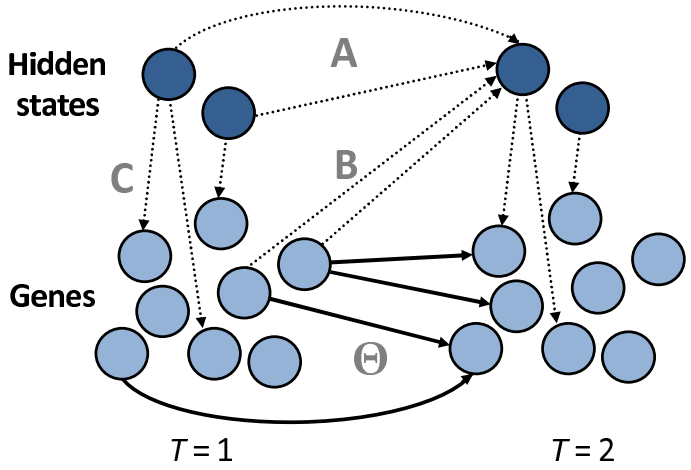


Figure 2.4. A visual representation of the linear feedback state space model, with the observed expression of a set of genes (light blue nodes) and the unobserved expression of a set of hidden states (dark blue nodes) at two time points, $T = 1$ and $T = 2$, where A, B, C, and Θ correspond to the matrices in Equation (2.3). The solid arrows, representing the nonzero elements of Θ, correspond to the direct gene-gene interactions that make up the gene regulatory network.

former consists of a prediction and update step, and the latter smooths the filtered estimations using the full dataset. See Bremer (2006) for additional details about the use of the Kalman filter and smoother in the context of gene expression data.

Because the number of time points and biological replicates in gene expression data are typically much smaller than the number of genes, estimation of model parameters in the SSM requires some care. In addition, choosing the dimension of the state space remains a difficult statistical problem with ramifications to the applicability of state space models in gene regulatory networks. In recent years, several authors have proposed approaches using state space models to reverse-engineer gene regulatory networks. Perrin et al. (2003) applied a generalized Expectation-Maximization (EM) algorithm with a parsimony constraint on network connections to penalize the model likelihood, but limited the choice of the hidden state dimension to 0, 1, or 2. Wu et al. (2004) used a factor analysis and Bayesian Information Criterion (BIC)

penalization for model selection. More recently, Bremer and Doerge (2009) used a SSM model with Kalman smoothing and maximum likelihood estimation techniques to identify regulated genes in time-course gene expression data. Beal et al. (2005) considered a SSM with feedback in a hierarchical Bayesian framework, using a variational Bayes procedure to calculate a bound on the marginal likelihood in order to learn the network structure and the dimensionality of the hidden state. The hierarchical nature of this prior structure is particularly appealing, as its structure allows a shrinkage of the network parameters towards zero, corresponding to the biological assumption of network sparsity.

2.2.2 Auto-Regressive Models

Linear autoregressive (AR) models are another special case of DBN that have proved to be a useful approximation to the complicated network dynamics underlying time-series expression data (Beal et al., 2005; Opgen-Rhein and Strimmer, 2007; Wilkinson, 2009). AR models are finite-order parametric models commonly used in time series analysis. Let y_t denote a set of observations measured on a single variable at time t, $t = 1, \ldots, T$. For notational simplicity we consider only the case with $R = 1$ replicate, but the extension to multiple replicates is straightforward. The notation AR(p) refers to an autoregressive model of order p, which is defined by

$$y_t = \theta_1 y_{t-1} + \theta_2 y_{t-2} + \ldots + \theta_p y_{t-p} + z_t \tag{2.4}$$

where z_t is white noise such that $E(z_t) = 0$, $E(z_t^2) = \sigma^2$, and $E(z_t z_s) = 0$ for $t \neq s$. The model parameters are thus $\theta_1, \ldots, \theta_p$, and σ^2. See Shumway and Stoffer (2000) for a more detailed description of AR models.

In the context of biological networks, expression data represents a multivariate time series $\mathbf{y}_t = [y_{1t}, \ldots, y_{Pt}]^T$ for P genes rather than a univariate time series y_t as in Equation (2.4). A natural extension to the univariate autoregressive model is known as a vector autoregressive (VAR) model, which has received a great deal of attention in the econometrics community (Enders, 2004) to describe the dynamic

behavior of economic and financial time series. The notation VAR(p) refers to a VAR model of order p, defined by

$$\mathbf{y}_t = \Theta_1 \mathbf{y}_{t-1} + \Theta_2 \mathbf{y}_{t-2} + \ldots + \Theta_p \mathbf{y}_{t-p} + \mathbf{z}_t \tag{2.5}$$

where $\mathbf{z}_t$ is a vector of white noise as before, and each Θ_i is a $P \times P$ coefficient matrix. In this work, we focus on the inference of gene regulatory networks using VAR(1) models:

$$\mathbf{y}_t = \Theta \mathbf{y}_{t-1} + \mathbf{z}_t. \tag{2.6}$$

Let θ_{ij} represent the element of Θ in the ith row and the jth column (that is, the element encoding the relationship between gene j at time $t-1$ and gene i at time t). In order to infer a network from multivariate time series data, the objective is to determine the i and j for which $\theta_{ij} = 0$ (the null, or non-existent, edges) and for which $\theta_{ij} \neq 0$ (the true edges), as well as estimates, $\hat{\theta}_{ij}$, for true edges. This combined information yields both the structure of the underlying network, as well as the type of effect (activation versus repression) and magnitude of each gene-to-gene interaction. Note that the VAR(1) model is a simplification of the SSM in Equations 2.2 and 2.3, where $A = B = C = 0$ and the distributional assumption on $\mathbf{z}_t$ is removed.

Although VAR models are popular in econometric analyses, they have seen limited application in the systems biology community for modeling genetic networks, which is likely due, in part, to the difficulty in estimating the model parameters for sparse, high-dimensional data. To deal with this issue, Opgen-Rhein and Strimmer (2007) made use of a VAR model to learn causal networks by implementing a regularized estimation procedure for the coefficients using an analytic shrinkage approach. Fujita et al. (2007) proposed a sparse VAR model which implemented a penalized regression for variable selection. More recently, Shimamura et al. (2009) incorporated the relative importance of VAR coefficients into a regularization method that minimized a penalized loss function with ℓ_1- and ℓ_2-penalties of the coefficients. In a similar approach, Charbonnier et al. (2010) proposed a weighted method based

on an ℓ_1-regularization approach to infer VAR parameters while incorporating prior knowledge about network topology. However, all of these approaches infer a single network structure from gene expression data, without any measure or characterization of inherent variability in estimates of gene-to-gene interactions or overall network architecture.

2.3 Approximate Bayesian Methods for Reverse Engineering Gene Regulatory Networks

Due to the high dimensionality of gene expression data, the limited number of biological replicates and time points typically measured, and the complexity of biological systems themselves, the problem of reverse-engineering networks from transcriptome data demands a specialized suite of appropriate statistical tools and methodologies. We focus on two models that are able to handle continuous, noisy time-course expression data: state space models (Equations 2.2 and 2.3) and vector autoregressive models (Equation 2.6). Both are simple, yet powerful models that make use of linear relationships and first-order Markovian dynamics to describe the patterns of gene expression measurements over time.

In this context, the Bayesian paradigm is particularly well-suited to the inference of gene regulatory networks. First, the number of possible network structures $\mathcal{G}$ increases super-exponentially as the number of genes increases (Husmeier et al., 2005). Because a large number of network structures may yield similarly high likelihoods (due in part to the sparsity of expression data), attempting to infer a single globally optimal structure may be meaningless. Instead, examining the posterior distribution of network structures may be more informative about whether edges in the network can be inferred to have significantly positive or negative values. In addition, a Bayesian framework allows *a priori* knowledge to be encoded in the prior distribution structure. This knowledge can refer to certain features of the network topology (e.g., sparsity in biological networks or the maximum number of regulators per gene)

and to prior biological information about well-characterized pathways gleaned from bioinformatics databases.

The two proposed approaches infer the edges of gene regulatory networks from time-course gene expression data, based on approximate Bayesian methodology (Carlin and Louis, 2000; Beaumont et al., 2002). Both methods incorporate the joint behavior of a set of genes over time, rather than examining each time point independently. This ensures that the correlation structure of gene expression between adjacent time points is maintained and elucidates important information about the network. In the first approach, we develop an empirical Bayes estimation procedure to perform inference (Rau et al., 2010). This method was motivated by that of Beal et al. (2005), based on variational Bayesian learning of state space models. Due to its restrictive distributional assumptions, it is best suited to exploratory analyses of gene regulatory networks where little *a priori* biological information is known. In the second approach, we apply a simulation-based Bayesian method to conduct a detailed analysis of small, well-characterized pathways under fewer model assumptions. By exploiting the capabilities of modern computing, this method makes possible inference on the posterior distribution of gene networks, even in cases where the likelihood is intractable or difficult to calculate. The two approaches, while not comparable, are complementary, and help illustrate the need for a variety of network inference methods adapted for different contexts. Both approaches are discussed in the context of Bayesian inference and rely on approximate Bayesian methodologies, known as empirical Bayes and approximate Bayesian computing.

3. APPROXIMATE BAYESIAN METHODOLOGY

Bayesian inference refers to fitting a probability model $f(Y|\Theta)$ to observed data Y and quantifying uncertainty in the result using probability distributions on the model parameters Θ (Gelman et al., 2004). In this framework, the model parameters are themselves considered to be random variables, following a prior distribution $\pi(\Theta)$. Typically, the model likelihood and prior distributions on the parameters are used to compute the conditional distribution of the parameters given the observed data, known as the posterior distribution. This is done using Bayes' Theorem, proposed by the Reverend Thomas Bayes in the mid-18$^{\text{th}}$ century (Bayes, 1763):

$$\pi(\Theta|Y) = \frac{f(Y|\Theta)\pi(\Theta)}{f(Y)} \propto f(Y|\Theta)\pi(\Theta). \tag{3.1}$$

Two types of approximate Bayesian inference will be discussed, empirical Bayes methods (Carlin and Louis, 2000) and approximate Bayesian computation (ABC) methods (Beaumont et al., 2002).

3.1 Empirical Bayes Methods

In applications where multiple parameters are related due to the nature of the problem, the model structure can often be better represented by a hierarchical Bayes model via sampling parameters from a common population distribution. In such cases, if the prior distribution of Θ itself depends on other parameters ψ, known as hyperparameters, the prior can be written as $\pi(\Theta|\psi)$. The resulting posterior probability given observed data Y is

$$\pi(\Theta|Y,\psi) = \frac{f(Y|\Theta)\pi(\Theta|\psi)\pi(\psi)}{f(Y|\psi)} \propto f(Y|\Theta)\pi(\Theta|\psi)\pi(\psi) \tag{3.2}$$

where $\pi(\psi)$ is the hyperprior on the hyperparameters ψ. This process may be repeated for additional levels of prior distributions and hyperparameters, if required.

Empirical Bayes (EB) methods refer to analyses that use the observed data Y to estimate the hyperparameters ψ of the prior distributions. These approaches can be viewed as an approximation to a complete hierarchical Bayesian analysis (Gelman et al., 2004), since point estimates are used for ψ rather than the whole distribution. EB methods have been in use for nearly 60 years, with the first major work in the area attributed to Robbins (1955). Although this early work defined nonparametric EB methods, we focus this discussion on parametric EB analysis (Efron and Morris, 1972, 1973, 1975). The major difference between the two is that the former approach leaves the prior unspecified, while the latter specifies a parametric family of prior distributions.

Some of the most common parametric EB methods include the Poisson-Gamma model, the Beta-Binomial model, the multinomial-Dirichlet model, and the Gaussian-Gaussian model (Gelman et al., 2004; Carlin and Louis, 2000). To illustrate the application of parametric EB methods, we briefly present the Gaussian-Gaussian model as an example (Casella, 1985; Carlin and Louis, 2000). Suppose we observe P random variables $X = \{X_1, \ldots, X_P\}$ such that each comes from a Gaussian distribution with different means but a common known variance, i.e., $X_i \sim N(\theta_i, \sigma^2)$ for $i = 1, \ldots, P$. Assuming the means θ_i follow a common Gaussian distribution, such that $\theta_i \sim N(\mu, \tau^2)$ for $i = 1, \ldots, P$, it can be shown that the Bayes estimate $\delta^B(X_i)$ for θ_i is a weighted average of the prior estimate μ and the sample estimate X_i:

$$\delta^B(X_i) = \left[\sigma^2/(\sigma^2 + \tau^2)\right] \mu + \left[\tau^2/(\tau^2 + \sigma^2)\right] X_i. \tag{3.3}$$

Rather than pre-specifying values for μ and τ^2, an EB approach estimates these parameters from the data. To do so, first consider the marginal distribution of the data, $f(X_i)$, which is given by

$$f(X_i) \sim N(\mu, \sigma^2 + \tau^2). \tag{3.4}$$

Following from this marginal distribution, it is straightforward to show

$$E(\bar{X}) = \mu \qquad \text{and} \qquad E\left[\frac{(P-3)\sigma^2}{\sum(X_i - \bar{X})^2}\right] = \frac{\sigma^2}{\sigma^2 + \tau^2}. \tag{3.5}$$

These unbiased estimators can be substituted into Equation 3.3 to find $\delta_i^E(X)$, the EB estimator of θ_i:

$$\delta_i^E(X) = \left[\frac{(P-3)\sigma^2}{\sum(X_i - \bar{X})^2}\right]\bar{X} + \left[1 - \frac{(P-3)\sigma^2}{\sum(X_i - \bar{X})^2}\right]X_i. \tag{3.6}$$

Because each $\delta_i^E(X)$ uses information from the full set of observed data, the EB approach takes advantage of the so-called Stein effect (Stein, 1981), which asserts that using combined information can improve estimates for each individual parameter.

EB methods have become a popular choice for the analysis of genomic data, particularly for microarray data (Efron, 2003). EB methods can improve power by borrowing information from the ensemble of genes to assist inference for individual genes (Stein, 1981). Efron et al. (2001) proposed a simple nonparametric EB model to make simultaneous inferences on the differential expression between treatment groups of seven thousand human genes. Lönnstedt and Speed (2002) instead considered a parametric EB approach for the differential analysis of replicated two-color microarrays, based on a mixture of normal models with a conjugate prior. Smyth (2004) further generalized this model to experiments with arbitrary numbers of treatments and RNA samples. In the context of inferring gene regulatory networks, Schäfer and Strimmer (2005) developed an EB estimation procedure for the network topology of a GGM, similar in spirit to that of Efron et al. (2001) for differential expression. Rogers and Girolami (2005) used a sparse Bayesian regression based on a Gaussian linear model for each gene in the network, where point estimates of model hyperparameters based on observed data were obtained and substituted into the model.

3.2 Approximate Bayesian Computation

In cases where the likelihood $L(\Theta|Y) = f(Y|\Theta)$ (Equation 3.1) cannot be calculated, sampling-based approximate Bayesian computation (ABC) methods, also referred to as likelihood-free (LF) methods, can enable Bayesian inference. ABC methods have become increasingly popular in recent years to infer approximate posterior distributions in situations where the likelihood of the model is computationally

Algorithm 3.1 The Rejection Sampler.

0. Set $i = 0$.
1. Sample a candidate parameter vector $\Theta^\star$ from some proposal distribution $\pi(\Theta)$, e.g., a prior distribution.
2. Simulate data $Y^\star$ from the model described by conditional probability distribution $f(\cdot|\Theta^\star)$.
3. If $Y^\star = Y$, accept $\Theta^\star$ and set $i = i + 1$. Otherwise, reject $\Theta^\star$.
4. If $i < N$ (a pre-set number of acceptances), return to 1.

intractable or difficult to evaluate (Beaumont et al., 2002; Marjoram et al., 2003; Ratmann et al., 2007). The power and simplicity of these approaches arise from the exploitation of a simulation-based procedure which takes advantage of the capabilities of modern computing. Although rejection-type techniques, which are explained next, can sometimes enable direct sampling from the posterior distribution, as noted by Rubin (1984), a naive application of these methods can be time-consuming and inefficient. Toward this end, some of the recent adaptations to ABC methods are also discussed. A more detailed summary of these adaptations may be found in Chapter 2 of Grelaud (2009).

3.2.1 Approximate Sampling from the Posterior

At their core, all ABC methods follow the same general form, known as the rejection sampler (Pritchard et al., 1999), shown in Algorithm 3.1. This simple algorithm is based on the idea that rejection techniques can be used to sample exactly from the posterior distribution (Rubin, 1984). The end result of Algorithm 3.1 is not approximate, as it is simulated from the true posterior, $\pi(\Theta|Y) \propto f(Y|\Theta)\pi(\Theta)$. However, particularly in cases where data are continuous rather than discrete, it may be inefficient or impossible to generate simulated data such that $Y^\star = Y$. In these cases, Algorithm 3.1 can be modified to include a distance function ρ and tolerance ϵ to

Algorithm 3.2 The ϵ-Tolerance Rejection Sampler.

0. Set $i = 0$.
1. Sample a candidate parameter vector $\Theta^\star$ from some proposal distribution $\pi(\Theta)$, e.g., a prior distribution.
2. Simulate data $Y^\star$ from the model described by conditional probability distribution $f(\cdot|\Theta^\star)$.
3. Compare simulated data $Y^\star$ to the observed data Y using a distance function ρ and tolerance ϵ. If $\rho(Y^\star, Y) \leq \epsilon$, accept $\Theta^\star$, otherwise reject.
4. If $i < N$ (a pre-set number of acceptances), return to 1.

determine whether simulated and observed data are "close" to one another, as shown in Algorithm 3.2 (Beaumont et al., 2002). This ϵ-tolerance rejection algorithm is approximate when $\epsilon > 0$, and its output amounts to simulating from the prior when $\epsilon \to \infty$. For $0 < \epsilon < \infty$, the algorithm results in a sample of parameters from the distribution $\pi(\Theta|\rho(Y^\star, Y) \leq \epsilon)$. If ϵ is sufficiently small, then this distribution will be a good approximation to the posterior distribution $\pi(\theta|Y)$. However, a balance must be achieved between a small enough tolerance to obtain a good approximation to the posterior and a large enough tolerance to allow for feasible computation time. Typically, the algorithm must be repeated a large number of times (on the order of $N = 1 \times 10^6$ or more), and only parameter values corresponding to the smallest $\alpha\%$ (e.g., 1%) of ϵ are used for inference (Beaumont et al., 2002).

A further improvement (Beaumont et al., 2002) to the ϵ-tolerance rejection sampler (Algorithm 3.2) for high-dimensional data involves replacing the data Y with lower-dimensional quantities (e.g., summary statistics) calculated on the data, denoted $S(Y)$. The distance $\rho(Y^\star, Y)$ in step 3 of Algorithm 3.2 is replaced with a corresponding distance $\rho(S(Y^\star), S(Y))$ between summary statistics computed on the simulated and observed data, respectively, as shown in Algorithm 3.3. The resulting output is a sample of parameters from $\pi(\Theta|\rho(S(Y^\star), S(Y)) \leq \epsilon$. If summary statistics

Algorithm 3.3 The ϵ-Tolerance Rejection Sampler with Summary Statistics.

0. Set $i = 0$.
1. Sample a candidate parameter vector $\Theta^\star$ from some proposal distribution $\pi(\Theta)$, e.g., a prior distribution.
2. Simulate data $Y^\star$ from the model described by conditional probability distribution $f(\cdot|\Theta^\star)$.
3. Calculate summary statistics $S(Y^\star)$ and $S(Y)$ on the simulated and observed data, respectively. Using distance function ρ and tolerance ϵ, if $\rho(S(Y^\star), S(Y)) \leq \epsilon$, accept $\Theta^\star$, otherwise reject.
4. If $i < N$ (a pre-set number of acceptances), return to 1.

S are chosen judiciously (e.g., sufficient or nearly sufficient statistics), reduction of the data can be achieved without negatively impacting the approximation.

3.2.2 Post-Adjustment Techniques

Data simulated using samples $(\Theta^1, \ldots, \Theta^N)$ from the approximate posterior distribution correspond to varying distances from the observed data. To account for this fact, several post-adjustment techniques have been developed to give higher weights to parameters associated with lower discrepancies from the observed data. One approach is to obtain a pointwise estimation of the posterior distribution using kernel density estimation (Parzen, 1962):

$$\hat{\pi}(\Theta|Y) = \frac{\sum_{i=1}^{N} K_\delta(\Theta^i - \Theta) K_\epsilon(||S(Y^\star) - S(Y)||)}{\sum_{i=1}^{N} K_\epsilon(||S(Y^\star) - S(Y)||)} \tag{3.7}$$

where $K_\delta(\cdot)$ and $K_\epsilon(\cdot)$ are kernel functions with bandwidths δ and ϵ, and $||\cdot||$ denotes the ℓ_2-norm. If a uniform kernel is used, all samples are given the same weight and the results are equal to those of the ϵ-tolerance rejection sampler (Algorithm 3.2).

Another post-hoc method to adjust the output from an ABC algorithm is through regression techniques. Beaumont et al. (2002) suggested the use of a local linear regression, which applies the following model:

$$\Theta^i = \alpha + (S(Y^i) - S(Y))'\beta + \xi_i \tag{3.8}$$

where Θ^i is the ith sample, $S(Y^i)$ is a set of summary statistics calculated on simulated data Y^i, $\xi_i \sim N(0, \sigma^2)$, and $i = 1, \ldots, N$. The regression parameters (α, β) are estimated by minimizing the least squares weighted by an Epanechnikov kernel, and kernel density estimation of the posterior is based on the corrected sample $(\tilde{\Theta}^1, \ldots, \tilde{\Theta}^N)$:

$$\tilde{\Theta}^i = \Theta^i - (S(Y^i) - S(Y))'\hat{\beta} \tag{3.9}$$

for $i = 1, \ldots, N$. One issue with kernel density estimation is that the support of the posterior distribution in Equation 3.7 can be larger than that of the prior, since the adjustment potentially allows parameters to be shifted outside of the prior support. To account for this, as well as the strong correlation that may be present among components of summary statistics, Leuenberger and Wegmann (2009) instead approached the issue of post-adjustment using a general linear model (GLM) based on a Gaussian multivariate kernel and ordinary least squares estimates. A more general approach was proposed by Blum and François (2010) using a nonlinear conditional heteroscedastic model and feed-forward neural networks regression models.

3.2.3 Monte Carlo Techniques

In practice, the rejection sampler (Algorithm 3.1) is often very inefficient, as the acceptance rate can be very low when the prior distribution is very different from the posterior distribution. To avoid this issue, several adaptations have been proposed based on Monte Carlo techniques. One class of ABC algorithms based on Monte Carlo methods makes use of importance sampling (Robert and Casella, 2004) to produce weighted samples from the approximate posterior distribution. Beaumont et al. (2009)

Algorithm 3.4 The Metropolis-Hastings Algorithm.

0. Initialize Θ^i, $i = 0$.
1. Propose $\Theta^\star$ according to a proposal distribution $q(\Theta|\Theta^i)$.
2. Set $\Theta^{i+1} = \Theta^\star$ with probability $\alpha = \min\{1, \frac{\pi(\Theta^\star|Y)q(\Theta^i|\Theta^\star)}{\pi(\Theta^i|Y)q(\Theta^\star|\Theta^i)}\}$ and $\Theta^{i+1} = \Theta^i$ with probability $1 - \alpha$.
3. Set $i = i + 1$. If $i < N$ (a pre-set number of iterations), return to 1.

applied an adaptive sequential technique known as Population Monte Carlo (PMC) (Cappé et al., 2004) to the general ABC method in Algorithm 3.1 to improve its efficiency. This approach, known as the ABC-PMC algorithm, uses iterated importance sampling to obtain weighted draws from progressively better proposal distributions. In related work, Sisson et al. (2007) used a Sequential Monte Carlo (Robert and Casella, 2004) technique to improve the efficiency of ABC algorithms (ABC-SMC). This approach simulates a sample of weighted particles from a sequence of target distributions $\{\pi_t\}_{t=1,\ldots,T}$ using a sequence of decreasing tolerances $\epsilon_0 > \ldots > \epsilon_T = \epsilon$. In this way, a population of parameters is propagated through a sequence of intermediary distributions, until it ultimately represents a sample from the approximate posterior distribution.

Another adaptation to the basic framework of Algorithm 3.1 combines the principles of ABC techniques with Markov chain Monte Carlo (MCMC) methods (Gilks et al., 1996). In this context, the goal is to construct a Markov chain that has as its target (stationary) distribution the approximate posterior distribution, $\pi(\Theta|\rho(Y^\star, Y) \leq \epsilon)$. One of the most widely used algorithms for this purpose is known as the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970), shown in Algorithm 3.4. The Metropolis-Hastings algorithm is typically used to sample directly from the posterior distribution $\pi(\Theta|Y)$. However, even in cases where the posterior is not known, the acceptance ratio α in step 2 of Algorithm 3.4 can still be calculated, since

$$\frac{\pi(\Theta^\star|Y)}{\pi(\Theta|Y)} = \frac{f(Y|\Theta^\star)\pi(\Theta^\star)}{f(Y|\Theta)\pi(\Theta)}. \tag{3.10}$$

Algorithm 3.5 ABC-Markov Chain Monte Carlo (ABC-MCMC).

0. Initialize Θ^i, $i = 0$.
1. Propose $\Theta^\star$ according to a proposal distribution $q(\Theta|\Theta^i)$.
2. Simulate data $Y^\star$ from $f(\cdot|\Theta^\star)$.
3. If $\rho(Y^\star, Y) \leq \epsilon$, go to 4, otherwise set $\Theta^{i+1} = \Theta^i$ and go to 5.
4. Set $\Theta^{i+1} = \Theta^\star$ with probability $\alpha = \min\{1, \frac{\pi(\Theta^\star)q(\Theta^i|\Theta^\star)}{\pi(\Theta^i)q(\Theta^\star|\Theta^i)}\}$ and $\Theta^{i+1} = \Theta^i$ with probability $1 - \alpha$.
5. Set $i = i + 1$. If $i < N$ (a pre-set number of iterations), return to 1.

Under certain regularity conditions, the output of MCMC algorithms such as the Metropolis-Hastings algorithm consists of a Markov chain $(\Theta^1, \ldots, \Theta^N)$ that is approximately sampled from the posterior distribution, $\pi(\Theta|Y)$. For additional details about MCMC methods, see Gilks et al. (1996) and Robert and Casella (2004).

In some cases, the calculation of the likelihood $f(Y|\Theta)$ in Equation (3.10) is not feasible. To deal with this, Marjoram et al. (2003) proposed a method applying the standard Metropolis-Hastings scheme (Algorithm 3.4) in the context of approximate Bayesian computation, known as ABC-MCMC (Algorithm 3.5). Note that the ABC-MCMC algorithm does not require the calculation of the likelihood for the Metropolis-Hastings ratio (step 4 of Algorithm 3.5). Under suitable regularity conditions (Marjoram et al., 2003), it can be shown that the stationary distribution of the chain is indeed the approximate posterior distribution, $\pi(\Theta|\rho(Y^\star, Y) \leq \epsilon)$:

Theorem 3.2.1 *The stationary distribution of the chain produced by the ABC-MCMC algorithm (Algorithm 3.5) is* $\pi(\Theta|\rho(Y^\star, Y) \leq \epsilon)$.

Proof Let $r(\Theta \to \Theta^\star)$ be the transition mechanism of the chain. We must check whether $f(\Theta|\rho(Y^\star, Y) \leq \epsilon)r(\Theta \to \Theta^\star) = f(\Theta^\star|\rho(Y^\star, Y) \leq \epsilon)r(\Theta^\star \to \Theta)$, known as the detailed balance equation. Without loss of generality, choose $\Theta^\star \neq \Theta$ such that

$$\frac{\pi(\Theta^\star)q(\Theta|\Theta^\star)}{\pi(\Theta)q(\Theta^\star|\Theta)} \leq 1. \tag{3.11}$$

Then, using the detailed balance equation,

$$
\begin{aligned}
f(\Theta|\rho(Y^\star, Y) \leq \epsilon) r(\Theta \rightarrow \Theta^\star) &= \\
&= f(\Theta|\rho(Y^\star, Y) \leq \epsilon) q(\Theta^\star|\Theta) \mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon|\Theta^\star\right] \alpha(\Theta, \Theta^\star) \\
&= \frac{\mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon|\Theta\right] \pi(\Theta)}{\mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon\right]} \left\{ q(\Theta^\star|\Theta) \mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon|\Theta^\star\right] \times \frac{\pi(\Theta^\star) q(\Theta|\Theta^\star)}{\pi(\Theta) q(\Theta^\star|\Theta)} \right\} \\
&= \frac{\mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon|\Theta^\star\right] \pi(\Theta^\star)}{\mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon\right]} \left\{ q(\Theta|\Theta^\star) \mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon|\Theta\right] \right\} \\
&= f(\Theta^\star|\rho(Y^\star, Y) \leq \epsilon) q(\Theta|\Theta^\star) \mathbb{P}\left[\rho(Y^\star, Y) \leq \epsilon|\Theta\right] \alpha(\Theta^\star, \Theta) \\
&= f(\Theta^\star|\rho(Y^\star, Y) \leq \epsilon) r(\Theta^\star \rightarrow \Theta) \qquad (3.12)
\end{aligned}
$$

■

Bortot et al. (2007) proposed a further adaptation of ABC-MCMC (Algorithm 3.5) to improve its mixing properties using data augmentation techniques, known as the ABC-MCMC augmented algorithm. Specifically, the parameter space is augmented with the tolerance ϵ, which is treated as a model parameter with its own pseudo-prior distribution. Although this algorithm alleviates the problem of insufficient mixing, since larger values of ϵ may be accepted, it typically requires a much larger number of iterations than the ABC-MCMC algorithm.

3.3 Summary

Approximate Bayesian methods, such as empirical Bayes and approximate Bayesian computation methods, have potential to enable inference even in problems where a full Bayesian analysis is not possible. As previously described, the parametric empirical Bayes method fits a hierarchical Bayes model to the data, and allows parameters of the prior distributions to be estimated directly from the observations. For cases where the likelihood cannot easily be computed, the approximate Bayesian computation methods that were introduced enable sampling from an approximate posterior distribution by exploiting the advantages of modern computing. Both of these approximate Bayesian methods are implemented for the purpose of reverse engineering gene

regulatory networks in two different contexts. First, an empirical Bayes estimation procedure for Gaussian state space models (Section 2.2.1) is presented. Second, we develop an alternative approach to inferring networks, based on vector autoregressive models (Section 2.2.2) using the ABC-MCMC algorithm.

4. THE EMPIRICAL BAYES DYNAMIC BAYESIAN NETWORK ALGORITHM

The Empirical Bayes Dynamic Bayesian Network (EBDBN) algorithm is an exploratory approach to reverse engineer the structure of gene regulatory networks from longitudinal gene expression data. The method is motivated by the related work of Beal et al. (2005), in which a variational Bayes procedure was implemented to obtain estimates of the hyperparameters in a feedback state space model under a hierarchical Bayes framework. The novelty of the EBDBN algorithm lies in a computationally efficient estimation procedure for the model hyperparameters, based in empirical Bayes methodology. This approach is best adapted to inferring the structure of moderately sized (e.g., 50-100 genes) networks in the absence of prior biological information about specific edges in the pathway. The method presented here is described in Rau et al. (2010) and has been implemented in the `R` package `ebdbNet`, publicly available on CRAN (R Development Core Team, 2009).

The EBDBN algorithm makes use of a linear state space model (Equations (2.2) and (2.3)) to describe the interactions among a set of genes, a set of hidden states (e.g., unmeasured epigenetic factors or genes), and a set of inputs (e.g., known transcription factors) from one time point to the next. First, we define $Y = \{\mathbf{y}_{tr}\}$, $X = \{\mathbf{x}_{tr}\}$, and $U = \{\mathbf{u}_{tr}\}$, where $1 \leq r \leq R$ and $1 \leq t \leq T$. Let the vectors $\mathbf{y}_{tr} = [y_{1tr}, \ldots, y_{Ptr}]^T$, $\mathbf{x}_{tr} = [x_{1tr}, \ldots, x_{Ktr}]^T$, and $\mathbf{u}_{tr} = [u_{1tr}, \ldots, u_{Mtr}]^T$ represent the expression data of P genes, K hidden states, and M inputs, respectively, in replicate r at time t. The EBDBN algorithm is composed of three principal parts: model selection (choice of the hidden state dimension K), estimation of the hidden states X, and calculation of network posterior distributions. The focus here is on the input state space model

of Equation (2.2), but similar results may be obtained for the feedback state space model of Equation (2.3) by setting $M = P$, $\mathbf{u}_{1r} = \mathbf{0}$, and $\mathbf{u}_{tr} = \mathbf{y}_{t-1,r}$ for $t = 2, \ldots, T$.

4.1 Model Selection

The first step of the EBDBN algorithm involves the choice of the optimal dimension K of the hidden states. Although this is a difficult problem, it is crucial to the application of state space models for network structure recovery. Commonly used criteria for model selection include Akaike's Information Criterion (AIC) (Akaike, 1969) and the Bayesian Information Criterion (BIC) (Schwarz, 1978). Unfortunately, both of these criteria tend to perform poorly for expression data due to the large number of observations and model parameters. Following Bremer (2006) and Aoki and Havenner (1991), we apply a time series method for model selection, based on the autocovariances between observations. This technique shortens computation time considerably, as the algorithm does not run over a wide range of values for K.

Specifically, for each replicate we construct a block-Hankel matrix of autocovariances of the time series gene expression observations

$$H_r = \begin{pmatrix} \hat{\Gamma}_{1r} & \hat{\Gamma}_{2r} & \cdots & \hat{\Gamma}_{mr} \\ \hat{\Gamma}_{2r} & \hat{\Gamma}_{3r} & \cdots & \hat{\Gamma}_{m+1,r} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\Gamma}_{mr} & \hat{\Gamma}_{m+1,r} & \cdots & \hat{\Gamma}_{2m-1,r} \end{pmatrix} \tag{4.1}$$

where $\hat{\Gamma}_{ir} = \frac{1}{T}\sum_{t=1}^{T-i} \mathbf{y}_{tr}\mathbf{y}'_{t+i,r}$ is the autocovariance matrix of the observations $\mathbf{y}_{tr}$ at time t in replicate r for lag i, and m represents the maximum relevant biological time lag between a gene and its regulators. In other words, m is the number of forward time units that a gene is able to influence the expression of other genes. This value must be pre-specified depending on the data under consideration, but in microarray experiments this value is typically small ($m = 1$, 2, or 3), and depends on both the biological process being studied and the time lag between consecutive measurements.

In the absence of error, the rank of H_r equals the number of hidden states K needed to characterize the time series (Aoki and Havenner, 1991). However, gene expression data (e.g., from microarrays) contain both biological and technical errors, meaning the rank of H_r is not exactly equal to K. As such, after finding the singular value decomposition (Eckart and Young, 1936) of H_r, there will be K singular values of "large" amplitude (as discussed below), provided the signal-to-noise ratio (SNR) is also large (SNR $\gg 1$). The SVD for H_r is $H_r = USV'$, where S is a diagonal matrix with diagonal entries $\lambda_1, ..., \lambda_{mP}$ ordered by size, such that $\lambda_1 > ... > \lambda_{mP}$. We scale these singular values by the value of the largest singular value, such that $1 > \frac{\lambda_2}{\lambda_1} > ... > \frac{\lambda_{mP}}{\lambda_1}$. Note that if there are T time points in a particular microarray experiment, only the first $T-1$ singular values will be non-zero.

When plotting the scaled singular values of the block-Hankel matrix, we typically note a rapidly decreasing value for the first singular values, followed by a more moderate decrease. As an illustration, we simulate a small dataset based on the feedback SSM in Equation (2.3) for $P = 10$ genes, $K = 2$ hidden states, $T = 10$ time points, and a single replicate. We set $\mathbf{x}_0 = \mathbf{y}_0 = \mathbf{0}$, take $\mathbf{x}_1 \sim N(0, 1)$, and sample all elements of the matrices A, B, C, and Θ uniformly from (-1,1). Based on these values, we use Equation (2.3) to simulate data forward in a recursive manner for Y and X. Using these data, we construct the block-Hankel matrix H and apply the SVD as previously described. In examining the plot of singular values (Figure 4.1), it can be seen that the "elbow" of the plot falls at the true value of the hidden state dimension, $K = 2$. Intuitively, the SVD reduces H to a small set of singular values which still contain a large fraction of the original variability.

Choosing the number of large singular values results in finding the point at which the inclusion of an additional singular value does not increase the amount of explained variation enough to justify its inclusion (this is similar to choosing the number of components in a Principal Components Analysis). Several "rule of thumb" criteria have been proposed to choose the optimal value of K (Mardia et al., 1980). Some of the best-known criteria to choose K include finding the "elbow" of a plot of the singular

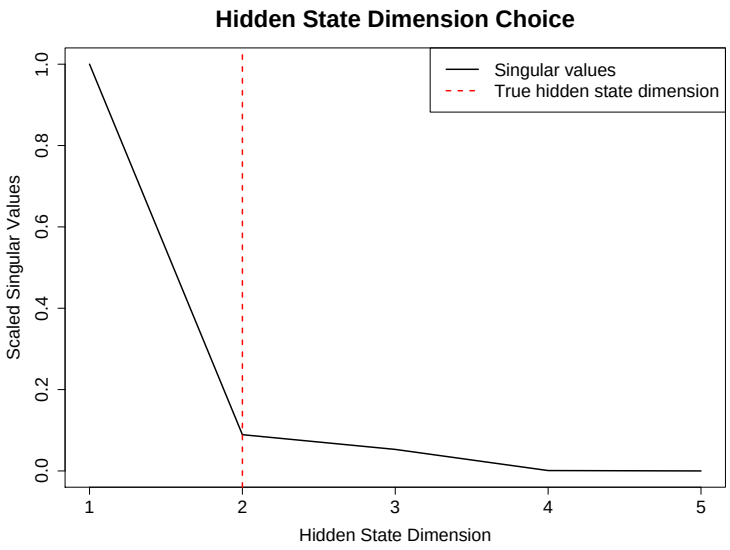


Figure 4.1. The scaled singular values of the block-Hankel matrix H, based on simulated data for $P = 10$ genes with a single replicate, $T = 10$ time points, and a true hidden state dimension of $K = 2$. A dramatic decrease is seen between the first and second singular values of H, followed by a more moderate decrease thereafter. The "elbow" of the graph thus occurs at the true value of $K = 2$.

values as in Figure 4.1, using the Eigenvalue-One criterion of Kaiser (1960), or fixing a cutoff based on percent of total variance explained by the singular values. In this work, we choose the latter criterion, with the cutoff for K chosen to be the smallest number of singular values needed to explain 90% of the total variance. Although some attempts have been made in the literature to formally justify this rather ad-hoc criterion, the justification for such a cutoff rests in its intuitive interpretation and its applicability in practice (Jolliffe, 2002).

4.2 Estimation of Hidden States

The second major component of the EBDBN algorithm is the estimation of the hidden states X. When the matrices A, B, C, Θ, and V in Equations (2.2) and (2.3) are known, a set of recursive calculations known as the Kalman filter and smoother

(Kalman, 1960) may be used to estimate the hidden states (Bremer and Doerge, 2009). The Kalman filter consists of a prediction and update step, and the smoother stabilizes filtered estimates using the full dataset. Specifically, for the input state space model in Equation (2.2), the filter step is

$$\begin{aligned} \hat{\mathbf{x}}_{tr}^{-} &= A\hat{\mathbf{x}}_{t-1,r} + B\mathbf{u}_{tr} \\ \hat{\mathbf{x}}_{tr} &= \hat{\mathbf{x}}_{tr}^{-} + \mathbb{K}(\mathbf{y}_{tr} - C\hat{\mathbf{x}}_{tr}^{-} - \Theta\mathbf{u}_{tr}) \end{aligned} \tag{4.2}$$

where $\mathbf{y}_{tr}$, $\mathbf{x}_{tr}$, $\mathbf{u}_{tr}$, A, B, C, and Θ are as before, $\hat{\mathbf{x}}_{tr}$ represents the filtered estimate of $\mathbf{x}_{tr}$, $\hat{\mathbf{x}}_{tr}^{-}$ represents the *a priori* estimate of $\mathbf{x}_{tr}$ based on the previous time step, and $\mathbb{K}$ is the Kalman gain matrix defined in Kalman (1960). Then, in the smoothing step,

$$\hat{\mathbf{x}}_{tr}^{T} = \hat{\mathbf{x}}_{tr} + \mathbb{J}(\hat{\mathbf{x}}_{t+1,r}^{T} - A\hat{\mathbf{x}}_{tr} - B\mathbf{u}_{tr}) \tag{4.3}$$

where $\hat{\mathbf{x}}_{tr}^{T}$ represents the smoothed estimate of $\mathbf{x}_{tr}$, $\mathbb{J}$ is the Kalman smoothing matrix defined in Kalman (1960), and all other variables are as before. Both the Kalman gain matrix $\mathbb{K}$ and smoothing matrix $\mathbb{J}$ are calculated using the standard formulas (Kalman, 1960).

4.3 Calculation of Posterior Distributions

The final major component of the EBDBN is the estimation of posterior distributions for the model parameters A, B, C, and Θ. To this end, we implement the same hierarchical Bayesian structure as Beal et al. (2005). Let $\mathbf{a}_{(j)}$, $\mathbf{b}_{(j)}$, $\mathbf{c}_{(j)}$, and $\boldsymbol{\theta}_{(j)}$ denote vectors made up of the jth rows of matrices A, B, C, and Θ, respectively. Then

$$\begin{aligned} \mathbf{a}_{(j)}|\boldsymbol{\alpha} &\sim N(0, \text{diag}(\boldsymbol{\alpha})^{-1}) \\ \mathbf{b}_{(j)}|\boldsymbol{\beta} &\sim N(0, \text{diag}(\boldsymbol{\beta})^{-1}) \\ \mathbf{c}_{(i)}|\boldsymbol{\gamma}, v_i &\sim N(0, v_i^{-1}\text{diag}(\boldsymbol{\gamma})^{-1}) \\ \boldsymbol{\theta}_{(i)}|\boldsymbol{\delta}, v_i &\sim N(0, v_i^{-1}\text{diag}(\boldsymbol{\delta})^{-1}) \end{aligned} \tag{4.4}$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_K]^T$, $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_M]^T$, $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_K]^T$, $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_M]^T$, v_i is the ith component of vector $\mathbf{v}$, $j = 1, \ldots, K$ and $i = 1, \ldots, M$. Thus, we have a set of parameters $\{A, B, C, \Theta, \mathbf{v}\}$ and a set of hyperparameters $\psi = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}\}$ describing the *a priori* precisions of the parameter set.

As Gaussian distributions are assumed throughout in the SSM in Equations (2.2) and (2.3) and its corresponding prior structure in Equation (4.4), the joint likelihood may be explicitly written as

$$p(A, B, C, \Theta, \mathbf{v}, Y, X, U) = P(A|\boldsymbol{\alpha})P(B|\boldsymbol{\beta})P(\mathbf{v})P(C|\mathbf{v}, \boldsymbol{\gamma})P(\Theta|\mathbf{v}, \boldsymbol{\delta})\times \\ \times \prod_{t=1}^{T}\prod_{r=1}^{R} P(\mathbf{x}_{tr}|\mathbf{x}_{t-1,r}, \mathbf{u}_{tr}, A, B)P(\mathbf{y}_{tr}|\mathbf{x}_{tr}, \mathbf{u}_{tr}, C, \Theta, \mathbf{v}). \tag{4.5}$$

Let $Z = (Y, X, U, A, B, C, \Theta, \mathbf{v})$ represent the "complete data" and $W = (A, B, C, \Theta)$ represent the "missing data." Because the SSM is in the exponential family, we can write the complete data Z in the form

$$f(Z; \psi) = h(Z)\exp\left\{\sum_{i=1}^{2M+2K} \eta_i(\psi)t_i(Z) - A(\psi)\right\} \tag{4.6}$$

where $h(\cdot)$, $\eta(\cdot)$, and $A(\cdot)$ are known functions, and $t(\cdot)$ denotes the sufficient statistics from the complete-data likelihood. In this case, the Expectation-Maximization (EM) algorithm (Dempster et al., 1977; Bilmes, 1997) can be applied in a straightforward manner to find point estimates $\hat{\psi}$ of the hyperparameters, conditioned on the current estimates $\hat{X}$ and $\hat{\mathbf{v}}$ of the hidden states and gene precisions, respectively.

Specifically, to implement the EM algorithm, we first define the following notation:

$$M = \text{diag}(\boldsymbol{\alpha}) + \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{x}_{t-1,r}\mathbf{x}'_{t-1,r} \qquad N = \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{u}_{tr}\mathbf{x}'_{t-1,r}$$

$$L = \text{diag}(\boldsymbol{\beta}) + \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{u}_{tr}\mathbf{u}'_{tr} \qquad H_j = \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{x}_{t-1,r}x_{jtr}$$

$$S_j = \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{u}_{tr}x_{jtr} \qquad J_i = \text{diag}(v_i\boldsymbol{\gamma}) + \textstyle\sum_{t=1}^{T}\sum_{r=1}^{R} \mathbf{x}_{tr}v_i\mathbf{x}'_{tr}$$

$$G_i = \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{x}_{tr}v_i\mathbf{u}'_{tr} \qquad F_i = \text{diag}(v_i\delta) + \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{u}_{tr}v_i\mathbf{u}'_{tr}$$

$$E_i = \textstyle\sum_{t=1}^{T}\sum_{r=1}^{R} \mathbf{x}_{tr}v_iy_{itr} \qquad Q_i = \textstyle\sum_{t=2}^{T}\sum_{r=1}^{R} \mathbf{u}_{tr}v_iy_{itr}$$

$$\dot{y}_i = \textstyle\sum_{r=1}^{R}\sum_{t=1}^{T} y_{itr}^2$$

$$\Omega_i = Q_i'F^{-1}Q_i + (E_i - GF^{-1}Q_i)'(J - G'F^{-1}G)^{-1}(E_i - GF^{-1}Q_i)$$

where $j = 1, \ldots, K$ and $i = 1, \ldots, P$. Using this notation, the posterior distributions of A, B, C, and Θ, given observed gene expression values Y, hyperparameters ψ, hidden states X, inputs U, and gene precisions $\mathbf{v}$, are as follows:

$$\begin{aligned}
\mathbf{a}_{(j)}|Y, X, U, \mathbf{v}, \psi &\sim N(\mu_j^{\mathbf{a}}, \Sigma^{\mathbf{a}}) \\
\mathbf{b}_{(j)}|Y, X, U, \mathbf{v}, \psi &\sim N(\mu_j^{\mathbf{b}}, \Sigma^{\mathbf{b}}) \\
\mathbf{c}_{(i)}|Y, X, U, \mathbf{v}, \psi &\sim N(\mu_i^{\mathbf{c}}, \Sigma_i^{\mathbf{c}}) \\
\boldsymbol{\theta}_{(i)}|Y, X, U, \mathbf{v}, \psi &\sim N(\mu_i^{\theta}, \Sigma_i^{\theta})
\end{aligned} \tag{4.7}$$

where

$$\begin{aligned}
\mu_j^{\mathbf{a}} &= \Sigma^{\mathbf{a}}(H_j - N'L^{-1}S_j) & \Sigma^{\mathbf{a}} &= (M - N'L^{-1}N)^{-1} \\
\mu_j^{\mathbf{b}} &= \Sigma^{\mathbf{b}}(S_j - NM^{-1}H_j) & \Sigma^{\mathbf{b}} &= (L - NM^{-1}N')^{-1} \\
\mu_i^{\mathbf{c}} &= \Sigma_i^{\mathbf{c}}(E_i - G_i'F_i^{-1}Q_i) & \Sigma_i^{\mathbf{c}} &= (J_i - G_iF_i^{-1}G_i')^{-1} \\
\mu_i^{\theta} &= \Sigma_i^{\theta}(Q_i - G_i'J_i^{-1}E_i) & \Sigma_i^{\theta} &= (F_i - G_i'J_i^{-1}G_i)^{-1}.
\end{aligned}$$

The E-step of the EM algorithm corresponds to finding $t^{(k)} = E_{\psi^{(k)}}\{t|Y, X, U, \mathbf{v}\}$, as follows:

$$\begin{aligned}
E_{\alpha_j^{(t)}}\{A_{\cdot j}'A_{\cdot j}|Y, X, U, \mathbf{v}\} = K(\hat{M} - N'\hat{L}^{-1}N)_{j,j}^{-1} + \\
+ \sum_{l=1}^{K}((\hat{M} - N'\hat{L}^{-1}N)_{j\cdot}^{-1}(H_l - N'\hat{L}^{-1}S_l))^2
\end{aligned} \tag{4.8}$$

$$\begin{aligned}
E_{\beta_m^{(t)}}\{B_{\cdot m}'B_{\cdot m}|Y, X, U, \mathbf{v}\} = K(\hat{L} - N\hat{M}^{-1}N')_{m,m}^{-1} + \\
+ \sum_{l=1}^{K}((\hat{L} - N\hat{M}^{-1}N')_{m\cdot}^{-1}(S_l - N\hat{M}^{-1}H_l))^2
\end{aligned} \tag{4.9}$$

$$\begin{aligned}
E_{\gamma_j^{(t)}}\{C_{\cdot j}'\mathrm{diag}(\mathbf{v})C_{\cdot j}|Y, X, U, \mathbf{v}\} = P(\hat{J} - G\hat{F}^{-1}G')_{j,j}^{-1} + \\
+ \sum_{m=1}^{P}((\hat{J} - G\hat{F}^{-1}G')_{j\cdot}^{-1}(E_m - G\hat{F}^{-1}Q_m))^2 \times \\
\times (a - \frac{P+K}{2}) \times (b + \frac{1}{2}\dot{y}_m - \hat{\Omega}_m)
\end{aligned} \tag{4.10}$$

$$E_{\delta_m^{(t)}}\{\Theta'_{\cdot m}\text{diag}(\mathbf{v})\Theta_{\cdot m}|Y,X,U,\mathbf{v}\} = P(\hat{F}-G'\hat{J}^{-1}G)^{-1}_{m,m}+ \tag{4.11}$$
$$+\sum_{i=1}^{P}((\hat{F}-G'\hat{F}^{-1}G)^{-1}_{m\cdot}(Q_i-G'\hat{J}^{-1}E_i))^2\times$$
$$\times(a-\frac{P+K}{2})\times(b+\frac{1}{2}\dot{y}_i-\hat{\Omega}_i)$$

where we use the convention for an arbitrary matrix U that $U_{j,j}$ indicates the (j,j)th entry of U, $U_{j\cdot}$ the jth row of matrix U, $U_{\cdot j}$ the jth column of matrix U, and $\hat{U}$ indicates matrix U calculated using values $\hat{\psi}^{(t)}$ and $\hat{v}_i^{(t)}$ in the place of ψ and v_i.

For the M-step of the EM algorithm, we must find $\psi^{(k+1)} = \text{argmax}_\psi\{c'(\psi)t^{(k)} - \log(a(\psi))\}$. We maximize each term in the summand individually to find the following:

$$\hat{\alpha}_j^{(t+1)} = \frac{K}{2}\left\{K(\hat{M}-N'\hat{L}^{-1}N)^{-1}_{j,j}+ \right. \tag{4.12}$$
$$\left.+\sum_{l=1}^{K}((\hat{M}-N'\hat{L}^{-1}N)^{-1}_{j\cdot}(H_l-N'\hat{L}^{-1}S_l))^2\right\}^{-1}$$

$$\hat{\beta}_m^{(t+1)} = \frac{K}{2}\left\{K(\hat{L}-N\hat{M}^{-1}N')^{-1}_{m,m}+ \right. \tag{4.13}$$
$$\left.+\sum_{l=1}^{K}((\hat{L}-N\hat{M}^{-1}N')^{-1}_{m\cdot}(S_l-N\hat{M}^{-1}H_l))^2\right\}^{-1}$$

$$\hat{\gamma}_j^{(t+1)} = \frac{P}{2}\left\{P(\hat{J}-G\hat{F}^{-1}G')^{-1}_{j,j}+ \right. \tag{4.14}$$
$$+\sum_{i=1}^{P}((\hat{J}-G\hat{F}^{-1}G')^{-1}_{j\cdot}(E_i-G\hat{F}^{-1}Q_i))^2\times$$
$$\left.\times(a-\frac{P+K}{2})\times(b+\frac{1}{2}\dot{y}_i-\hat{\Omega}_i)\right\}^{-1}$$

Algorithm 4.1 Two-Step EM-Like Estimation of Hyperparameters ψ.

0. At iteration $i+1$, start with $X^{(i)}$ and $\mathbf{v}^{(i)}$, the hidden state and precision estimates from the previous iteration, respectively.
1. Run the EM algorithm with $X^{(i)}$, holding $\mathbf{v}^{(i)}$ constant, until convergence is reached (using a stopping criterion Δ_1). Set the initial hyperparameter estimates to be $\tilde{\psi}^{(i+1)}$.
2. Update $\mathbf{v}$ by calculating the innovation precisions: $\hat{v}^{(i+1)} = \left(\sum_{r=1}^{R}\sum_{t=1}^{T}(\mathbf{y}_{tr} - \hat{C}\mathbf{x}_{tr}^{(i)} - \hat{\Theta}\mathbf{u}_{tr})^2/(RT-1)\right)^{-1}$, where $\hat{C}$ and $\hat{\Theta}$ are the posterior means of C and Θ, given $\tilde{\psi}^{(i+1)}$, $X^{(i)}$, and $\mathbf{v}^{(i)}$.
3. Re-run the EM algorithm with $X^{(i)}$, holding the updated precisions $\hat{\mathbf{v}}^{(i+1)}$ constant, until convergence is reached (using a stopping criterion Δ_2). Set the final hyperparameter estimates to be $\hat{\psi}^{(i+1)}$.

$$\begin{aligned}\hat{\delta}_m^{(t+1)} = \frac{P}{2}\Bigg\{ & P(\hat{F} - G'\hat{J}^{-1}G)^{-1}_{m,m} + \\ & + \sum_{i=1}^{P}((\hat{F} - G'\hat{J}^{-1}G)^{-1}_{m\cdot}(Q_i - G'\hat{J}^{-1}E_i))^2 \times \\ & \times (a - \frac{P+K}{2}) \times (b + \frac{1}{2}\dot{y}_i - \hat{\Omega}_i)\Bigg\}^{-1}.\end{aligned} \tag{4.15}$$

4.3.1 Implementation of the EM Algorithm

In practice, we apply the EM Algorithm detailed in Section 4.3 in two steps within the EBDBN approach, as shown in Algorithm 4.1. First, hyperparameter estimates $\tilde{\psi}$ are stabilized in an initial run of the EM algorithm, based on the current values of the hyperparameters, gene precisions, and hidden states at the end of iteration i. This step is followed by an estimation of the gene precisions $\mathbf{v}$ and a subsequent fine-tuning run of the EM algorithm to obtain final estimates $\hat{\psi}$. We discuss the choice of the stopping criteria Δ_1 and Δ_2 in greater detail later. Because of our unique use

of the EM algorithm in this two-step fashion, we refer to this portion of the EBDBN algorithm as an "EM-like algorithm."

4.3.2 Initial Values

Once the dimension K of the hidden states has been pre-specified or determined (Section 4.1) in the EBDBN algorithm, the hidden states X and hyperparameters ψ must be initialized. For the hidden states, note that a coordinate transformation of X with a nonsingular matrix can give rise to the same joint distribution over a sequence of observed variables Y (Rangel et al., 2005). This identifiability problem can be resolved by constraining the matrices A and C in Equations (2.2) and (2.3) or by restricting interest to either Θ or $CB + \Theta$ (as is the case in our work). For this reason, it is reasonable to initialize the hidden states by setting $\mathbf{x}_{tr}^{(0)} \sim N(0, 1)$ for $t = 1, \ldots, T$ and $r = 1, \ldots, R$. For the hyperparameters, we sample each element of $\psi^{(0)} = \{\boldsymbol{\alpha}^{(0)}, \boldsymbol{\beta}^{(0)}, \boldsymbol{\gamma}^{(0)}, \boldsymbol{\delta}^{(0)}\}$ from $\mathcal{U}(0, 1)$. As the hyperparameters represent the precisions of the matrices A, B, C, and Θ in Equations (2.2) and (2.3), these initial values correspond to initial gene variances of 1 or greater.

4.4 EBDBN Algorithm Iterations

Given that the EBDBN method is composed of three principal parts, model selection (Section 4.1), estimation of the hidden states X (Section 4.2), and calculation of network posterior distributions (Section 4.3), it proceeds as shown in Algorithm 4.2 and Figure 4.2. There are three convergence criteria used in the implementation of the EBDBN algorithm. The first two, Δ_1 and Δ_2 (Algorithm 4.1), are used to determine convergence of the initial and fine-tuning runs of the EM algorithm in the estimation of ψ. The third, Δ_3, is used to determine global convergence of the EBDBN method

Algorithm 4.2 Empirical Bayes Dynamic Bayesian Network (EBDBN).

0. Specify or determine the dimension of the hidden state K (Section 4.1), and initialize ψ and X (Section 4.3.2).
1. (a) Perform EM stabilizing run until convergence criterion Δ_1 is attained.
   (b) Perform EM fine-tuning run until convergence criterion Δ_2 is attained (Algorithm 4.1).
2. Calculate the posterior means of the matrices in the state and dynamic equations (Section 4.3).
3. Obtain estimates of the hidden states using the Kalman filter and smoother (Section 4.2).
4. If global convergence has been attained (Δ_3), go to 5. Otherwise, return to 1.
5. Calculate final estimates of the posterior means of state space matrices, conditioned on hyperparameters and hidden states.

(Algorithm 4.2). In general, for parameter values at the i^{th} and $(i+1)^{\text{th}}$ iteration, these convergence criteria are a distance metric of the form

$$D = \max_{\psi=\{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta}\}} \sqrt{\frac{\sum(\psi^{(i+1)} - \psi^{(i)})^2}{\sum(\psi^{(i)})^2}} \tag{4.16}$$

corresponding to the maximum of these distances for $\psi = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}\}$. That is, when all of the values ψ change very little from one iteration to the next (whether within a run of the EM algorithm or in the larger loop of the full algorithm), approximate convergence is declared and estimation is considered to be complete. Unfortunately, determining the convergence properties of the algorithm is difficult since simulating plausible data based on a state space model with arbitrary fixed hyperparameter values (i.e., the precisions of the state space matrices A, B, C, and Θ) rather than fixed parameter values (i.e., the state space matrices themselves) is difficult. In practice, we assign the values of 0.15, 0.05, and 0.01 to Δ_1, Δ_2, and Δ_3, respectively. Note that the largest value among these three cutoffs is Δ_1, as this criterion is used to terminate the initial run of the EM-like estimation procedure (Algorithm 4.1). Criterion Δ_2 is

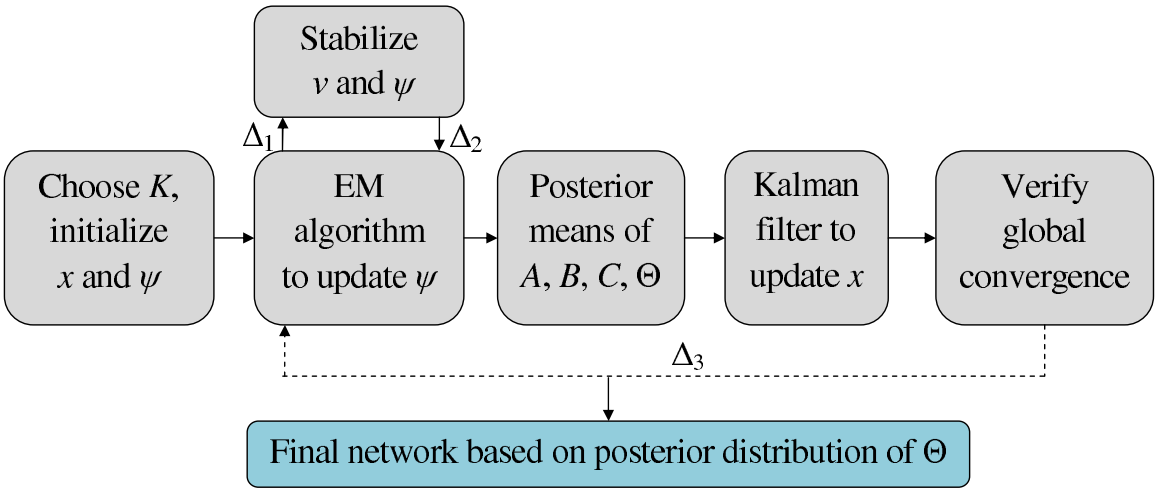


Figure 4.2. Visual representation of the typical workflow of the EBDBN algorithm (Algorithm 4.2). After selecting the hidden state dimension K as in 4.1, two sub-loops of the EM algorithm are used to update model hyperparameters ψ (using convergence criteria Δ_1 and Δ_2). Posterior means of the model parameters and Kalman filter estimates of the hidden states are subsequently calculated as in Sections 4.2 and 4.3. When global convergence is attained (based on convergence criterion Δ_3), the posterior distribution of matrix Θ may be obtained.

somewhat stricter, as it determines the cutoff of the fine-tuning run of the EM-like algorithm, and thus determines the final estimates of the hyperparameters $\hat{\psi}^{(i)}$ at each iteration i. The smallest value is given to Δ_3, as this criterion ultimately determines whether the algorithm has stabilized enough for convergence to be declared. However, the complexity of the state space equation and of the EBDBN algorithm itself make it difficult to ascertain whether convergence can always be attained using these values.

Once the algorithm has converged as determined by the criterion Δ_3, the posterior distributions of the elements of Θ may be calculated. Because the posterior distributions are Gaussian, we can compute the standard z-statistic for normally distributed variables for each edge. Edges whose distributions lie far above the zero point are interpreted as activations, while those far below the zero point as repressions. Con-

Example 4.1 Sample code for `R` package `ebdbNet`.

```
library(ebdbNet)
# Hidden state dimension
K <- hankel(y, ...)$dim
# Run EBDBN algorithm
net <- ebdbn(input = u, y, ...)
z <- zcutoff(net$DPost, net$DvarPost, ...)
```

sequently, to decide which edges are present in the network, we use the standard thresholds $(1.96, 2.58, 3.30)$ for $(95, 99, 99.9)\%$ confidences, respectively.

4.5 `R` package: `ebdbNet`

The EBDBN algorithm (Rau et al., 2010) described in Sections 4.1-4.4 is implemented in the `R` package `ebdbNet`. The package is built around a core of necessary functions, and interfaces `R` with `C` sub-routines. To run the EBDBN algorithm, `ebdbNet` contains three principal functions: 1) The hidden state dimension described in Section 4.1 is chosen using the `hankel` function. 2) Estimates of the hidden states and posterior means are obtained via the `ebdbn` function. 3) The significance of edges is determined using the `zcutoff` function, and the subsequent graph may be displayed using the `visualize` function.

In practice, for observed longitudinal gene expression data `y` with known inputs `u`, the user infers a networks with inputs and hidden states using `R` code similar to that shown in Example 4.1, where ellipses designate additional user-supplied parameters. Alternatively, the call of the function `ebdbn` may be modified to

```
net <- ebdbn(input = "feedback", y, ...)
```

for networks with feedback loops rather than inputs. If no hidden states are to estimated, the user may specify `K <- 0`, rather than using the `hankel` function to

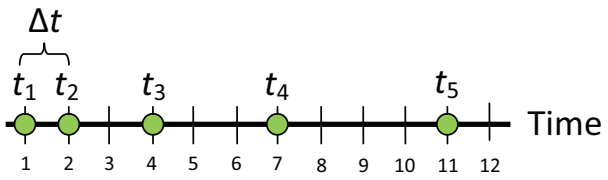


Figure 4.3. For unequally sampled time points $t_1, \ldots, t_5$, the largest common time unit is $\Delta t = 1$.

estimate this parameter. The final network structure may be visualized using the open-source software application Cytoscape (Shannon et al., 2003) or the Bioconductor package `Rgraphviz` (Carey et al., 2005). For both options, `ebdbNet` formats the results accordingly using the function `visualize`, and in the latter case, calls `Rgraphviz` directly to produce a graph in `R`.

4.6 Unequally Spaced Observations and Missing Data

The EBDBN algorithm makes use of the first-order linear SSM in Equations (2.2) and (2.3), which operate under the assumption that time points are equidistant. However, it is often the case in real data that observations are spaced unequally in time. For some experiments, the dynamics of a biological system can be assumed to be equal across different time intervals (e.g., the intensity of a biological reaction may decrease over time and be measured over increasing intervals of time). In such cases, it may be justifiable to use models with equally spaced intervals that are interpreted as equal relative reaction rates.

In cases where longitudinal data are not uniformly sampled across time and reaction rates between time points are not constant, it is possible to adapt Equations (2.2) and (2.3) accordingly. Specifically, define Δt to be the largest common time

unit among the observed intervals (Figure 4.3). Equation (2.2) may be modified to account for differing interval lengths (Wu et al., 2004; Bremer, 2006) as follows:

$$\begin{aligned} \mathbf{x}_{t_k r} &= A^{j_k}\mathbf{x}_{t_{k-1},r} + B\mathbf{u}_{t_k r} + \mathbf{w}_{t_k r} \\ \mathbf{y}_{t_k r} &= C\mathbf{x}_{t_k r} + \Theta\mathbf{u}_{t_k r} + \mathbf{z}_{t_k r} \end{aligned} \tag{4.17}$$

where $j_k = \frac{t_k - t_{k-1}}{\Delta t}$ is an integer for $k = 1, \ldots, T$. Similarly, Equation (2.3) may be modified as follows:

$$\begin{aligned} \mathbf{x}_{t_k r} &= A^{j_k}\mathbf{x}_{t_{k-1},r} + B^{j_k}\mathbf{y}_{t_{k-1} r} + \mathbf{w}_{t_k r} \\ \mathbf{y}_{t_k r} &= C\mathbf{x}_{t_k r} + \Theta^{j_k}\mathbf{y}_{t_{k-1} r} + \mathbf{z}_{t_k r} \end{aligned} \tag{4.18}$$

where j_k is as before. Subsequently, the EBDBN algorithm would proceed as before, where the estimation of the hidden states (Section 4.2) would replace matrices A, B, and Θ with A^{j_k}, B^{j_k}, and Θ^{j_k} as appropriate.

Another issue that often arises with real data is the presence of missing data for some gene expression measurements. One potential solution would be to replace missing $\log_2$ signal ratios with the mean gene expression value across time points, although this ignores any correlation structure in the data across time points. Troyanskaya et al. (2001) suggested two alternatives for missing value estimation in longitudinal microarray data based on singular value decomposition and k-nearest neighbor averages. The former approach has proven to yield satisfactory results for time-course expression data, particularly when moderate to low noise is present and genes exhibit strong patterns over time. Shumway and Stoffer (1982, 2000) developed a modified SSM that can account for missing values in the observed data by splitting $\mathbf{y}_{tr}$ into an observed part $\mathbf{y}_{tr}^{(1)}$ and a missing part $\mathbf{y}_{tr}^{(2)}$, and partitioning the hidden state covariance matrix accordingly. Once missing data have been appropriately acknowledged, whether through imputation (Troyanskaya et al., 2001) or model modifications (Shumway and Stoffer, 1982, 2000), the EBDBN algorithm may be implemented as previously described.

4.7 Simulations

Many authors of inference approaches for gene regulatory networks (Rangel et al., 2004; Beal et al., 2005; Opgen-Rhein and Strimmer, 2007) assess the performance of their methods by comparing results based on real expression data to known genetic regulatory interactions from bioinformatics databases. This type of validation can be problematic for two reasons (Husmeier, 2003). First, at present no real gold standard gene regulatory network exists, and spurious gene-to-gene interactions may be included in the biological literature. Second, the absence of a particular gene-to-gene interaction in the literature does not necessarily indicate that such a relationship is known to be absent. The consequence of this is that it can be very difficult to determine whether a detected edge by a given method is truly a false positive, or whether the absence of a given edge is a true negative.

For these reasons, the importance of using simulation studies for network inference methods has been stressed by several authors (Husmeier, 2003; Rogers and Girolami, 2005; Rice et al., 2005). However, simulating expression data arising from a given gene regulatory network is far from straightforward. Two standard approaches are typically adopted to simulate such data. The first approach is to use the model underlying a particular inference method to simulate data (i.e., in our case, a state space or VAR(1) model). Although in some cases these "model-based simulations" can lead to a good approximation of the general nature of complicated biological systems, they often represent an over-simplified version of reality and lead to overly optimistic results. The second approach uses ordinary differential equations based on biochemical equations to simulate more realistic data. The drawback of such "data-based simulations" is that they can only model well-known networks, and they produce much smaller datasets than would typically be used for network inference.

In these simulations, we consider both model-based and data-based simulations. For each, there are several characteristics of reverse engineering approaches that relate to their success in modelling gene regulatory networks. It is important to consider

a wide range of comparison criteria in such studies, as network inference methods that perform well based on one often underperform in others (Wessels et al., 2001). As such, the consequence of using a single criterion can lead to a biased view of the performance of a given method. In order to thoroughly understand the advantages and limitations of different inferences methods, a more global perspective on performance is required. To this end, we consider a variety of comparison criteria, including the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve, the sensitivity, specificity, and positive predictive value at a given threshold, and the computational time. For these simulations, let TP denote the number of true positives, FP represent the number of false positives, and FN and TN denote the number of false and true negatives, respectively. We define the following three criteria as follows: a) Sensitivity = TP/(TP + FN), b) Specificity = TN/(TN + FP), and c) Positive Predictive Value (Precision) = TP/(TP + FP).

4.7.1 Methods for Comparison

In these simulations, the EBDBN method is compared to two other pre-established methods developed to infer gene feedback networks from longitudinal gene expression data: the Variational Bayes State Space Model (VBSSM) of Beal et al. (2005) and the shrinkage model of Opgen-Rhein and Strimmer (2007). Like the EBDBN, the VBSSM method of (Beal et al., 2005) describes the interactions among a set of genes and hidden states using the linear feedback SSM of Equation (2.3) and the conjugate hierarchical prior structure of Equation (4.4). However, they use a variational Bayes algorithm to estimate the posterior distribution of the network. This approach exploits Jensen's inequality to determine a lower bound for the marginal likelihood

of the observed data Y for a model m with parameters Ψ, by introducing a free distribution $q(X, \Psi)$ as follows:

$$\begin{aligned}\ln p(Y|m) &= \ln \int p(Y, X, \Psi|m) dX d\Psi \\ &= \ln \int q(X, \Psi) \frac{p(Y, X, \Psi|m)}{q(X, \Psi)} dX d\Psi \\ &\leq \int q(X, \Psi) \ln \frac{p(Y, X, \Psi|m)}{q(X, \Psi)} dX d\Psi. \end{aligned} \tag{4.19}$$

Because maximizing the lower bound with respect to $q(X, \Psi)$ is computationally intractable, Beal et al. (2005) instead maximize a simpler, factorized approximation $q(X, \Psi) = q_X(X) q_\Psi(\Psi)$:

$$\begin{aligned}\ln p(Y|m) &\leq \int q_X(X) q_\Psi(\Psi) \ln \frac{p(Y, X, \Psi|m)}{q_X(X) q_\Psi(\Psi)} dX d\Psi \\ &= \mathcal{F}_m(q_X(X), q_\Psi(\Psi), Y). \end{aligned} \tag{4.20}$$

Then $\mathcal{F}$ is iteratively maximized with respect to $q_X(X)$ and $q_\Psi(\Psi)$ using a Variational Bayesian EM algorithm, and edges are chosen based on their standard z-statistics as in Section 4.4.

The shrinkage model of Opgen-Rhein and Strimmer (2007) uses a simple first-order vector autoregressive (VAR) model as in Equation (2.6) to characterize gene-gene interactions over time. As such, no hidden states or driving inputs are incorporated. Due to the high dimensionality of the data, the authors proposed a method to obtain a regularized estimator of the covariance matrix $\mathbf{z}_t \mathbf{z}_t'$ by shrinking the empirical correlations towards zero and the empirical variances against their median. Edge selection is subsequently performed by identifying significant partial correlations using a local False Discovery Rate (FDR) approach (Efron, 2005).

Finally, in addition to the EBDBN, VBSSM, and VAR, we also consider the EBDBN method where no hidden states are estimated, denoted EBDBN(-); this amounts to a SSM model where $A = B = C = 0$, Θ is estimated using the EM-like algorithm of Section 4.3, and no Kalman filter/smoother step is applied. This model closely resembles the VAR model of Equation (2.6), with the added assumption of

normally distributed error terms $\mathbf{z}_t$. We refer to the EBDBN method with hidden states incorporated as EBDBN(x) in general, and EBDBN($K = k$) for a particular hidden state dimension k.

4.7.2 Model-Based Simulations

We first simulate gene expression data based on a first-order autoregressive model (which is the same model for all of the methods under consideration). While this model undoubtedly oversimplifies the dynamics of real microarray data, it is a useful starting point that allows for straightforward simulation with a known network structure. To simulate data, we choose $P = 53$ genes, and vary both the number of replicates R and the number of time points T to be $\{5, 10, 15\}$. The $P \times P$ matrix Θ, representing the direct gene-gene interaction matrix in the feedback network, is simulated such that 10% of its elements follow either a $U(-1, -0.2)$ or $U(0.2, 1)$ distribution and 90% of its elements equal 0. We initialize the gene expression values at time $t = 1$ to be $\mathbf{y_t} \sim N(0, 0.1)$ and simulate subsequent times $t = 2, ..., T$ following a VAR(1) model, such that $\mathbf{y_t} \sim N(\Theta \mathbf{y_{t-1}}, 0.1)$. We then remove the 3 genes with the most outward connections, denoted $\{\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, \tilde{\mathbf{y}}_3\}$, from the dataset. This set of genes play the role of the $K = 3$ hidden states. The goal in these simulations is to infer the remaining structure of the gene network, $\Theta^\star = \Theta \setminus \{\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, \tilde{\mathbf{y}}_3\}$ using only the 50 genes in the "observed" data $\mathbf{y}^\star = \mathbf{y} \setminus \{\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, \tilde{\mathbf{y}}_3\}$.

First, consider the results for the AUC of the ROC curve, which can be interpreted as the average probability that a randomly chosen edge is correctly characterized as present or not, assuming any given edge is equally likely to be present or not (Beal et al., 2005). The simulation results are presented in Figure 4.4. Several trends are apparent: first, for small datasets (that is, few replicates and time points), all methods perform poorly. For the EBDBN(x), EBDBN(-), and VBSSM, increasing either R, T, or both improves performance; interestingly, for the VAR method this is true when R is increased while the inverse holds for increasing T. Furthermore, the VBSSM

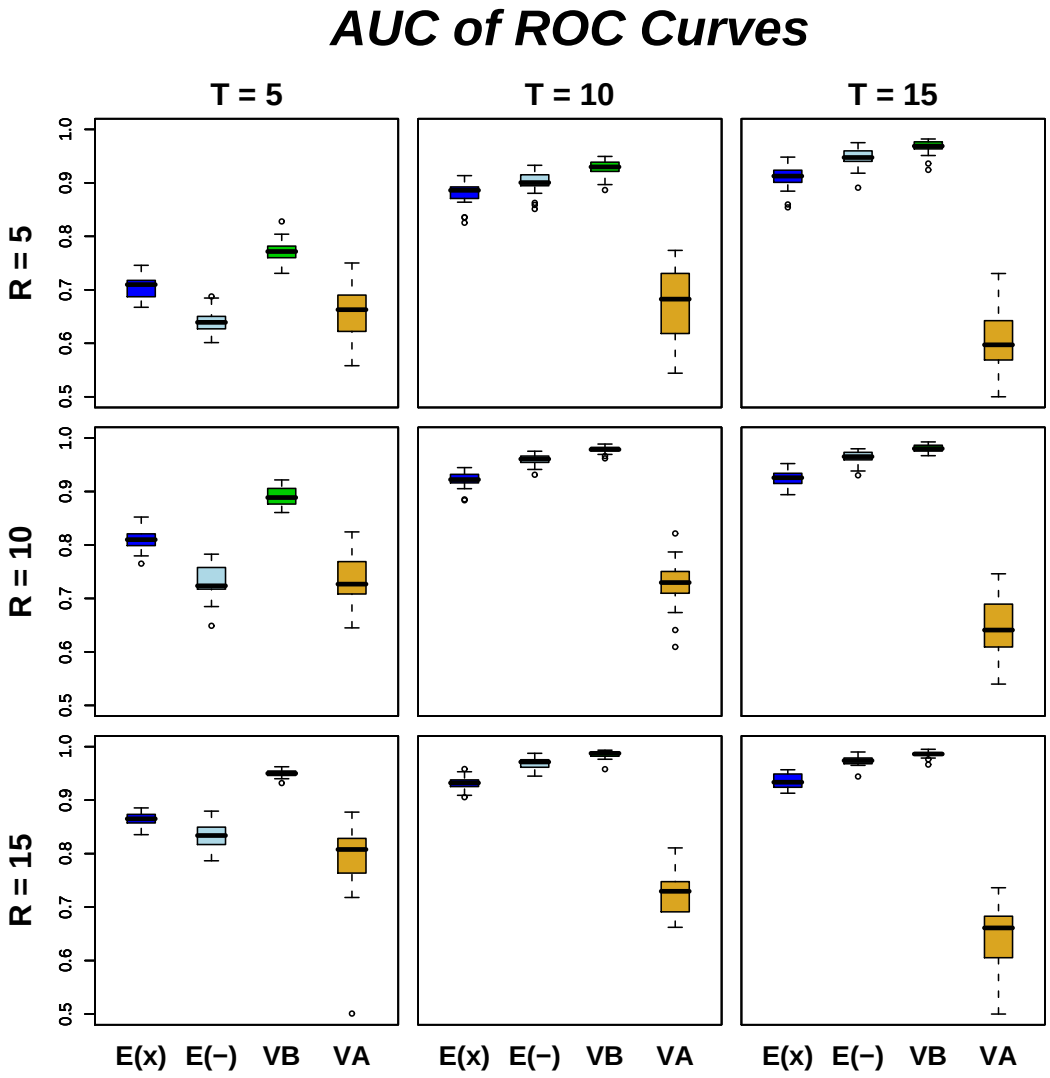


Figure 4.4. Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve of model-based simulations, by number of replicates and number of time points simulated in the data. Each row of the graphical matrix corresponds to the number of replicates ($R = 5, 10, 15$) and each column to the number of time points ($T = 5, 10, 15$), with 25 datasets simulated per evaluation. Within each individual plot, the methods represented (from left to right) are as follows: E(x) = Empirical Bayes Dynamic Bayesian Network (EBDBN) method with hidden states (dark blue), E(-) = EBDBN method without hidden states (light blue), VB = Variational Bayes method (green), and VA = VAR method (yellow).

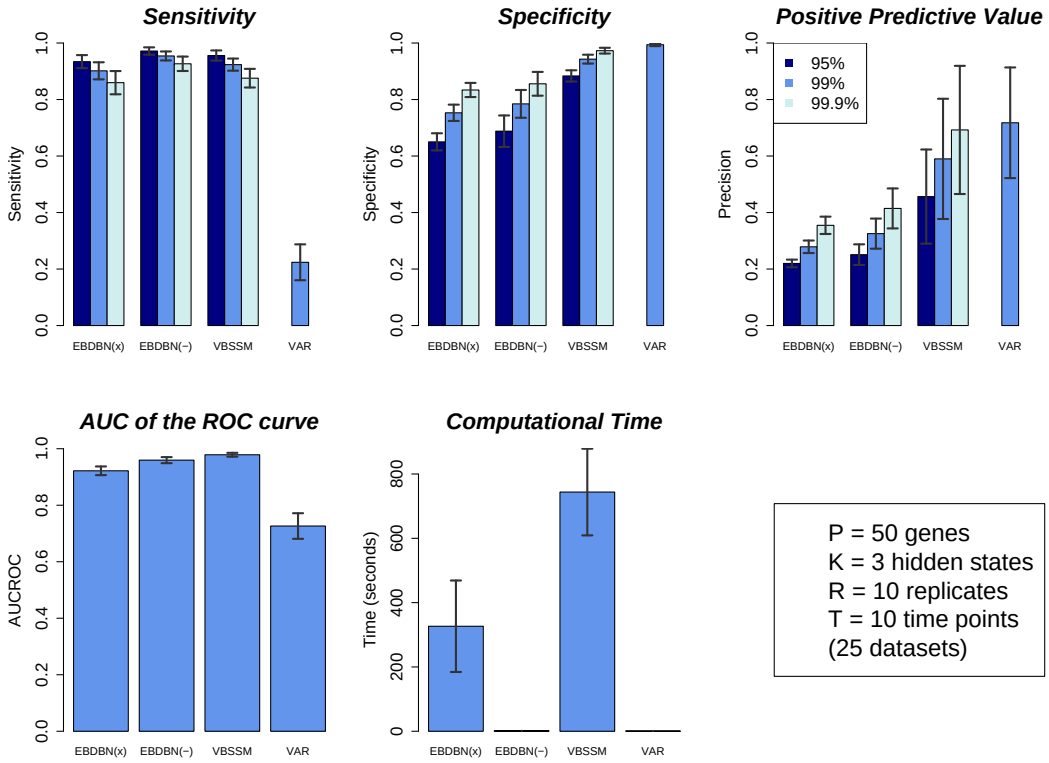


Figure 4.5. Additional comparison criteria for model-based simulations, for data with $R = 10$ replicates and $T = 10$ time points for all methods, using a cutoff for the z-scores of $\{95, 99, 99.9\}\%$ for the EBDBN(x), EBDBN(-), and VBSSM. A cutoff of 80% is used for the local false discovery rate in the VAR method, as suggested by Opgen-Rhein and Strimmer (2007).

outperforms all methods considered here, although the margin of difference decreases considerably for larger datasets. When comparing the EBDBN(x) and EBDBN(-) methods, the results indicate that the latter outperforms the former only for data with few time points ($T = 5$).

To obtain a more complete perspective on the performance of the models, we also consider the sensitivity, specificity, and positive predictive value for given threshold values for the z-score, as well as the computational time required (Figure 4.5). While the EBDBN(x), EBDBN(-), and VBSSM all seem to be comparable in terms of sensitivity and AUC, the VBSSM and VAR methods perform best in terms of specificity

and positive predictive value. For the model-based simulations, the VBSSM and VAR are more stringent in edge selection, resulting in a lower rate of *false* positives (although in the case of the VAR, this also results in a much lower rate of *true* positives). The calculation time for each method is also worth considering. The VBSSM required about 12 minutes for algorithm convergence, while for the EBDBN(x), EBDBN(-), and VAR this time was about 5 1/2 minutes, 2 seconds, and 1 second, respectively. All simulations were run on a dual-processor Dell PowerEdge 1850 (quad-core 2.8 GHz Intel (R) Xeon (TM)) with 12GB RAM, running Red Hat Enterprise Linux 5.3 Server x86-64.

4.7.3 Data-Based Simulations

One of the challenges of relying on simulations to compare different methods for network inference is that results based on data simulated under the same model as used by a particular method (as is the case for all the methods under consideration) are undoubtedly over-optimistic. For this reason, it is also important to consider "realistic" simulations of well-known systems based on ordinary differential equations, which more closely approximate real microarray datasets. One such set of simulations by Zak et al. (2001, 2003) appears to be particularly promising, as it has already been used for this purpose by several other authors (Husmeier, 2003; Beal et al., 2005).

The data in Zak et al. (2001, 2003) include 10 genes that interact with one another via a set of transcription factor proteins, bound promoters, and a ligand input, organized into regulatory motifs taken from the biological literature (Figure 4.6). A set of *in silico* simulations in Matlab, carried out by integrating the ordinary differential equations describing the model, yields a dataset of 55 variables over 500 time points. We follow Beal et al. (2005) and use only the observations for the 10 genes for network inference (thus leaving 45 variables as "hidden states"). We construct datasets of length $T = \{5, 12, 35, 50, 75, 120\}$ by subsampling equally spaced time points, and we artificially create replicates of size $R = \{1, 2, 4, 8, 16, 32\}$ by adding Gaussian noise

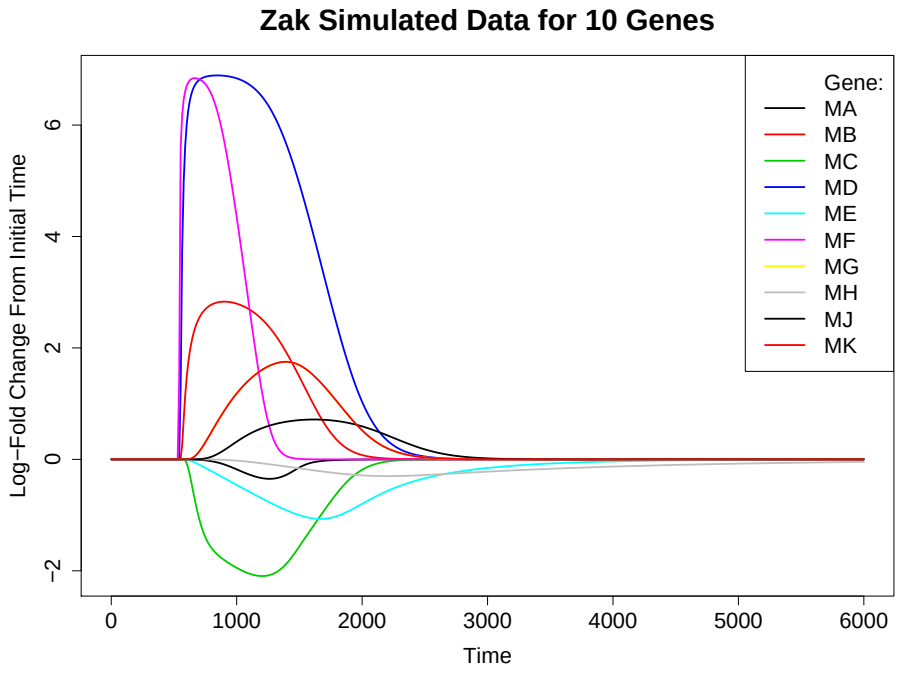


Figure 4.6. Simulated data from Zak et al. (2001, 2003) on 10 genes over 500 time points, based on the integration of a set of ordinary differential equations describing the network dynamics. Data are shown as the log-fold change of expression at each time point relative to the initial value.

at each time point to the log-ratios of the observed gene expression data. The VAR method is not included in the results below, as the small number of genes (10) in the dataset did not permit accurate edge selection.

The results of the median AUC for the EBDBN(x), EBDBN(-), and VBSSM are shown in Figure 4.7. It can be seen that data simulated with only 5 time points yield poor performance for all methods under consideration; this is alleviated by increasing the number of time points, although the improvement seems to plateau at around 50 or 75 time points. This is likely due to the artificial nature of the time point subsampling; that is, the same interval of time is covered by data sampled with 75 and 120 time points, albeit in greater detail in the latter case. The benefits of ever-

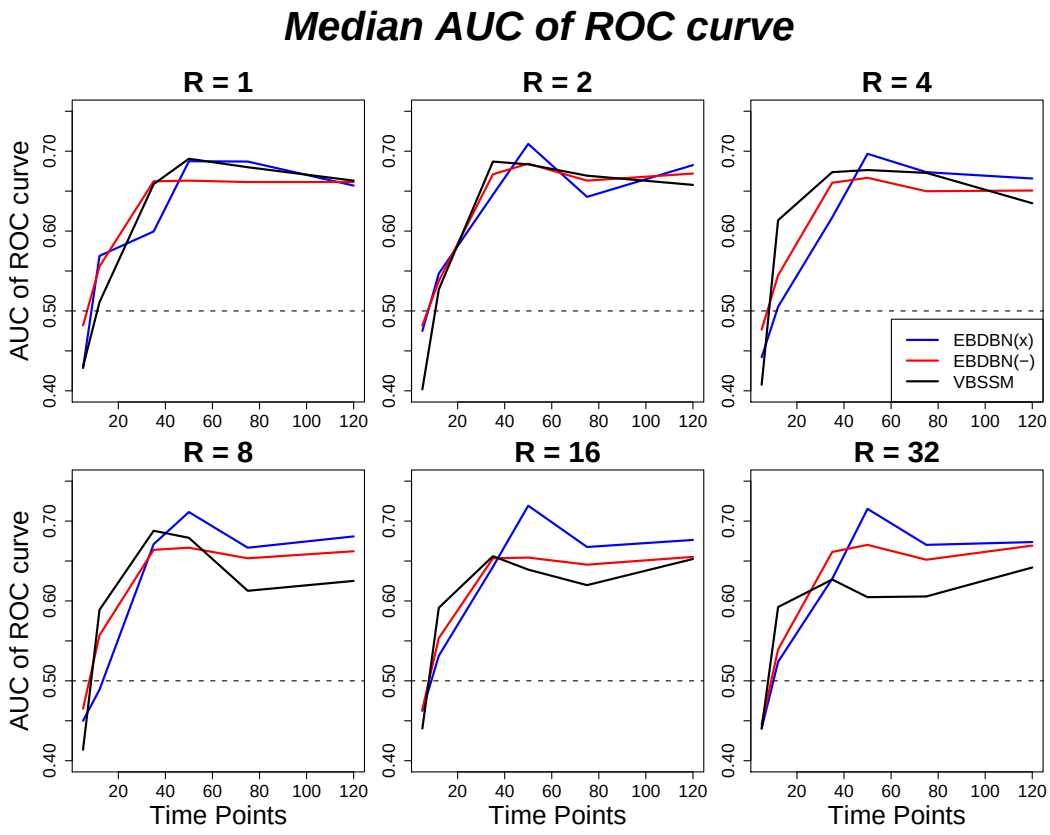


Figure 4.7. Median Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve for the EBDBN(x), EBDBN(-), and VBSSM, by number of replicates R and time points T. The horizontal dotted line in each graph represents an AUC of 0.5, corresponding to a random-guess classifier.

finer sampling are thus eventually overrun by the additional noise incurred by the extra data points.

Not surprisingly, a similar phenomenon is observed with the number of replicates, which were also artificially created by adding Gaussian noise. As with the subsampled time points, the creation of additional artificial replicates tends to add noise to the data without contributing additional information about the network dynamics. For data with 4 or more replicates, the VBSSM appears to outperform the EBDBN, but only for datasets with less than 35 time points; this relationship is reversed for a

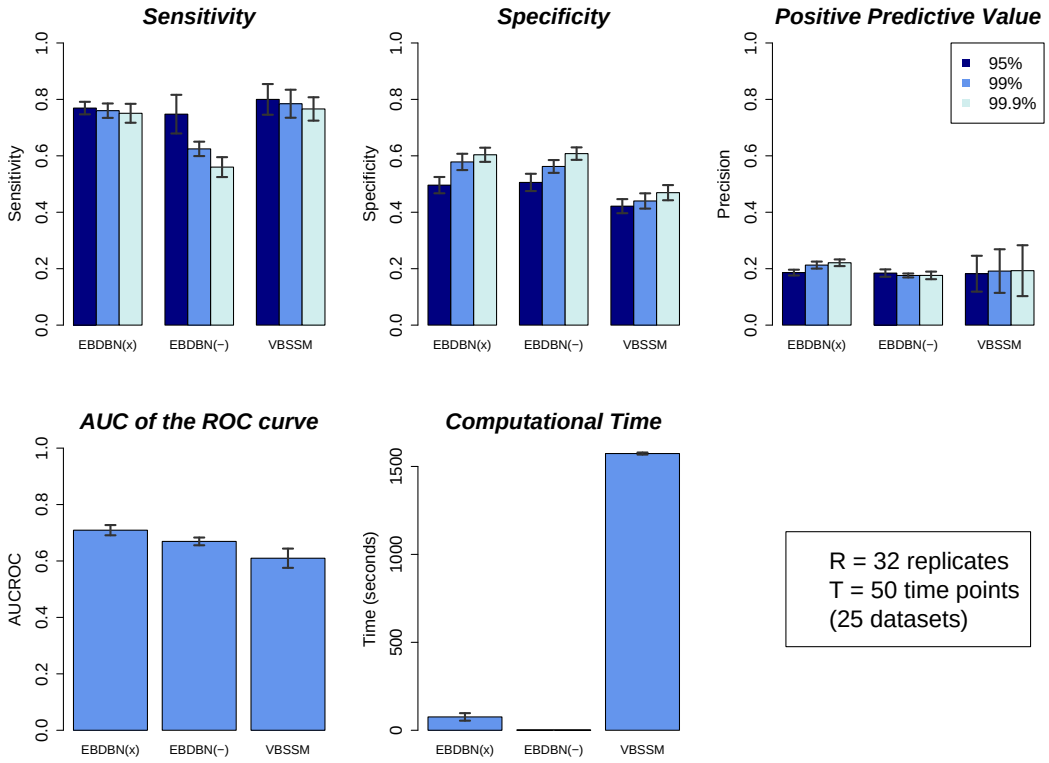


Figure 4.8. Additional comparison criteria of data-based simulations, for data with $R = 32$ replicates and $T = 50$ time points, using a cutoff for the z-scores of $\{95, 99, 99.9\}\%$ for the EBDBN(x), EBDBN(-), and VBSSM.

larger number of time points. Finally, note that the AUCROC values observed in Figure 4.7, on the order of 45% to 70%, are much lower than those observed in Figure 4.4, which were on the order of 60% to 98%. This is undoubtedly due to the more realistic, and thus more complicated, biological relationships simulated here.

We also consider the performance of the EBDBN and VBSSM on additional comparison criteria (Figure 4.8). Here, the EBDBN(x), EBDBN(-), and VBSSM perform comparably in terms of sensitivity and positive predictive value, while the EBDBN(x) and EBDBN(-) outperform the VBSSM in terms of specificity and AUC of the ROC curve. There is also a considerable difference in computational time, as the VBSSM required about 26 minutes on average to converge, while the EBDBN(x) and EBDBN(-) required only a little over 1 minute and 1 second, respectively.

4.8 Summary

The EBDBN algorithm is based on a novel empirical Bayes estimation procedure to reverse engineer the structure of gene regulatory networks from longitudinal gene expression data. This approach serves as a complement to the VBSSM of Beal et al. (2005), with the advantage of a straightforward, computationally efficient EM-like estimation procedure. The proposed method performs comparably to previously published methods on model-based data when a minimum number of replicates (≈ 10) and time points (≈ 10) are measured.

In order to incorporate hidden variables and driving inputs into the dynamics of complicated biological systems, the EBDBN relies on a hierarchical Bayes framework with conjugate Gaussian distributions to enable straightforward computation. Such distributional assumptions enable computationally efficient parameter estimation, as posterior distributions can be determined explicitly. However, the Gaussian distributions used in the EBDBN represent a simplification of the complicated dynamics inherent in biological systems. For this reason, the EBDBN is best suited for exploratory analyses of moderately-sized networks, where little prior biological information is known. However, for simple organisms that have small, well-known networks, an alternative and less restrictive approach is necessary. To address the need for diverse methods that are required in systems biology, in the next chapter we present a sampling-based Bayesian approach that is adapted to small-scale, detailed analyses of gene regulatory networks.

5. ABC-MCMC NETWORK ALGORITHM

Although the EBDBN algorithm presented in Chapter 4 has the potential to identify testable novel regulatory interactions from time-course microarray data, its estimation procedure relies on the assumption of conjugate (Gaussian) distributions for network edges. In many cases, this assumption may be unrealistic or unnecessarily restrictive. However, without such distributional assumptions on edges, it may be impossible or computationally prohibitive to compute the likelihood of a given network. To address this issue, here we present a novel adaptation of a sampling-based approximate Bayesian approach that avoids the need to use or calculate likelihoods. This method, known as the ABC-MCMC Network (ABC-Net) algorithm, aims to gain further insight into small, well-characterized biological processes. As before, let $\mathbf{y}_t = [y_{1t}, \ldots, y_{Pt}]^T$ represent the expression of a set of P genes at time t, where $t = 1, \ldots, T$. For clarity, we limit this discussion to data with a single biological replicate, but the extension to multiple replicates is straightforward.

The ABC-Net algorithm is based on the ABC-MCMC algorithm (Algorithm 3.5) of Marjoram et al. (2003), where draws from the posterior distribution are generated using a Metropolis-Hastings scheme (Hastings, 1970) without the use of likelihoods. However, adapting such an algorithm to the context of gene regulatory networks requires several considerations to be taken into account. First, we must identify computationally efficient methods for simulating data $Y^\star$ from a known gene regulatory network $\Theta^\star$. In addition, we must define a distance function ρ and threshold ϵ to compare two multivariate time series (observed and simulated data). Finally, appropriate prior and proposal distributions, $\pi(\cdot)$ and $q(\cdot|\cdot)$, must be specified.

5.1 Simulating Data for Gene Regulatory Networks

One of the most important considerations in adapting the ABC-MCMC algorithm (Algorithm 3.5) to the inference of gene regulatory networks is identifying an efficient simulator for a proposed network structure $\Theta^\star$. Several authors (Beal et al., 2005; Opgen-Rhein and Strimmer, 2007; Wilkinson, 2009) have found that simple, linear models can in some cases yield good approximations of the dynamics occurring within complicated biological systems. To this end, we use a first-order vector autoregressive model, shown in Equation (2.6), to simulate longitudinal expression data for genes involved in a given network. Specifically, after setting $\mathbf{y}_1^\star = \mathbf{y}_1$, we exploit the Markov property of the VAR(1) model to obtain one-step-ahead predictions (i.e., fitted values) of gene expression at time points $t = 2, \ldots, T$ as follows:

$$\mathbf{y}_t^\star = \Theta^\star \mathbf{y}_{t-1} \tag{5.1}$$

for $t = 2, \ldots, T$. Note that the one-step-ahead predictions for $\mathbf{y}_t$ are made using the observed data $\mathbf{y}_{t-1}$, and not the simulated data $\mathbf{y}_{t-1}^\star$. That is, we simulate data by calculating the expected value of gene expression at each time point given the network structure and observed expression values at the previous time point, rather than attempting to estimate and simulate the noise present in a particular set of observed data.

The appropriateness of using a VAR(1) simulator as in Equation (5.1) is largely dependent on the amount and the noise structure present in observed data, as well as the adequacy of the assumption of time-invariant, first-order autoregressive dynamics for complicated gene regulatory networks. In the absence of more detailed information about the underlying network, it may be reasonable to use a simple model such as the VAR(1) to generate simulated data, as shown here. However, the ABC-Net algorithm has the flexibility to incorporate arbitrary models as data simulators, provided they are computationally efficient. For instance, in some cases second-order models, nonlinear models, linear differential equations, draws from a Dirichlet process, or Michaelis-Menten kinetics may more aptly describe the dynamics of a particular gene

regulatory network; in these cases, the appropriate simulator model would be used in place of Equation (5.1). In this way, it is straightforward to tailor the ABC-Net algorithm to situations where an alternative model is known to better represent a given dataset.

5.2 Distance Function

The choice of the distance function (ρ) and cutoff threshold (ϵ) both play an important role in the applicability of ABC methods for inferring gene regulatory networks. Rather than defining a set of summary statistics, we directly compare the observed and simulated data over time using ρ. Several standard choices for this function include the Canberra, Euclidean, and Manhattan distance functions (see Table 5.1). In addition, we also apply a distance measure proposed by Lund and Li (2009) tailored to multivariate longitudinal data that we refer to as the Multivariate Time-Series (MVT) distance. For the MVT distance, we make the assumption that two sets of gene expression data arising from the same underlying network would have a smaller distance than two with differing networks. That is, under the null hypothesis that simulated ($Y^\star$) and observed (Y) data have the same network dynamics, we define

$$\hat{\Theta}_Y = \frac{1}{T}\sum_{t=1}^{T-1} \mathbf{y}_{t+1}\mathbf{y}_t'$$

$$\hat{\Theta}_{Y^\star} = \frac{1}{T}\sum_{t=1}^{T-1} \mathbf{y}^\star_{t+1}\mathbf{y}^{\star\prime}_t$$

$$\hat{\Theta} = \frac{\hat{\Theta}_Y + \hat{\Theta}_{Y^\star}}{2}$$

$$\hat{\mathbf{y}}^\star_t = \hat{\Theta}\mathbf{y}^\star_{t-1}$$

$$\hat{\mathbf{y}}_t = \hat{\Theta}\mathbf{y}_{t-1}$$

$$\hat{\Sigma} = \frac{1}{2T}\sum_{t=1}^{T}\{(\mathbf{y}^\star_t - \hat{\mathbf{y}}^\star_t)(\mathbf{y}^\star_t - \hat{\mathbf{y}}^\star_t)' + (\mathbf{y}_t - \hat{\mathbf{y}}_t)(\mathbf{y}_t - \hat{\mathbf{y}}_t)'\}$$

where $\mathbf{y}_t$ and $\mathbf{y}^\star_t$ are the observed and simulated time-course data, $\hat{\mathbf{y}}_t$ and $\hat{\mathbf{y}}^\star_t$ are the best one-step ahead linear predictors of $\mathbf{y}_t$ and $\mathbf{y}^\star_t$, respectively, and $\hat{\Sigma}$ is an estimate of the common covariance matrix of the errors Σ. With these terms defined, the MVT

Table 5.1
Distance functions to compare observed (Y) and simulated ($Y^\star$) data in the ABC-Net algorithm, where T denotes the number of time points collected and P represents the total number of genes.

Canberra:	$\rho(Y^\star, Y) = \sum_{t=1}^{T} \sum_{i=1}^{P} \frac{\lvert y_{it}^\star - y_{it} \rvert}{\lvert y_{it}^\star + y_{it} \rvert}$
Euclidean:	$\rho(Y^\star, Y) = \sqrt{\sum_{t=1}^{T} \sum_{i=1}^{P} (y_{it}^\star - y_{it})^2}$
Manhattan:	$\rho(Y^\star, Y) = \sum_{t=1}^{T} \sum_{i=1}^{P} \lvert y_{it}^\star - y_{it} \rvert$
MVT:	$\rho(Y^\star, Y) = \frac{1}{T} \sum_{t=1}^{T} [(\mathbf{y}_t - \mathbf{y}_t^\star) - (\hat{\mathbf{y}}_t - \hat{\mathbf{y}}_t^\star)]' \, \hat{\Sigma}^{-1} \, [(\mathbf{y}_t - \mathbf{y}_t^\star) - (\hat{\mathbf{y}}_t - \hat{\mathbf{y}}_t^\star)]$

distance may be calculated using the equation shown in Table 5.1. In practice, to avoid the computationally expensive task of inverting the estimated covariance matrix Σ, we standardize the two time series $Y^\star$ and Y and set $\hat{\Sigma}$ equal to the identity matrix.

Regardless of the choice of distance function ρ, a threshold value ϵ is also required to determine when simulated and observed data are "close enough." The appropriate value for ϵ is highly dependent on the choice of function ρ, the amount of noise in the data, and the size of the data (i.e., number of genes in the network, number of time points measured). To choose this threshold, we propose a heuristic method where 5000 randomly generated networks are used to simulate data as in Section 5.1, and the corresponding distance (chosen from Table 5.1) between simulated and observed data are calculated for each. We then set ϵ to the value of a given quantile (typically 1% or 5%) of these randomly generated distances. We examine the choice of both ρ and ϵ in great detail later.

5.3 Network Proposal Distributions

Another important aspect of the ABC-Net algorithm is the proposal distribution $q(\cdot|\cdot)$, which defines how to transition from the current state of a network to an updated proposal. In the context of gene regulatory networks, we rely on two different characterizations of the graph to propose updates, as previously described: the topology of the network (i.e., the edges that are present in the graph) and the magnitude of existing interactions (i.e., the strength of a particular activation or repression). The former can be represented using a matrix G, where $G_{ij} = 1$ if gene j regulates gene i, and $G_{ij} = 0$ otherwise. G is referred to as the adjacency matrix of the underlying gene regulatory network. The latter is represented by a matrix Θ of parameter values (see Equation (2.6)). Note that $G_{ij} = 0$ implies $\Theta_{ij} = 0$, and $G_{ij} = 1$ implies $\Theta_{ij} \neq 0$.

In the ABC-Net algorithm, a two-step proposal distribution (Figure 5.2) is used to produce new samples $G^\star$ and $\Theta^\star$ for the adjacency and parameter matrices, respectively. In the first step, we apply one of three basic moves (Figure 5.1) to the current adjacency matrix G^i: adding an edge, deleting an edge, or reversing the direction of an edge (Husmeier et al., 2005). Let $\mathcal{N}(G)$ represent the neighborhood size of a graph G, i.e., the number of other network structures that can be obtained by applying one of the three basic moves. The transition probability is given by $q(G^\star|G^i) = 1/\mathcal{N}(G^i)$. In the second step, the proposal distribution of Θ, given the current value Θ^i and the updated adjacency matrix $G^\star$, is defined to be

$$q(\Theta_{ij}|\Theta^i_{ij}, G^\star_{ij}) \sim \begin{cases} 0 & \text{if } G^\star_{ij} = 0 \\ N(\Theta^i_{ij}, \sigma_\Theta) & \text{if } G^\star_{ij} \neq 0 \end{cases} \tag{5.2}$$

where σ_Θ may be tuned to obtain an empirical acceptance rate between 15% and 50%, as recommended in Gilks et al. (1996). A simple example of the two-step proposal distribution for gene regulatory networks is shown in Figure 5.2.

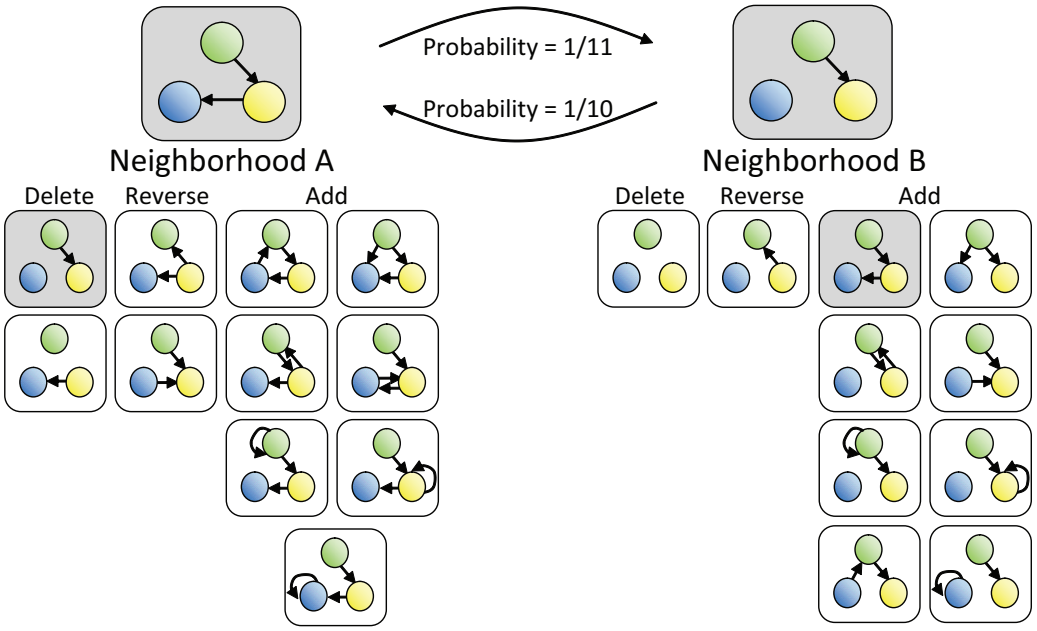


Figure 5.1. Proposal distribution for network adjacency matrix G through application of three basic moves: adding, deleting, and reversing an edge. Neighborhood A is made up of 11 graphs. The proposal probability for moving from this graph to that shown on the top right is thus 1/11. As Neighborhood B is made up of 10 graphs, the proposal probability of moving back to the first graph is 1/10. Similar image shown in Husmeier et al. (2005).

5.4 Network Prior Distributions

One final consideration for the ABC-Net algorithm is the choice of prior distributions for the adjacency matrix G and parameter matrix Θ. In gene regulatory networks, as the number of genes (P) in a network increases, the number of possible edges within the network quickly increases as well ($P \times P$). A large number of genes may be interacting simultaneously with one another in very sophisticated regulatory circuits, and the network topology itself may be quite complicated. Even so, certain properties of biological networks can be useful in limiting the support of the prior distribution to realistic network topologies. In particular, recall that most genes are

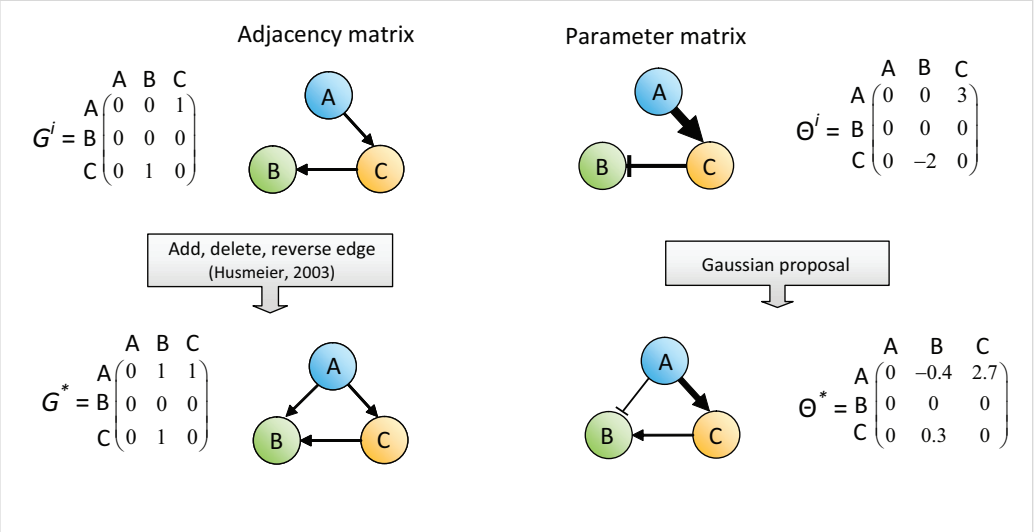


Figure 5.2. Example of two-step proposal distribution for gene regulatory networks. Top row: A network in iteration i of the ABC-Net algorithm may be characterized both by its adjacency matrix G^i (left) and its parameter matrix Θ^i (right). The former encodes only the presence (1) or absence (0) of an edge. The latter encodes additional information about the magnitude of a particular interaction, where zeros indicate that an edge is not present, positive values indicate an activation, and negative values indicate a repression (edges with values further away from zero correspond to stronger effects). Bottom row: An updated network is proposed by adding, deleting, or reversing an edge in G^i to produce $G^\star$ (left). The parameter matrix Θ^i is updated using a Gaussian proposal distribution for the nonzero edges of $G^\star$ to produce $\Theta^\star$ (right).

regulated just one step away from their regulator (Alon, 2007), and gene networks tend to be sparse with a limited number of regulator genes (Leclerc, 2008).

In keeping with these biological hypotheses, we elect to use uninformative prior distributions with some restrictions for both $\pi(G)$ and $\pi(\Theta|G)$. One such restriction is on a graph attribute known as the fan-in for each node in the network, which refers to the in-degree of each node. Because gene regulatory networks are known to be sparse, we choose the prior on the adjacency matrix, $\pi(G)$, to be uniform over all possible structures, subject to a constraint on the maximum fan-in for each node in the network (i.e., the number of regulators per gene), as has been suggested

Algorithm 5.1 The ABC-MCMC Network (ABC-Net) Algorithm.

0. Initialize Θ^i, G^i, $i = 0$.
1. (a) Propose $G^\star$ according to a proposal distribution $q(G|G^i)$ by adding, deleting, or reversing an edge as in Figure 5.1.
   (b) Propose $\Theta^\star$ according to a proposal distribution $q(\Theta|\Theta^i, G^\star)$.
2. Simulate data $Y^\star$ from $f(\cdot|\Theta^\star, G^\star)$.
3. If $\rho(Y^\star, Y) \leq \epsilon$, go to 4, otherwise set $\Theta^{i+1} = \Theta^i$, $G^{i+1} = G^i$ and go to 5.
4. Set $\{G^{i+1}, \Theta^{i+1}\} = \{G^\star, \Theta^\star\}$ with probability $\alpha = \min\{1, \frac{\pi(G^\star)\pi(\Theta^\star|G^\star)q(G^i|G^\star)q(\Theta^i|\Theta^\star,G^i)}{\pi(G^i)\pi(\Theta^i|G^i)q(G^\star|G^i)q(\Theta^\star|\Theta^i,G^\star)}\}$ and $\{G^{i+1}, \Theta^{i+1}\} = \{G^i, \Theta^i\}$ with probability $1 - \alpha$.
5. Set $i = i + 1$. If $i < n$ (a pre-set number of iterations), return to 1.

by several authors (Friedman, 2000; Husmeier, 2003; Werhli and Husmeier, 2007). This restriction is supported by the biological literature, as genes do not tend to be synchronously regulated by a large number of genes (Leclerc, 2008). For the parameter prior $\pi(\Theta|G)$, we use a uniform distribution on the interval $(-2, 2)$, where the bounds are chosen to represent a realistic range of edge magnitudes in gene regulatory networks. In addition, both priors can easily be adapted to incorporate prior biological information (i.e., in the case of $\pi(G)$ the presence/absence of an edge, and in the case of $\pi(\Theta|G)$ the type and magnitude of a particular gene-to-gene interaction).

5.5 ABC-Net Implementation

The proposed ABC-Net algorithm adapts the ABC-MCMC algorithm (Algorithm 3.5) to the context of gene regulatory networks by incorporating a specialized data simulator, distance function, network proposal distribution, and prior distribution, as detailed in Sections 5.1-5.4. The full ABC-Net algorithm is shown in Algorithm

5.1. The output from the ABC-Net algorithm consists of dependent samples from the stationary distribution of the chain, $f(\Theta, G|\rho(Y^\star, Y \leq \epsilon)$. Because saving all iterations from the MCMC run can take up a large amount of storage (particularly as the size of the network increases) and consecutive draws tend to be highly correlated, we typically save only every 50^{th} iteration, which is referred to as thinning the chain (Gilks et al., 1996). In practice, two additional considerations must be taken into account when implementing the ABC-Net algorithm: a burn-in period and a measure of chain convergence.

5.5.1 Burn-In Period

As with many MCMC methods, a burn-in period is implemented to reduce the impact of initial values and to improve mixing for the chain. After a sufficiently long burn-in period of b iterations, inference is based on the output from the Markov chain $\{(G^i, \Theta^i); i = b+1, \ldots, b+n\}$, which consists of dependent samples from the approximate posterior distribution $\pi(\Theta|G, \rho(Y^\star, Y) \leq \epsilon)$ (Gilks et al., 1996). The length b of the burn-in depends on the starting values of the chain, Θ^0 and G^0, the rate of convergence of the chain, and the similarity of the transition mechanism of the chain to the approximate posterior distribution. Geyer (1992) suggests that b should be set to between 1% and 2% of the run length n, if extreme starting values are avoided. In addition, several formal tools to determine b, known as convergence diagnostics, have also been proposed; see Cowles and Carlin (1994) for a review.

We also implement a "cooling" procedure during the burn-in period similar to that used in Ratmann et al. (2007), where acceptance of $(G^\star, \Theta^\star)$ is controlled by a decreasing sequence of thresholds, until the minimum pre-set value ϵ is reached. Note that tempering the acceptance threshold ϵ in this way reduces the number of accepted parameters as the number of iterations increases. This cooling scheme also addresses the poor mixing often observed in the ABC-Net algorithm, as larger tolerances in the early iterations of the burn-in are associated with higher acceptance rates. A

total of 200 iterations are run for each of ten cooled threshold values, and the burn-in period is repeated if the empirical acceptance rate is less than 1%. This ensures a minimum burn-in period of 2000 iterations, with additional iterations included for chains affected by poor mixing.

5.5.2 Chain Convergence

Because the ABC-Net algorithm relies on a comparison between simulated and observed data to avoid a likelihood calculation, long chains are required to ensure the adequacy of the approximation. Although a single long chain could be run, it is also possible to run multiple overdispersed chains. There is still much debate in the statistical community concerning whether it is better to run a single, very long chain or several long chains. The former approach maintains that a single chain has the best chance of more fully exploring the target distribution and that comparisons among chains cannot absolutely prove convergence. The latter approach holds that comparing chains can reveal genuine differences if the individual chains have not yet approached stationarity (Gilks et al., 1996). For this reason, we run 10 independent chains of length 1×10^6 simultaneously (rather than a single chain of length 1×10^7). This approach contributes a two-fold benefit, as calculations can be performed in parallel to improve computational speed and a convergence assessment can be conducted using the Gelman-Rubin statistic (Gelman and Rubin, 1992).

Specifically, to assess convergence among chains, consider the case where m parallel sequences of length n (after the burn-in period) have been run. For a single parameter η, let (η_{ij}) be the sample of η at the j^{th} iteration of the i^{th} chain for $j = 1, \ldots, n$ and $i = 1, \ldots, m$, and calculate the between-sequence variance B and the within-sequence variance W as follows:

$$B = \frac{n}{m-1} \sum_{i=1}^{m} (\bar{\eta}_{i\cdot} - \bar{\eta}_{\cdot\cdot})^2$$

$$W = \frac{1}{m} \sum_{i=1}^{m} s_i^2 \tag{5.3}$$

where

$$\bar{\eta}_{i\cdot} = \frac{1}{n}\sum_{j=1}^{n}\eta_{ij}, \qquad \bar{\eta}_{\cdot\cdot} = \frac{1}{m}\sum_{i=1}^{m}\bar{\eta}_{i\cdot}, \qquad \text{and} \qquad s_i^2 = \frac{1}{n-1}\sum_{j=1}^{n}(\eta_{ij} - \bar{\eta}_{i\cdot})^2.$$

The within-sequence variance W in Equation (5.3) is an underestimate of the variance of η since individual sequences do not cover the entirety of the target distribution. We also define an overestimate of the variance of η in the target distribution, based on an assumption of overdispersed starting points:

$$\hat{\text{Var}}(\eta) = \frac{n-1}{n}W + \frac{1}{n}B. \tag{5.4}$$

The Gelman-Rubin statistic (Gelman and Rubin, 1992) monitors chain convergence by estimating the ratio of $\hat{\text{Var}}(\eta)$ and W (the upper and lower bounds, respectively) of the standard deviation in $\hat{\eta}$:

$$\sqrt{\hat{R}} = \sqrt{\frac{\hat{\text{Var}}(\eta)}{W}}. \tag{5.5}$$

As the chains converge, the Gelman-Rubin statistic $\hat{R}$ decreases towards 1 (which indicates that the parallel Markov chains essentially overlap). Following the recommendation in Gilks et al. (1996) we declare chain convergence if $\hat{R} < 1.2$ for all parameters in Θ. After the chains have converged, draws corresponding to the smallest 1% of the distance criterion are retained for inference.

5.6 Extension to RNA Sequencing Data

As the cost of NGS technology continues to drop, sequence-based longitudinal studies of gene expression (e.g., RNA-Seq) will become increasingly popular. However, data from NGS platforms differ substantially from microarray data, particularly as they are made up of digital counts of transcripts rather than continuous intensity measures. Because the VAR(1) model described in Section 5.1 generates continuous expression measurements over time, an alternative simulator must be proposed to deal with time series of integer counts.

Specifically, let $\mathbf{y}_t = [y_{1t}, \ldots, y_{Pt}]$ be the digital gene expression (DGE), or count, of genes at time t, where $t = 1, \ldots, T$. It has become standard practice (Mortazavi et al., 2008; Marioni et al., 2008; Auer and Doerge, 2010) to model the level of gene transcription with a Poisson distribution:

$$y_{it} \sim \text{Poisson}(\lambda_{it}) \tag{5.6}$$

where $\lambda_{it} = \pi_{it} \sum_{i=1}^{P} y_{it}$ and $\sum_{i=1}^{P} \pi_{it} = 1$. In this way, the inclusion of the total number of reads per time point, $y_{\cdot t} = \sum_{i=1}^{P} y_{it}$, amounts to normalizing digital gene counts by the total number of reads per time point. To define gene-to-gene interactions across time, we incorporate the parameter matrix Θ (which defines the network structure, as before) into an autoregressive model for the gene transcription rate π_{it} of the i^{th} gene as follows:

$$\pi_{it} = \frac{\lambda_{it}}{y_{\cdot t}} = \exp\left\{\theta_{i1}\frac{\lambda_{1,t-1}}{y_{\cdot,t-1}} + \ldots + \theta_{iP}\frac{\lambda_{P,t-1}}{y_{\cdot,t-1}}\right\} \tag{5.7}$$

where θ_{ij} represents the element of Θ in the i^{th} row and j^{th} column, as before. In matrix form, this would be represented as

$$\boldsymbol{\pi}_t = \exp\left\{\frac{1}{y_{\cdot,t-1}}\Theta\boldsymbol{\lambda}_{t-1}\right\} \tag{5.8}$$

where $\boldsymbol{\pi}_t = \{\pi_{it}\}_{i=1,\ldots,P}$ and $\boldsymbol{\lambda}_{t-1} = \{\lambda_{i,t-1}\}_{i=1,\ldots,P}$ for $t = 1, \ldots, T$.

As in Section 5.1, to generate simulated data $Y^\star$ based on a given network $\Theta^\star$, we rely on one-step-ahead predictions. The difference in this case is that one-step-ahead predictions are on the level of expression π_{it} of each gene, and not on the observed values y_{it} themselves. Specifically, let $\mathbf{y}_1^\star = \mathbf{y}_1$ and

$$\begin{aligned}
\tilde{\boldsymbol{\pi}}_t^\star &= \exp\left\{\frac{1}{y_{\cdot,t-1}}\Theta^\star\mathbf{y}_{t-1}\right\} \\
\boldsymbol{\pi}_t^\star &= \frac{1}{\sum_{i=1}^{P}\tilde{\pi}_{it}^\star}\tilde{\boldsymbol{\pi}}_t^\star \\
\mathbf{y}_t^\star &\sim \text{Poisson}(\boldsymbol{\pi}_t^\star y_{\cdot t})
\end{aligned} \tag{5.9}$$

for $t = 2, \ldots, T$. In this way $\boldsymbol{\pi}^\star$ represents the level (or rate) of expression of gene i at time t, which in turn is influenced by the observed digital gene expression measures at the previous time point, $\mathbf{y}_{t-1}$, as determined by the matrix of gene-to-gene

interactions, $\Theta^\star$. In contrast to the simulation procedure described in Section 5.1, the Poisson simulator in Equation 5.9 incorporates an additional level of randomness since DGE measures are sampled from a Poisson distribution. To account for this, we suggest repeating the sampling procedure several (say, 10) times such that

$$\mathbf{y}_t^{(j)\star} \sim \text{Poisson}(\boldsymbol{\pi}_t^\star y_{\cdot t})$$

where $\mathbf{y}_1^{(j)\star} = \mathbf{y}_1$ and $j = 1, \ldots, 10$. Subsequently, the distance $\rho(Y^\star, Y)$ for a given network $\Theta^\star$ would be set to equal to the mean distance across all simulated data $Y^{(j)\star}$, that is

$$\rho(Y^\star, Y) = \text{mean}\left\{\rho(Y^{(j)\star}, Y) : j = 1, \ldots, 10\right\}.$$

The proposed extension of the ABC-Net algorithm to RNA-Seq data has the added benefit of a more realistic interpretation for the network parameters Θ as compared to the VAR(1) model in Section 5.1 or the EBDBN algorithm of Chapter 4. In the VAR(1) representation based on continuous expression measurements (i.e., from microarray data), Θ encodes how gene expression at one time point directly affects gene expression at the next. In the Poisson model representation based on digital gene expression measurements (i.e., from SAGE or RNA-Seq data), the elements of Θ encode how digital gene expression measures from one time point affect the level (or rate) of gene expression at the next. That is, in the continuous representation, if $\theta_{ij} = 2$, then the expression of gene i at time point t is strictly increased by twice the value of expression of gene j at time point $t-1$. However, in the Poisson representation, $\theta_{ij} = 2$ implies that the *rate* of expression of gene i at time t is increased by twice the digital gene expression of gene j at time point $t-1$. Although the difference is subtle, it is important to distinguish between these two interpretations.

One final consideration for the extension of the ABC-Net algorithm to longitudinal RNA-Seq data is the possibility of over-dispersion. The Poisson model in Equation 5.6 relies on the assumption that Poisson sampling accounts for the variation that occurs among observations in the same treatment group. That is, if R biological

replicates are included in the data, we assume $E(y_{itr}) = \text{Var}(y_{itr})$ for $r = 1, \ldots, R$. However, RNA-Seq data often exhibit extra-Poisson variation (also referred to as overdispersion) between biological replicates, and in such cases the approach must be modified appropriately (Auer, 2010). For example, Auer and Doerge (2010) proposed a random effects formulation with quasi-likelihood estimation to account for extra-Poisson variability. However, in our current formulation no estimate of overdispersion may be estimated, as only one biological replicate is included.

5.7 Simulations

The success of the ABC-Net method in inferring gene regulatory networks from expression data depends on several critical factors. In particular, some important considerations include the choice of distance function ρ and tolerance ϵ, the sensitivity of the method to the bounds of the prior distribution, and the suitability of the model used to generate simulated data (i.e., the VAR(1) simulator) when more complicated dynamics are at play. In addition, we also examine the effect of increasing the amount of noise present in the observed data and incorporating biological knowledge into the prior distribution structure. In this context, we focus on the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve as an indicator of performance, as well as qualitative examinations of the approximate posterior distributions of edges in the network.

To calculate the AUC for the ABC-Net method (Algorithm 5.1), we retain only the samples corresponding to the smallest 1% of distances $\rho(Y^\star, Y)$ for inference. Based on these samples, we calculate the bounds of the α% credible intervals for each edge, where $\alpha = \{1, \ldots, 100\}$. If the α% credible interval for a particular edge does not contain 0, the edge is declared to be present; otherwise, the edge is declared to be absent. In this way, because the simulation setting determines which edges are truly present and absent, true positives, false positives, true negatives, and false negatives may be calculated for each α, and the AUC may subsequently be calculated. We also

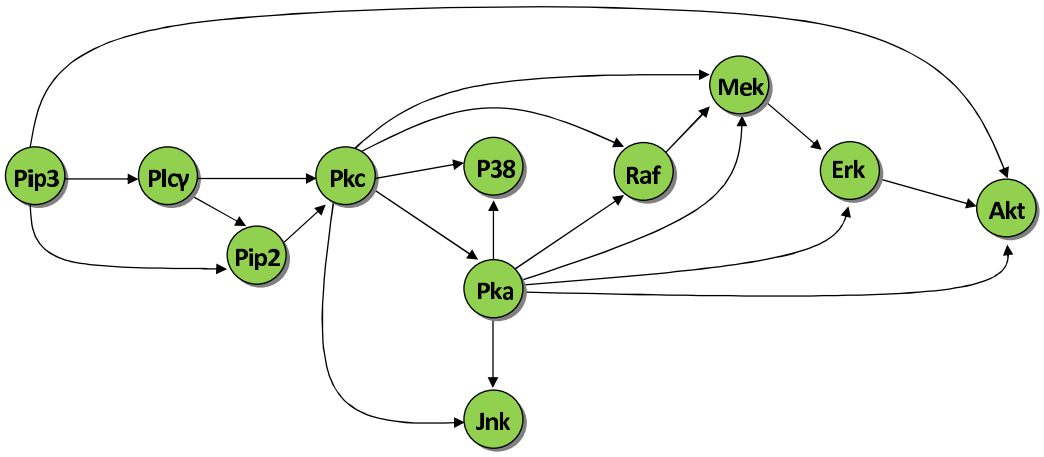


Figure 5.3. The currently accepted gold-standard Raf signalling pathway, which describes the interactions of eleven phosphorylated proteins in primary human immune system cells (Sachs et al., 2005). Nodes represent the proxy genes of each of the eleven proteins (i.e., the genes that are transcribed and translated into the corresponding proteins), and arrows indicate the direction of signal transduction. Similar figure given in Werhli and Husmeier (2007).

include as a reference the average acceptance rates and computation time required for each simulation.

For these simulations, rather than defining an arbitrary network Θ, we instead use the structure of a real, well-characterized pathway in human immune system cells involving the Raf signalling protein (Sachs et al., 2005). We generate "observed" data based on 11 genes, where the adjacency matrix Θ^{Raf} is defined using the structure of the currently accepted Raf signalling network, shown in Figure 5.3. If an edge exists from gene j to gene i, we sample Θ_{ij}^{Raf} uniformly from the interval $(-2, -0.25) \cup (0.25, 2)$, and otherwise $\Theta_{ij}^{\text{Raf}} = 0$. The bounds for nonzero gene-to-gene interactions were chosen to represent a range of moderate to strong interactions among genes. We generate one replicate of expression data for each of the 11 genes over 20 time points, using the first-order autoregressive model

$$\mathbf{y}_t = \Theta^{\text{Raf}} \mathbf{y}_{t-1} + \mathbf{z}_t \tag{5.10}$$

for $t = 2, ..., T$, where $\mathbf{y}_1 \sim N(0, I)$, $\mathbf{z}_t \sim N(0, \sigma)$. For each simulation, unless otherwise noted, the noise standard deviation is set to $\sigma = 1$, the Gaussian proposal standard deviation in Equation (5.2) is set to $\sigma_\Theta = 0.5$, and the maximum fan-in is constrained to 5 or less (i.e., each gene has a maximum of five regulators). In all cases, we ran the ABC-Net algorithm for ten independent chains of length 1×10^6, with a thinning interval of 50.

5.7.1 Choice of ρ and ϵ

The distance function ρ and threshold ϵ (Section 5.2) are essential components to the ABC-Net algorithm, as they directly affect the probability that simulated data $Y^\star$ generated by a network $\Theta^\star$ are accepted as being "close enough" to the observed data. Although there are many potential options for this distance function, in this work we focus on a comparison among the four options previously discussed: the Manhattan, Euclidean, Canberra, and MVT distances (see Table 5.1). For each choice of ρ, we set a threshold using the heuristic method described in Section 5.2, where ϵ is set to be either the 1%, 5%, or 10% quantile of distances associated with 5000 randomly generated networks. Data were generated using the structure of the Raf signalling network as in Equation (5.10), and each combination of ρ and ϵ was repeated over five independent datasets in order to include an assessment of their variability (Note: only two datasets were simulated for the MVT distance due to its computational burden). For all of these simulations, it is assumed that smaller distances correspond to data generated from networks $\Theta^\star$ similar to the real network Θ^{Raf}. In this section, all simulations were associated with Gelman-Rubin statistics less than 1.2 for all parameters, so it is assumed that samples from the ABC-Net algorithm were indeed from the stationary distribution of the chain (see Theorem 3.2.1).

Each distance function under consideration calculates and penalizes differences between simulated and observed data in a different way. As such, it is not surprising that the scales and distributions of distances for each choice of ρ differ as well (Figure

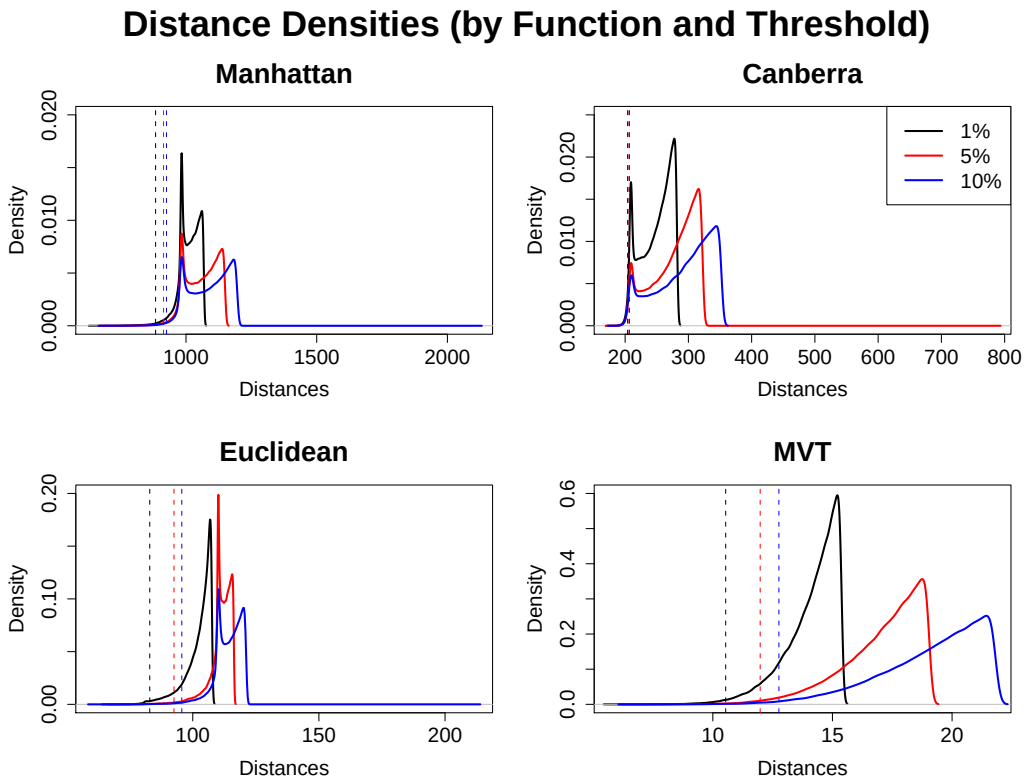


Figure 5.4. Densities of distances $\rho(Y^\star, Y)$ for four choices of distance functions in the ABC-Net algorithm: Manhattan, Canberra, Euclidean, and MVT distances (see Table 5.1). Black, red, and blue solid lines indicate the densities associated with thresholds ϵ chosen using the 1%, 5%, and 10% quantiles from 5000 randomly generated networks (see Section 5.2). Black, red, and blue dashed lines indicate the cutoff to retain samples inference for each of the three choices of ϵ (1%, 5%, and 10% quantiles, respectively).

5.4). The Manhattan, Canberra, and Euclidean functions all result in bi-modal densities for the distances, while the MVT function appears to be strictly increasing as the distance approaches the cutoff ϵ. In addition, because the Manhattan distance deals directly with the absolute differences between simulated and observed data, it is on a larger scale than the other distance functions. Figure 5.4 also includes dotted lines corresponding to the cutoff for samples retained for final inference (i.e., the smallest 1% of distances calculated for each function). For the Canberra distance, these lines

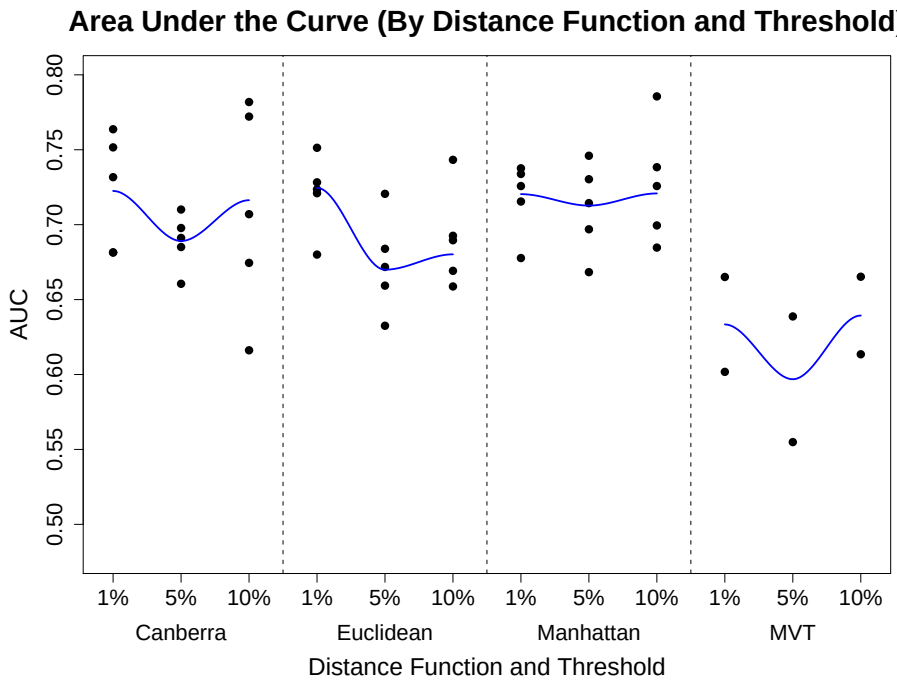


Figure 5.5. Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve for four choices of distance functions in the ABC-Net algorithm: Manhattan, Canberra, Euclidean, and MVT distances (see Table 5.1). Black dots represent the value of the AUC for each of five independent datasets per threshold and distance function (with the exception of the MVT distance, which was limited to two datasets due to its computational burden). The threshold ϵ was set at the 1%, 5%, and 10% quantiles from 5000 randomly generated networks (see Section 5.2). Blue lines represent loess curves (Cleveland, 1979).

essentially overlap, indicating that the distances associated with samples retained for inference are the same whether the ABC-Net algorithm was run based on the 1%, 5%, or 10% quantiles for ϵ. For the other three distance functions, it appears that these cutoffs are more spread out.

As it is difficult to determine the ramifications of each choice of ρ from Figure 5.4, we also consider a plot of the AUC for each combination of ρ and ϵ (Figure 5.5). This plot clearly shows a disconnect between the performance of the Canberra, Euclidean, and Manhattan distances as compared to the MVT distance. The first three distance

functions all appear to be on par with one another, particularly when ϵ is set at the 1% quantile of distances. For larger threshold values ($\epsilon = 5\%$ and 10%), the Euclidean distance seems to suffer slightly when compared to the Canberra and Manhattan distances, which display relatively constant performance regardless of the choice of ϵ. The MVT distance clearly yields much lower values of AUC regardless of the choice of ϵ. However, based on this criterion alone, there does not seem to be strong evidence that favors one choice among the Canberra, Euclidean, and Manhattan distances, particularly for a cutoff of $\epsilon = 1\%$.

We also conduct a qualitative examination of the approximate posterior distributions for the edges in the Raf signalling pathway (Figure 5.6) in one simulated dataset. The marginal approximate posterior distributions for each distance function are superimposed over each edge in the diagram, with different colors corresponding to different distance functions, as indicated in the figure legend. True values taken on by each edge are indicated with dotted black lines in each plot. Note that each plot has the same lower and upper bounds for the x-axis (-2 and 2, respectively) and y-axis (0 and 1.2, respectively) to facilitate comparisons among plots.

We first note (Figure 5.6) that some edges have very similar posterior distributions regardless of the distance function used (e.g., Pip3→Akt and Pka→P38), while others seem to have very different densities for different choices of ρ (e.g., Pkc→Jnk and Pka→Erk). It also seems that the Manhattan and Euclidean distance often yield approximate posterior distributions that are very close to one another, while those of the Canberra and MVT distances often deviate. Another notable feature of Figure 5.6 is that some edges have relatively flat (diffuse) approximate posterior distributions (e.g., Pip3→Akt), while others are much more peaked (e.g., Erk→Akt). We refer to edges with these two characteristics as "flexible" and "rigid" edges, respectively, and we will return to this idea in greater detail in later simulations.

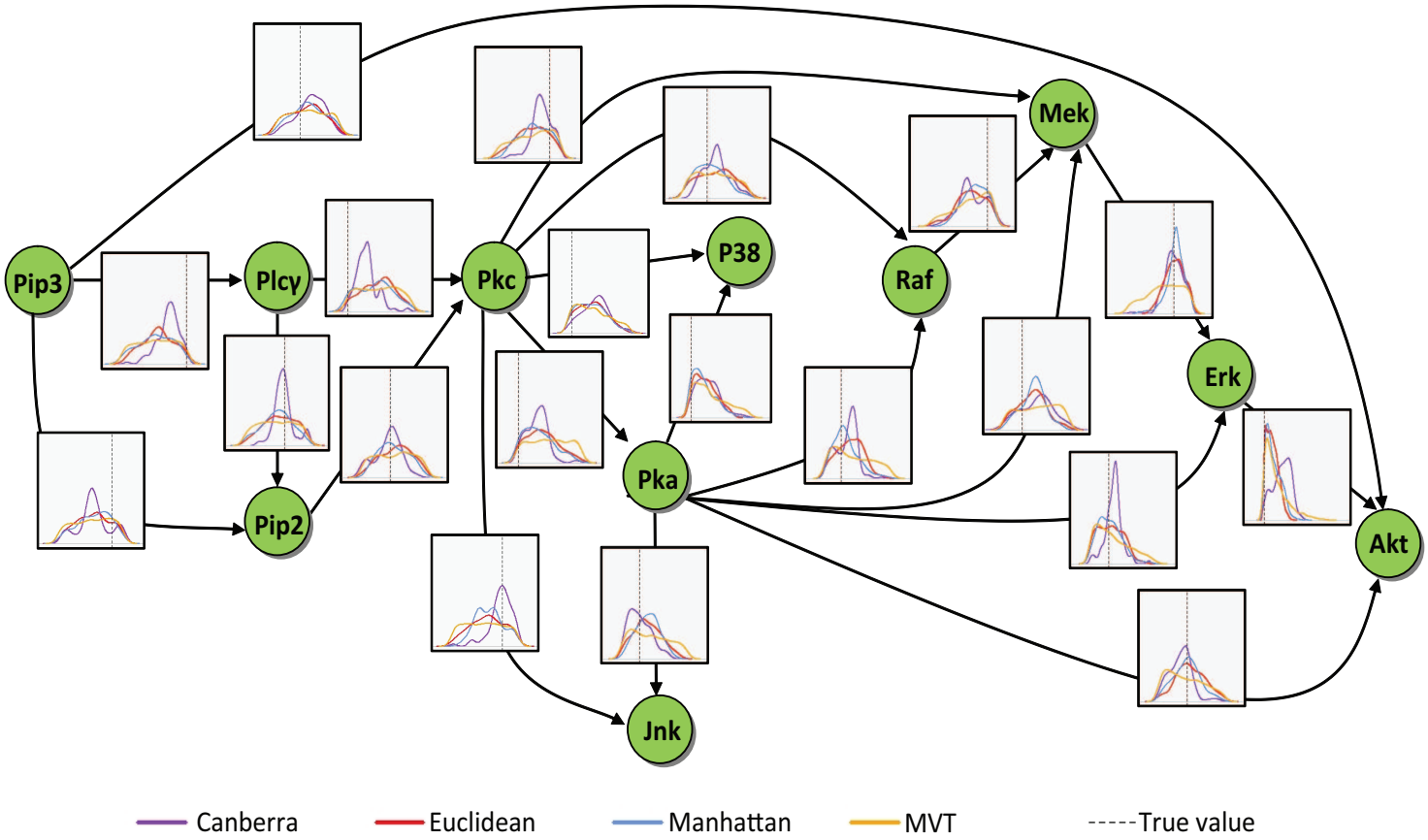


Figure 5.6. Approximate posterior distributions from the ABC-Net method for the simulated Raf signalling pathway, by distance function ρ. Each plot overlapping a given edge in the network corresponds to the marginal approximate posterior distribution for that edge, where the x-axis represents the values taken on by the edge over the interval (-2,2) and the y-axis is the density. Each color represents a different distance function, as shown in the legend. Black dotted lines indicate the true value taken on by the edge in the true network Θ^{Raf}.

Table 5.2
Average time in hours and acceptance rates (5 datasets each, except for MVT which has two) for different distance functions ρ and thresholds ϵ. Numbers in parentheses indicate standard deviations. Simulations were run on a dual-socket Dell PowerEdge 1950 (quad-core (8 effective CPUs) Intel Xeon E5410) with 32GB RAM, running RedHat Enterprise Linux 5.4 Server x86-64.

Distance (ρ)	Threshold (ϵ)	Time (hours)	Acceptance rates
Canberra	1	8.1 (0.6)	45.4 (2.7)
	5	9.6 (0.4)	41.2 (2.2)
	10	9.9 (0.6)	44.6 (1.5)
Euclidean	1	7.3 (0.4)	38.0 (7.1)
	5	7.8 (0.4)	56.5 (3.4)
	10	8.3 (0.6)	59.0 (1.9)
Manhattan	1	7.4 (0.5)	59.5 (3.4)
	5	8.0 (0.5)	66.7 (2.1)
	10	8.4 (0.5)	69.0 (7.1)
MVT	1	21.9 (1.4)	29.2 (2.5)
	5	23.5 (0.4)	44.3 (4.4)
	10	25.0 (1.3)	53.1 (3.3)

Based on these simulation results, we conclude that the MVT distance is a poor choice of distance function within the ABC-Net algorithm. Regardless of the threshold used, it displays considerably different behavior than the other functions (Figure 5.5) and largely worse performance in terms of AUC (Figure 5.5). In addition, the computational burden for the MVT distance is much larger than the other methods (see Table 5.2), requiring almost three times the number of hours for calculation as compared to the other methods. However, choosing among the Canberra, Euclidean, and Manhattan distance is not as straightforward. All have approximately the same computational burden (Table 5.2), and seem to display similar performance in terms

of AUC. Based on a qualitative examination of the approximate posterior distributions (Figure 5.6), the results of the Canberra distance often break away from those of the Manhattan and Euclidean distances. However, no strong evidence exists in favor of the Canberra distance. Between the remaining two distance functions, the Euclidean distance enjoys a slight advantage over the Manhattan distance in terms of computation time. For this reason, we elect to use the Euclidean distance with ϵ set to the 1% threshold for the remainder of the simulations.

5.7.2 Sensitivity to prior distribution bounds

In the work presented here, we focus on uniform prior distributions for both $\pi(G)$ and $\pi(\Theta|G)$ (Section 5.4), subject to a constraint on the maximum fan-in (i.e., number of regulators per gene) for the former. For the latter, we must decide on reasonable bounds for the prior distribution, which affect, in turn, the bounds of the approximate posterior distributions obtained from the ABC-Net method. When prior biological information on specific gene-to-gene interactions is to be incorporated into the ABC-Net method, it may be straightforward to fix the bounds of this prior distribution (e.g., known activations would be limited to positive values and known repressions would be limited to negative values). However, in most cases fixing an upper and lower bound for $\pi(\Theta|G)$ requires some careful thought.

In the case of continuous time-course data using a VAR(1) simulator as shown in Section 5.1, Θ encodes how gene expression at one time point directly affects gene expression at the next. That is, if $\theta_{ij} = 2$, then the expression of gene i at time point t is strictly increased by twice the value of expression of gene j at time point $t-1$. In addition, this relationship is assumed to be constant across time (i.e., gene j regulates gene i in the same manner from times $t = 1$ to $t = 2$, as well from times $t = 9$ to $t = 10$). Because values of -2 and 2 for a given θ_{ij} correspond to strong inhibitory and activatory effects, respectively, they represent one set of reasonable bounds for the uniform prior distribution on Θ. As a comparison, we also consider (-3,3), (-5,5), and

Table 5.3
Average time and acceptance rates (5 datasets each) for different prior distribution bounds. Prior distributions were all uniform, with bounds ranging from (-2,2) to (-10,10). Numbers in parentheses indicate standard deviations. Simulations were run on a dual-socket Dell PowerEdge 1950 (quad-core (8 effective CPUs) Intel Xeon E5410) with 32GB RAM, running RedHat Enterprise Linux 5.4 Server x86-64.

Prior bounds	Time (in hours)	Acceptance rates
(-2,2)	7.7 (0.3)	38.8 (9.2)
(-3,3)	7.7 (0.2)	35.2 (10.5)
(-5,5)	7.8 (0.4)	26.5 (9.0)
(-10,10)	9.5 (2.7)	23.6 (14.1)

(-10,10) as bounds for $\pi(\Theta|G)$; these intervals include somewhat "unrealistic" values for Θ, but are more diffuse (and hence less informative).

As in the previous simulation, we run the ABC-Net method over five independent datasets for each choice of prior bounds to obtain an assessment of the variability in performance. We compare the results from the ABC-Net method based on the four different prior bounds using the AUC (Figure 5.7). Although wider prior bounds seem to be associated with slightly larger variability in terms of this criterion, the differences in average AUC among the four bounds is negligible. However, when comparing the distributions of the Gelman-Rubin statistics ($\hat{R}$) over all potential edges in the network for each setting (Figure 5.8), a noticeable difference can be seen. Specifically, as the bounds of the prior are defined over wider intervals, a greater number of edges in the network exhibit evidence of non-convergence (i.e., $\hat{R} > 1.2$) across the ten independent chains of the ABC-Net method. This is most evident for prior bounds of (-10,10), where a large number of edges exceed the cutoff of 1.2 by a large amount. In addition, the computation time for the prior bounds of (-10,10) is on the average about an hour and a half longer than other choices for these bounds (Table 5.3), since the burn-in period was often extended significantly as a remedy to

poor chain mixing. Consequently, although the ABC-Net algorithm does not appear to be sensitive to most bounds of the prior distribution studied here, the widest interval of (-10,10) leads to problems in chain mixing and convergence.

We also consider the effect of the choice of prior bounds on the approximate posterior distributions in the network. In Figures 5.9 and 5.10, we present a graphical matrix of the marginal approximate posterior distributions of each edge in the network, for prior bounds (-2,2) and (-5,5). As may be expected, the posteriors in Figure 5.10 are generally more diffuse than those of Figure 5.9 due to the wider prior bounds. However, it seems that regardless of the choice of prior bound, some edges consistently have very flat posterior distributions (e.g., the edges in the Pip3 column), while others tend to be consistently peaked (e.g., the edges in the Erk column). This brings us back once again to the idea of flexible and rigid edges, first mentioned in the previous section.

Interestingly, in this network the most rigid edges appear to correspond to regulators that are furthest downstream in the pathway (Mek, Erk, and Akt), while those furthest upstream appear to be the most flexible. In the context of the ABC-Net method, this suggests that rigid edges (e.g., Mek→Erk) in $\Theta^\star$ must take on values within a tight interval in order to generate simulated data $Y^\star$ that are close (in terms of ρ and ϵ) to the observed data Y. Conversely, flexible edges (e.g., Pip3→Pip3) can take on values within a much wider interval without negatively affecting the proximity of simulated and observed data. It is thus likely that the model is most sensitive to parameters with narrow credible intervals (rigid edges) and least sensitive to those that cannot accurately be localized (flexible edges) by the approximate posterior distribution (Toni et al., 2009). That is, some edges may intrinsically be easy to infer even with relatively wide prior bounds (e.g., (-5,5)), while others cannot be accurately determined even with fairly narrow prior distribution bounds (e.g., (-2,2)).

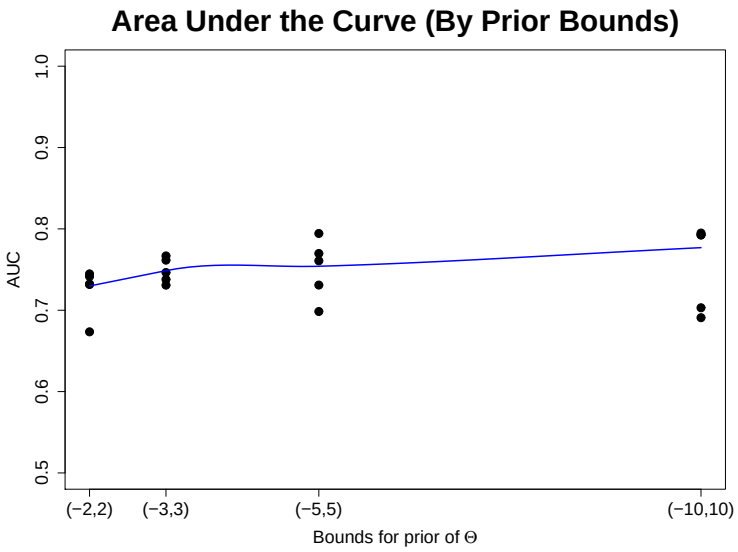


Figure 5.7. Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve for four choices of bounds on the prior distribution $\pi(\Theta|G)$: (-2,2), (-3,3), (-5,5), and (-10,10). Black dots represent the value of the AUC for each of five independent datasets per bound. Blue lines represent loess curves (Cleveland, 1979).

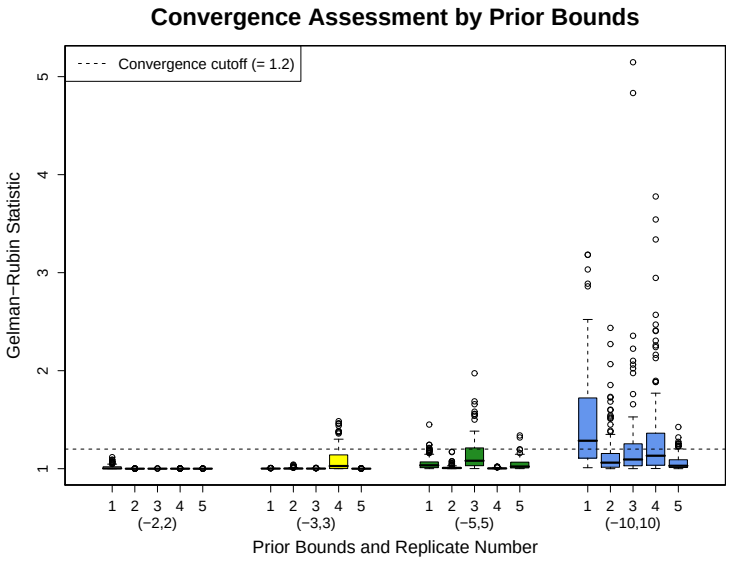


Figure 5.8. Gelman-Rubin statistics ($\hat{R}$) for each replicate of four choices of bounds on the prior distribution $\pi(\Theta|G)$. The black dotted line indicates a value of $\hat{R} = 1.2$, the cutoff at which convergence is declared among ten independent chains in the ABC-Net algorithm.

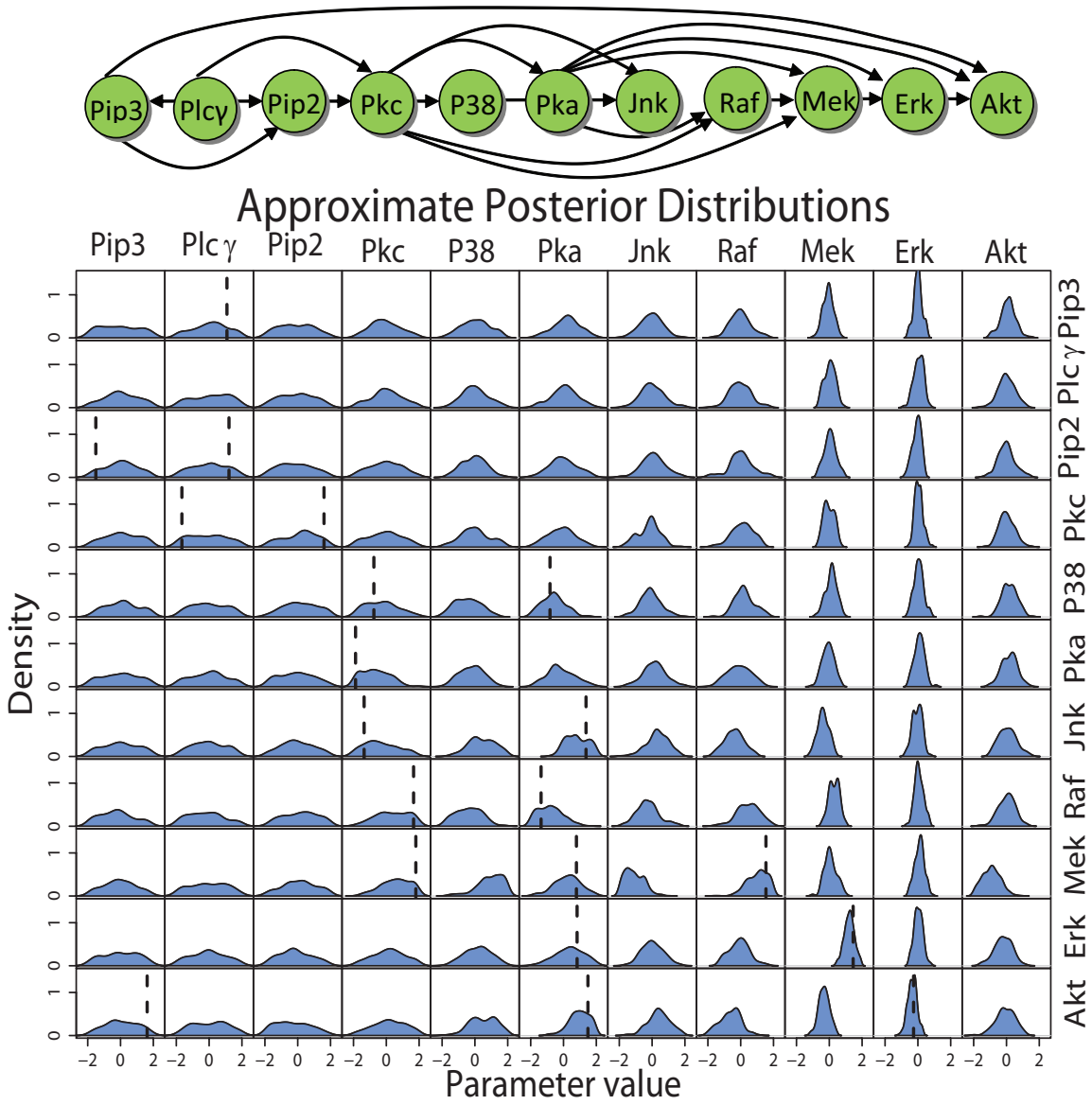


Figure 5.9. The structure of the true Raf signalling pathway, Θ^{Raf}, and a graphical matrix of the marginal approximate posterior distributions for every edge in the network, with prior bounds (-2,2). Each element of the graphical matrix corresponds to the same element of Θ^{Raf}, i.e., the density in the second row and first column corresponds to Θ^{Raf}_{21} (Pip3→Plcγ). The x-axis of each plot represents the values of each parameter Θ^{Raf}_{ij}, and the y-axis represents the corresponding density. Black dotted lines are included on plots where $\Theta^{\text{Raf}}_{ij} \neq 0$ at the true value.

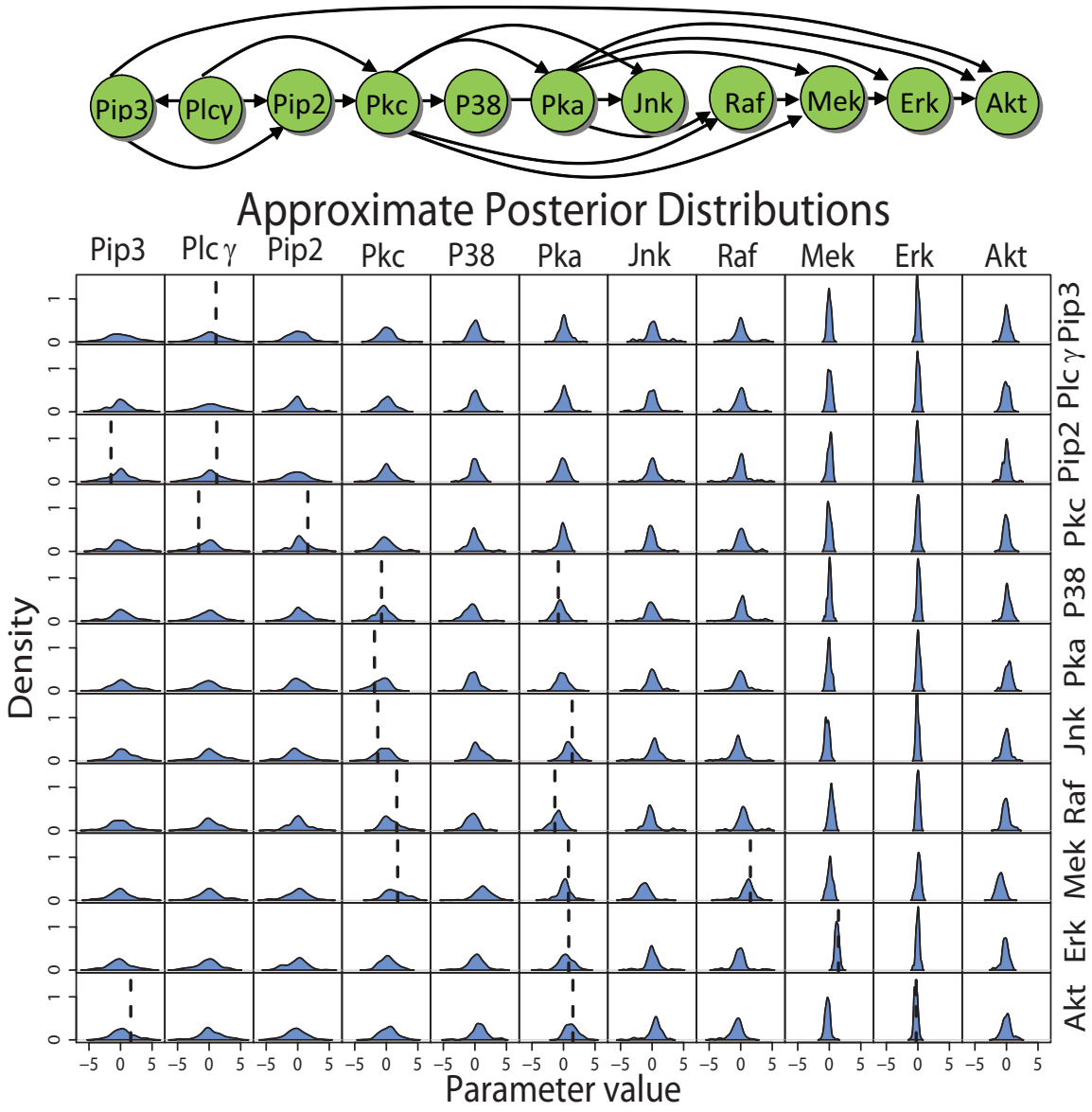


Figure 5.10. The structure of the true Raf signalling pathway, Θ^{Raf}, and a graphical matrix of the marginal approximate posterior distributions for every edge in the network, with prior bounds (-5,5). Each element of the graphical matrix corresponds to the same element of Θ^{Raf}, i.e., the density in the second row and first column corresponds to Θ^{Raf}_{21} (Pip3→Plcγ). The x-axis of each plot represents the values of each parameter Θ^{Raf}_{ij}, and the y-axis represents the corresponding density. Black dotted lines are included on plots where $\Theta^{\text{Raf}}_{ij} \neq 0$ at the true value.

5.7.3 Suitability of VAR(1) simulator

The applicability of the ABC-Net method to real gene regulatory networks depends heavily on its ability to accurately simulate data for a given network structure. In the absence of more detailed information about the dynamics of a particular system, we suggest using a VAR(1) simulator as described in Section 5.1. However, it is likely that real biological systems do not follow a VAR(1) model, and in fact, that they arise from very complicated, nonlinear relationships. In this set of simulations, we attempt to assess how the ABC-Net method reacts when observed data Y are actually generated from models other than the VAR(1) model. In particular, we focus on four models (see Table 5.4): a first-order nonlinear VAR model (VAR-NL(1)), a second-order VAR model (VAR(2)), a second-order nonlinear VAR model (VAR-NL(2)), and an ordinary differential equation (ODE). For the VAR models, Θ_1^{Raf} and Θ_2^{Raf} were each defined using the Raf signalling network (Figure 5.3), where existing edges were sampled uniformly from the interval $(-2, -0.25) \cup (0.25, 2)$ and otherwise set to 0. For the ODE model, coefficients were randomly drawn from a $\mathcal{U}(-1, 1)$ distribution and initial values for all genes were set to 1. After solving the ordinary differential equations for time points $t = 1, \ldots, 20$, random noise sampled from $N(0, 1)$ was added to each measurement at each time point.

We apply the ABC-Net algorithm to data Y generated from the four alternative models described in Table 5.4 and examine the AUC results for each (Figure 5.11). In this comparison, we also include the VAR(1) model from Equation (5.10) as a reference. We note that the ABC-Net has the best performance in terms of AUC for the VAR(1) model; this is unsurprising as the data Y are generated with the same model that is used to simulate data $Y^\star$ (Section 5.1). For the other models, the performance of the algorithm noticeably declines, with the lowest AUC values observed for the two second-order models, VAR(2) and VAR-NL(2). The nonlinear first-order VAR model shows wide variability in its results, ranging from an AUC of just over 0.40 to over 0.70. Of the alternative models, the ordinary differential equation appears to have

Table 5.4
Alternative models used to generate observed data Y: an ordinary differential equation (ODE), a second-order VAR model (VAR(2)), a first-order nonlinear VAR model (VAR-NL(1)), and a second-order nonlinear VAR model (VAR-NL(2)).

Model	Network Equations to Generate $\mathbf{y}$
VAR-NL(1)	$\mathbf{y}_1 = \mathbf{z}_1$ $\mathbf{y}_t = \Theta_1^{\text{Raf}} \mathbf{y}_{t-1}^{-1} + \mathbf{z}_t$, for $t = 2, \ldots, T$ $\mathbf{z}_t \sim N(0,1)$ for $t = 1, \ldots, T$
VAR(2)	$\mathbf{y}_1 = \mathbf{z}_1$ $\mathbf{y}_2 = \Theta_1^{\text{Raf}} \mathbf{y}_1 + \mathbf{z}_2$ $\mathbf{y}_t = \Theta_1^{\text{Raf}} \mathbf{y}_{t-1} + \Theta_2^{\text{Raf}} \mathbf{y}_{t-2} + \mathbf{z}_t$, for $t = 3, \ldots, T$ $\mathbf{z}_t \sim N(0,1)$ for $t = 1, \ldots, T$
VAR-NL(2)	$\mathbf{y}_1 = \mathbf{z}_1$ $\mathbf{y}_2 = \Theta_1^{\text{Raf}} \mathbf{y}_{t-1}^{-1} + \mathbf{z}_2$ $\mathbf{y}_t = \Theta_1^{\text{Raf}} \mathbf{y}_{t-1}^{-1} + \Theta_2^{\text{Raf}} \mathbf{y}_{t-2} + \mathbf{z}_t$, for $t = 3, \ldots, T$ $\mathbf{z}_t \sim N(0,1)$ for $t = 1, \ldots, T$
ODE	$y'_{\text{Pkc}} = 0.18 y_{\text{Plc}\gamma} - 0.75 y_{\text{Pip2}}$ $y'_{\text{Raf}} = -0.28 y_{\text{Pkc}} + 0.62 y_{\text{Pka}}$ $y'_{\text{Mek}} = 0.63 y_{\text{Pkc}} - 0.97 y_{\text{Raf}} - 0.52 y_{\text{Pka}}$ $y'_{\text{Erk}} = 0.70 y_{\text{Mek}} - 0.94 y_{\text{Pka}}$ $y'_{\text{Pka}} = 0.31 y_{\text{Pkc}}$ $y'_{\text{Akt}} = 0.28 y_{\text{Erk}} + 0.60 y_{\text{Pka}} + 0.92 y_{\text{Pip3}}$ $y'_{\text{P38}} = -0.19 y_{\text{Pkc}} - 0.32 y_{\text{Pka}}$ $y'_{\text{Jnk}} = 0.24 y_{\text{Pkc}} + 0.98 y_{\text{Pka}}$ $y'_{\text{Plc}\gamma} = 0$ $y'_{\text{Pip3}} = -0.28 y_{\text{Plc}\gamma}$ $y'_{\text{Pip2}} = 0.83 y_{\text{Plc}\gamma} - 0.98 y_{\text{Pip3}}$

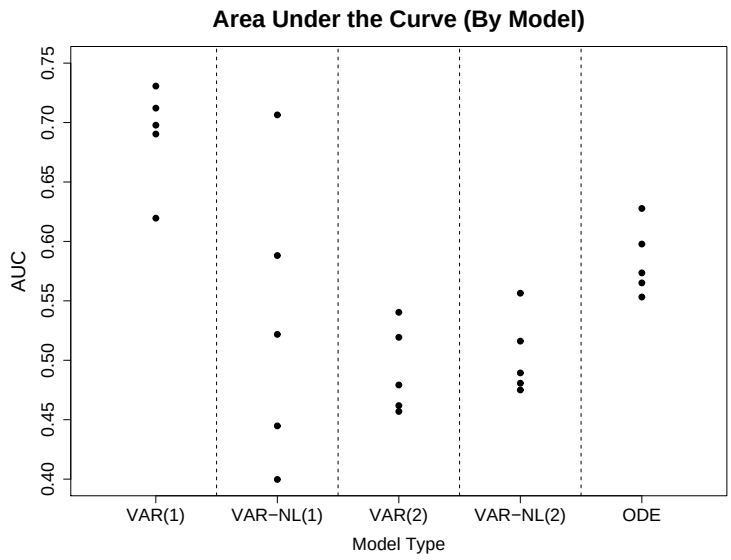


Figure 5.11. Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve for five different model choices to generate Y: VAR(1), VAR-NL(1), VAR(2), VAR-NL(2), and ODE. Black dots represent the value of the AUC for each of five independent datasets per bound.

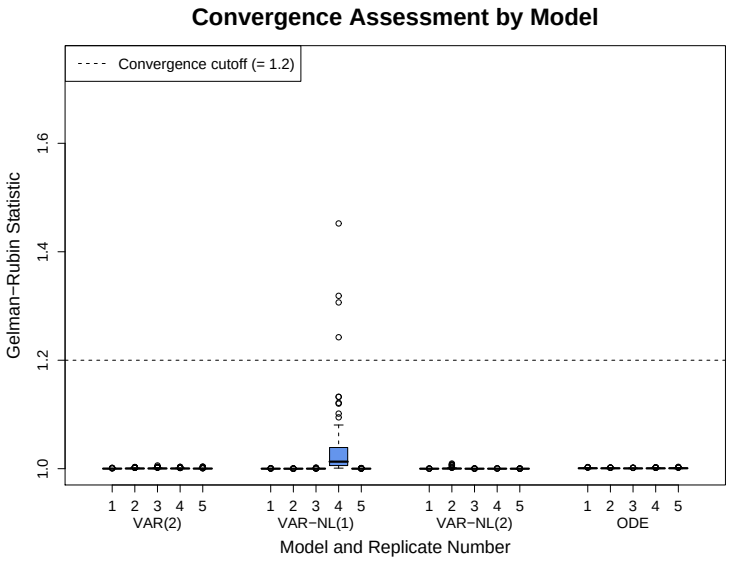


Figure 5.12. Gelman-Rubin statistics ($\hat{R}$) for each replicate of the four alternate models: VAR-NL(1), VAR(2), VAR-NL(2), and ODE. The black dotted line indicates a value of $\hat{R} = 1.2$, the cutoff at which convergence is declared among ten independent chains in the ABC-Net algorithm.

the highest performance in terms of AUC. Although one of the datasets simulated with the VAR-NL(1) displayed some evidence of non-convergence (Figure 5.12), all other models displayed sufficiently small values for this statistic.

We also consider the approximate posterior distributions in the network for model used to generate Y. In Figure 5.13, we present a graphical matrix of the marginal approximate posterior distributions of each edge in the simulated Raf signalling pathway, for each choice of model. The shape of the densities for each model varies widely. For instance, the VAR-NL(1) model tends to yield more peaked distributions (not necessarily centered about the true value), while the ODE displays relatively flat distributions for edges throughout the network. The two nonlinear models tend to have similar posterior distributions, as do the VAR(2) model and ODE. Throughout the network, it appears that different models are able to more easily identify different edges. For example, the VAR-NL(1), VAR(2), VAR-NL(2), and ODE models each seem to uniquely identify the Pka→Mek, Mek→Erk, Pip→Akt, and Pka→Jnk edges, respectively. However, the presence or absence of flexible and rigid edges in the inferred network clearly depends on the type of model used to generate the data Y.

As a final note concerning the performance of the ABC-Net algorithm when alternative models are used to generate Y, recall that the simulator described in Section 5.1 has the flexibility to incorporate alternative models, provided they are computationally efficient. We recommend the use of a VAR(1) model as a simple and efficient simulator in situations where such a simplification of the network dynamics would be appropriate. However, in cases where other models are known to better fit a given set of data (e.g., a second order or non-linear model), the ABC-Net method can be adapted accordingly.

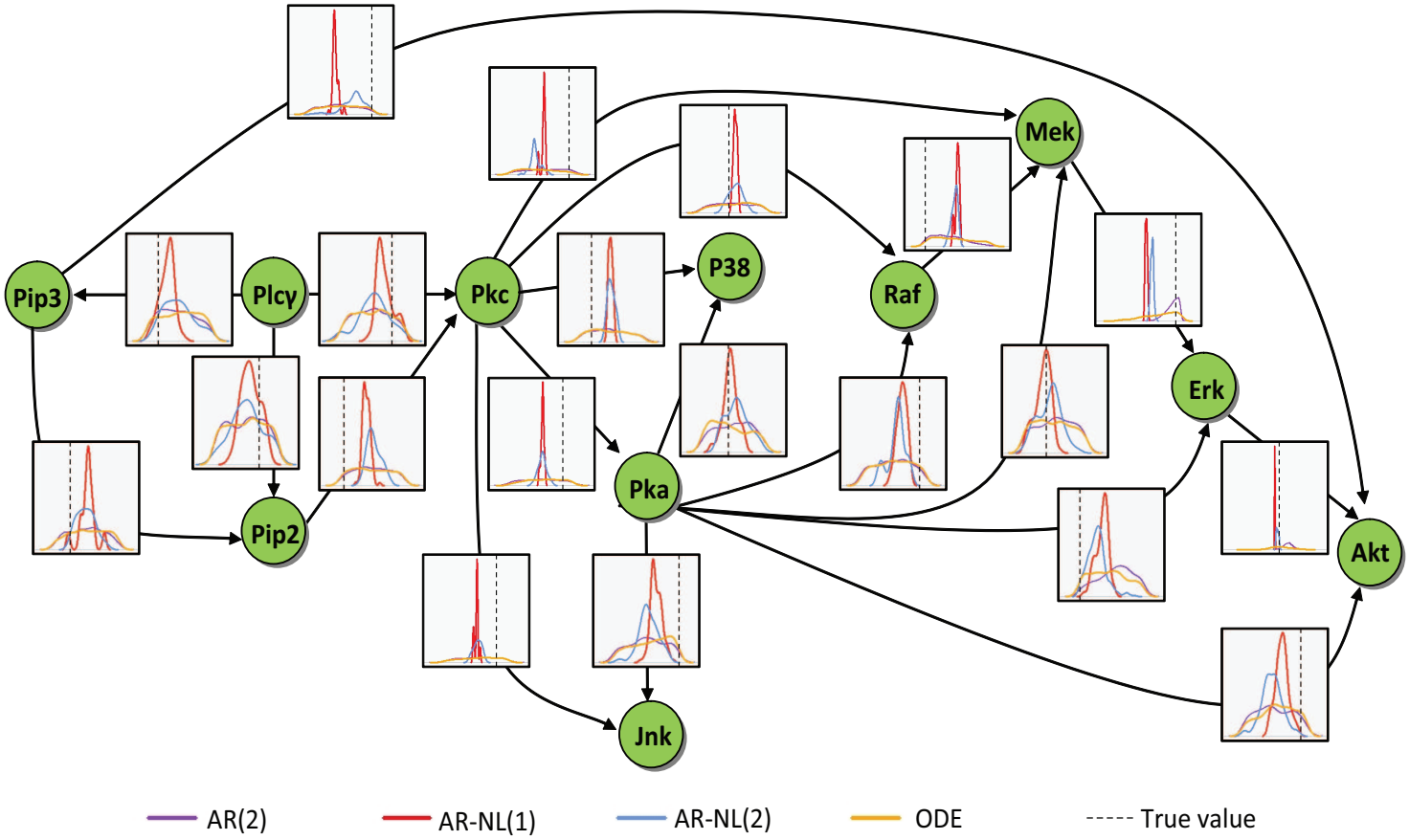


Figure 5.13. Approximate posterior distributions from the ABC-Net method for the simulated Raf signalling pathway, by model. Each plot overlapping a given edge in the network corresponds to the marginal approximate posterior distribution for that edge, where the x-axis represents the values taken on by the edge over the interval (-2,2) and the y-axis is the density. Each color represents a different model, as shown in the legend. Black dotted lines indicate the true value taken on by the edge in the true network Θ^{Raf}.

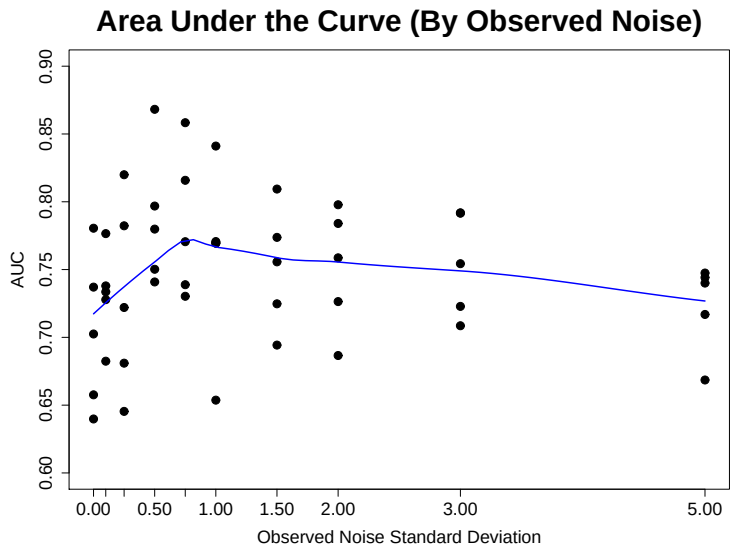


Figure 5.14. Scatterplots of the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve for the ABC-Net algorithm, with differing values of noise standard deviation σ (0, 0.1, 0.25, 0.5, 0.75, 1, 1.5, 2, 3, 5). Five datasets were generated for each value of noise standard deviation. The blue line represents a loess curve (Cleveland, 1979).

5.7.4 Effect of noise in observed data

We also examine the effect of increased variability in the random noise $\mathbf{z}_t$ added to each time point of the observed data (see Equation (5.10)), where $\mathbf{z}_t \sim N(0, \sigma)$. We would expect that increasing amounts of noise lead to reduced performance for the ABC-Net algorithm, particularly since the VAR(1) simulator described in Section 5.1 uses one-step ahead predictors to simulate data based on a given network $\Theta^\star$. For this purpose, we consider $\sigma = \{0, 0.1, 0.25, 0.5, 0.75, 1, 1.5, 2, 3, 5\}$. For each value of σ, we generate five independent datasets to assess the variability of results, and run the ABC-Net algorithm as detailed in Algorithm 5.1. The Gelman-Rubin statistics for all parameters were less than 1.2 in all simulation settings, and as such we may assume that the output from the ABC-Net method is truly made up of samples drawn from the stationary distribution of the chain.

The results for the AUC (Figure 5.14) indicate that the presence of increasing noise over the investigated range does seem to negatively affect the performance of the ABC-Net algorithm, although only for relatively large values of σ (e.g., $\sigma = 5$). The mean AUC (averaged over 5 datsets per noise setting) ranges from 0.70 (at $\sigma = 0$) to 0.79 (at $\sigma = 0.5$). Interestingly, the worst performance appears to fall at the two extremes for σ (i.e, 0 and 5). As the noise standard deviation is increased, it is not surprising that the performance of the algorithm deteriorates, since the one-step-ahead predictors described in Section 5.1 would fall increasingly further from the observed data (even when the true network is used). However, the result observed for $\sigma = 0$ is somewhat less intuitive. To explain these findings, consider the toy problem described in Example 5.1. For this example, because of the sparsity of the network structure, the only time points that yield any information about the true "closeness" of simulated and actual data are $t = 1$, 2, and 3, even when more time points are simulated. That is, regardless of the proposed network $\Theta^\star$ in the ABC-Net method, the one-step ahead predictors for times $t \geq 5$ are always exactly equal to the observed values, as all are exactly equal to 0. Because the Raf signalling network Θ^{Raf} used in the simulations is also very sparse (only 20 out of 121 possible edges are actually present in the network), this helps justify the results observed for $\sigma = 0$ in Figure 5.14.

As a comparison, we also examine the approximate posterior distributions of the network for two different values of noise standard deviation, $\sigma = 0.5$ and $\sigma = 5$ (Figure 5.15). For the most part, posterior distributions for both $\sigma = 0.5$ and $\sigma = 5$ seem to have the same general shape, with some occasional discrepancies (e.g., Akt→Akt and Akt→Erk). In addition, as in previous simulations, we note once again the marked difference in posterior distributions between rigid edges (peaked distributions) and flexible edges (diffuse distributions). Regardless of the amount of noise incorporated into the simulated data for the Raf signalling pathway, the approximate posterior distributions for the upstream and downstream portions of the network are consistently flexible and rigid, respectively. This seems to indicate that

Example 5.1 Sparse network of 3 genes, with no noise added ($\sigma = 0$).

Consider a simple network made up of 3 genes and 3 edges, where the parameter matrix Θ is defined by

$$\Theta = \begin{pmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 2 & 0 & 0 \end{pmatrix}.$$

Assume the expression values for the three genes at the first time point are $\mathbf{y}_1 = (y_{11}, y_{21}, y_{31})$. Using the VAR(1) model of Equation 2.6, if no noise is present in the data, the expression values at the second time point will be equal to

$$\mathbf{y}_2 = \Theta\mathbf{y}_1 = (0, -1y_{11} + y_{31}, 2y_{11}).$$

Following suit, the expression values at the third time point would be

$$\mathbf{y}_3 = \Theta\mathbf{y}_2 = (0, 2y_{11}, 0)$$

and those at the fourth time point (and all following time points) would be

$$\mathbf{y}_4 = \Theta\mathbf{y}_3 = (0, 0, 0).$$

---

some edges are intrinsically easier to infer (even in the presence of increased noise), while other cannot be accurately determined regardless of the amount of noise in the data. As such, the flexibility and rigidity of edges in a given system likely plays an important role in the global inferability of the network structure.

5.7.5 Including Prior Biological Information

One advantage of Bayesian approaches such as the ABC-Net method is that biological knowledge can easily be incorporated into the prior structure. To illustrate this, we incorporate prior knowledge for an increasing number of edges ($n = 0$, 1, 2, 3,

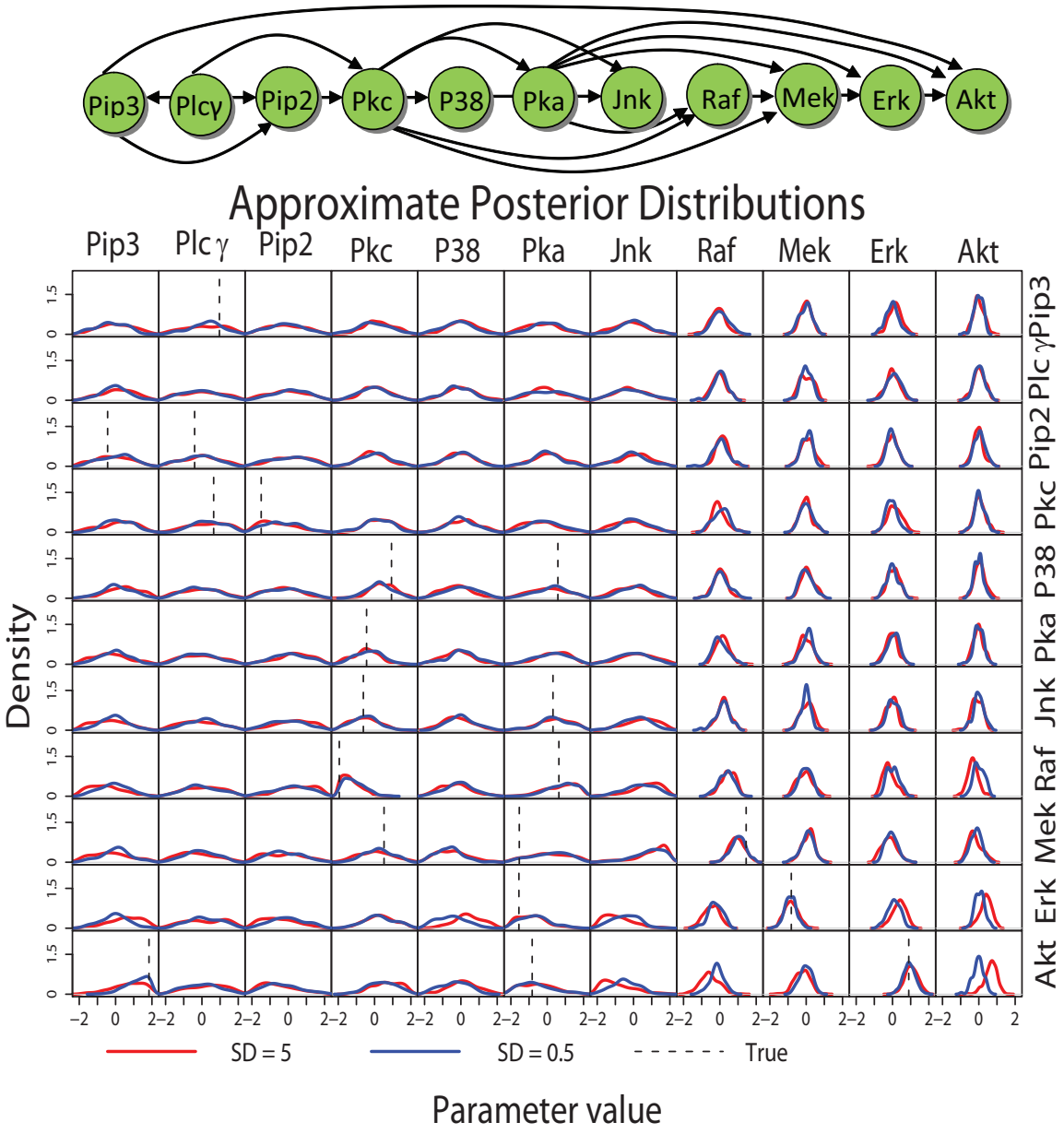


Figure 5.15. The structure of the true Raf signalling pathway, Θ^{Raf}, and a graphical matrix of the marginal approximate posterior distributions for every edge in the network, for $\sigma = 0.5$ and 5. Each element of the graphical matrix corresponds to the same element of Θ^{Raf}, i.e., the density in the second row and first column corresponds to Θ^{Raf}_{21} (Pip3→Plcγ). The x-axis of each plot represents the values of each parameter Θ^{Raf}_{ij}, and the y-axis represents the corresponding density. Red and blue lines correspond to results obtained with $\sigma = 5$ and $\sigma = 0.5$, respectively. Black dotted lines are included on plots where $\Theta^{\text{Raf}}_{ij} \neq 0$ at the true value.

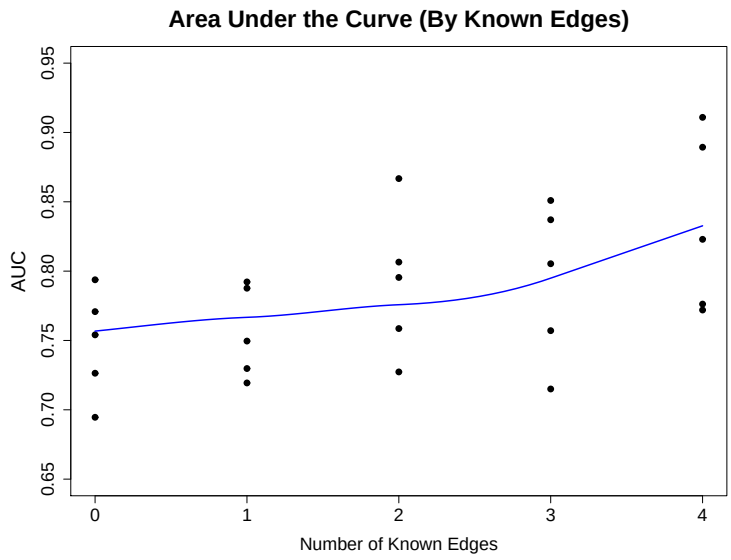


Figure 5.16. Scatterplots of the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve for the ABC-Net algorithm, with differing values of known edges (0, 1, 2, 3, 4). Five datasets were generated for each number of known edges. The blue line represents loess curves (Cleveland, 1979).

and 4) in the simulated Raf signalling pathway in the following way. If an edge in the Raf network from gene j to gene i is randomly selected to be "known", its prior distribution is bounded to be $\pi(\theta_{ij}^{\text{Raf}}|G) \sim \mathcal{U}(0,2)$ for activations and $\pi(\theta_{ij}^{\text{Raf}}|G) \sim \mathcal{U}(-2,0)$ for repressions. Not surprisingly, the performance of the ABC-Net algorithm improves markedly when prior information is included in this manner (Figure 5.16). The average AUC increases from about 75% when no known edges are included in the model to almost 82% when 4 known edges are included. These improvements are directly linked to the fact that using accurate informative prior distributions is certain to improve algorithm performance.

We also examine the effect of prior biological knowledge on the approximate posterior distributions over the entire network (Figure 5.17). It turns out that including prior biological information about a subset of four edges did not seem to tighten credible intervals for the flexible edges in the network. This observation held

whether known edges were defined at random or chosen among the edges supported by the strongest evidence in the previous simulations (e.g., Pkc→Raf, Raf→Mek, and Mek→Erk). The fact that these flexible edges cannot be inferred even in the presence of partial network information highlights the difficulties in achieving accurate, complete inference in the context of gene regulatory networks. Finally, we also stress that incorporating prior biological information as shown here relies on the assumption that such information is entirely accurate. If this is not the case, alternative modifications to the prior structure must be made to account for uncertainty in known gene-to-gene interactions.

5.8 Summary

Methods based in approximate Bayesian computation (ABC) are a valuable tool that enable Bayesian inference in complex, high-dimensional problems for which the likelihood is difficult to calculate. The ABC-Net algorithm presented here is a novel framework for sampling from the approximate posterior distributions of gene-to-gene interactions involved in gene regulatory networks. These approximate posterior distributions provide a wealth of information about the structure and inferability of complicated biological systems, particularly with respect to the flexibility and rigidity of network edges. For the time being, the computing time required for the ABC-Net approach limits its application to small networks. For example, for the 11-gene Raf signalling network used in the simulations, average computing time for 1×10^7 iterations was on the order of 7 hours (simulations were run on a dual-socket Dell PowerEdge 1950 (quad-core (8 effective CPUs) Intel Xeon E5410) with 32GB RAM, running RedHat Enterprise Linux 5.4 Server x86-64). However, as computing power continues to improve in terms of speed and memory, it is anticipated that the method will progressively be able to handle somewhat larger networks.

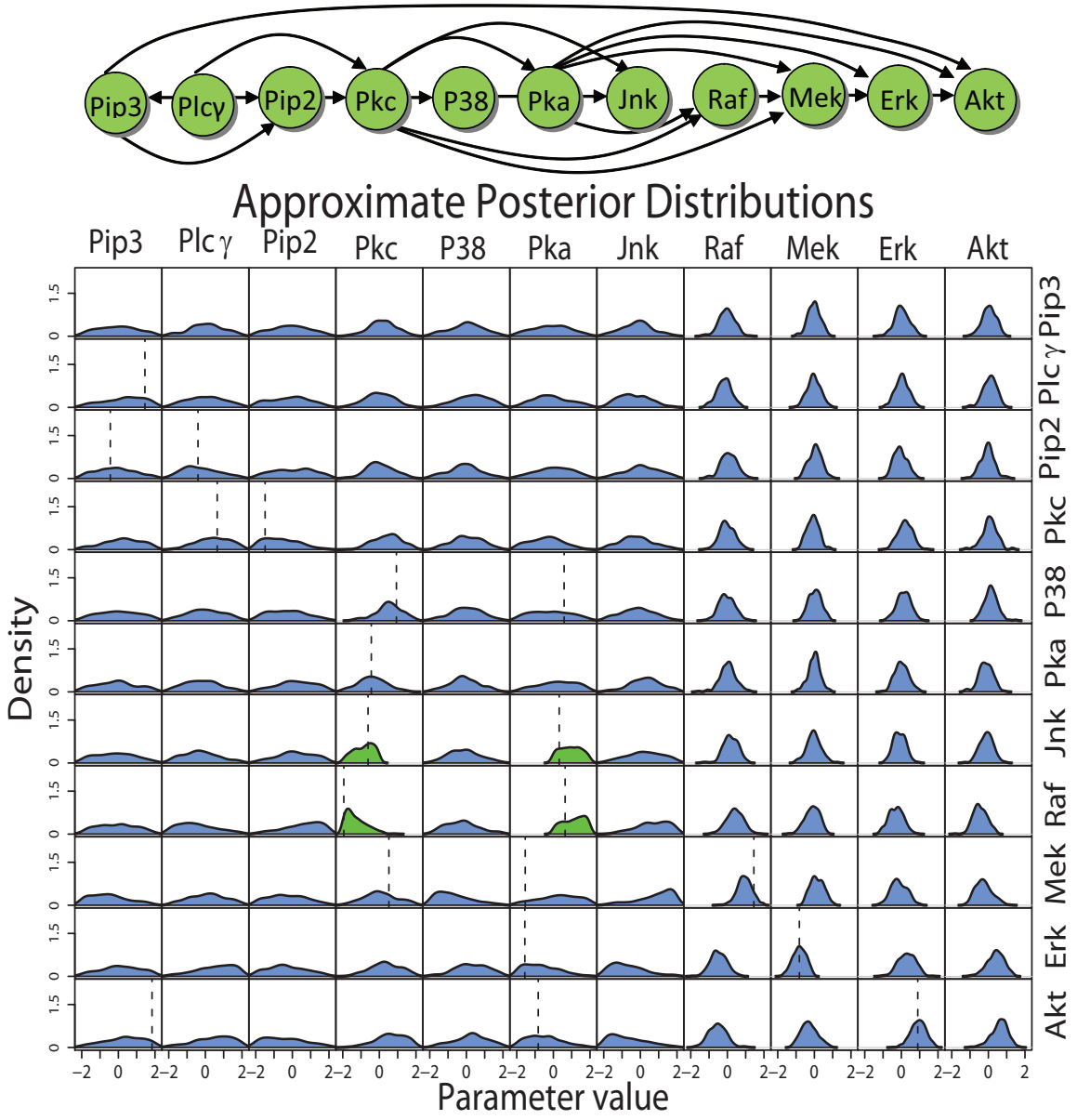


Figure 5.17. The structure of the true Raf signalling pathway, Θ^{Raf}, and a graphical matrix of the marginal approximate posterior distributions for every edge in the network, with four known edges. Each element of the graphical matrix corresponds to the same element of Θ^{Raf}, i.e., the density in the second row and first column corresponds to Θ^{Raf}_{21} (Pip3→Plcγ). The x-axis of each plot represents the values of each parameter Θ^{Raf}_{ij}, and the y-axis represents the corresponding density. Densities shaded in green and blue correspond to edges considered to be "known" and "unknown" in the simulation, respectively. Black dotted lines are included on plots where $\Theta^{\text{Raf}}_{ij} \neq 0$ at the true value.

6. APPLICATION TO REAL DATA

Wide-scale experiments designed to study gene expression in a variety of contexts have become routine in the past decade due to the development of high-throughput technologies such as microarrays, serial analysis of gene expression, and RNA Sequencing. As an example, one of the most well-known studies of time-course gene expression, a study of the cell cycle in *Saccharomyces cerevisiae* based on microarray technology (Spellman et al., 1998), is now over ten years old. To date, most longitudinal studies of gene expression have been conducted on microarrays, although the decreasing cost of next generation sequencing technologies will likely lead to an upsurge of longitudinal RNA-Seq experiments in the next few years. To illustrate the utility of the two proposed approaches, the Empirical Bayes Dynamic Bayesian Network (EBDBN) method (Chapter 4) and the ABC-MCMC for Networks (ABC-Net) method (Chapter 5), we apply each to real longitudinal expression data.

6.1 T-Cell Activation Network in Humans

Although several time-series microarray datasets are publicly available in simple organisms such as *Saccharomyces cerevisiae* (Spellman et al., 1998; Lee et al., 2004), *Escherichia coli* (Ronen et al., 2002), and *Drosophila melanogaster* (Arbeitman et al., 2002), few microarray experiments include the necessary level of replication and time points to effectively infer network structure. One exception arises from a set of experiments conducted by Rangel et al. (2004) to investigate the effect of two treatments (a phorbal ester called PMA and a calcium ionophore called ionomicin) on human T-cell response in a lymphoblast cell line. The data are made up of temporal gene expression measurements using cDNA microarrays on 58 genes over 10 unequally spaced time points (0, 2, 4, 6, 8, 18, 24, 32, 48, and 72 hours after treatment) on 34 biological repli-

cates. A second related experiment under identical conditions added ten additional biological replicates for the same set of genes and time points, yielding a total of 44 biological replicates between the two related experiments. Genes were pre-selected for modulation following the T-cell activation, meaning that genes with weak expression or low reproducibility across replicates were removed from the dataset. In addition, the data were log-transformed and quantile normalized by Rangel et al. (2004). The pre-processed expression measurements and gene descriptions may be found in the `R` package `GeneNet` (Schäfer et al., 2006) on CRAN (R Development Core Team, 2009) as well as on the authors' website (`http://public.kgi.edu/~wild/LDS/index.htm`). The longitudinal expression measurements for the first nine genes in the T-cell activation dataset are displayed in Figure 6.1.

Because the Rangel et al. (2004) data are longitudinal and highly replicated, they have been used by several authors to assess the performance of inference methods for gene regulatory networks (Rangel et al., 2004; Beal et al., 2005; Opgen-Rhein and Strimmer, 2006). We follow suit, and use these data to compare the results of our proposed Empirical Bayes Dynamic Bayesian Network (EBDBN) method from Chapter 4 to those of previous established methods. Specifically, we compare the EBDBN method with and without hidden states (denoted EBDBN(x) and EBDBN(-), respectively), the Variational Bayes State Space Model (VBSSM) algorithm of Beal et al. (2005), and the shrinkage-based Vector Autoregressive (VAR) method of Opgen-Rhein and Strimmer (2007), all described in greater detail in Section 4.7.1. This set of methods provides an interesting comparison, as the EBDBN(x) and VBSSM are based on the same model (Equation (2.2)) with different estimation procedures, as are the EBDBN(-) and VAR models. In this particular example, the ABC-Net method of Chapter 5 is not included in the comparison. The reason for this is twofold. First, a network of this size (58 genes, with a total of $58^2 = 3364$ possible gene-to-gene interactions) is best studied in an exploratory manner, rather than the more detailed analysis provided by the ABC-Net method. Second, without detailed prior informa-

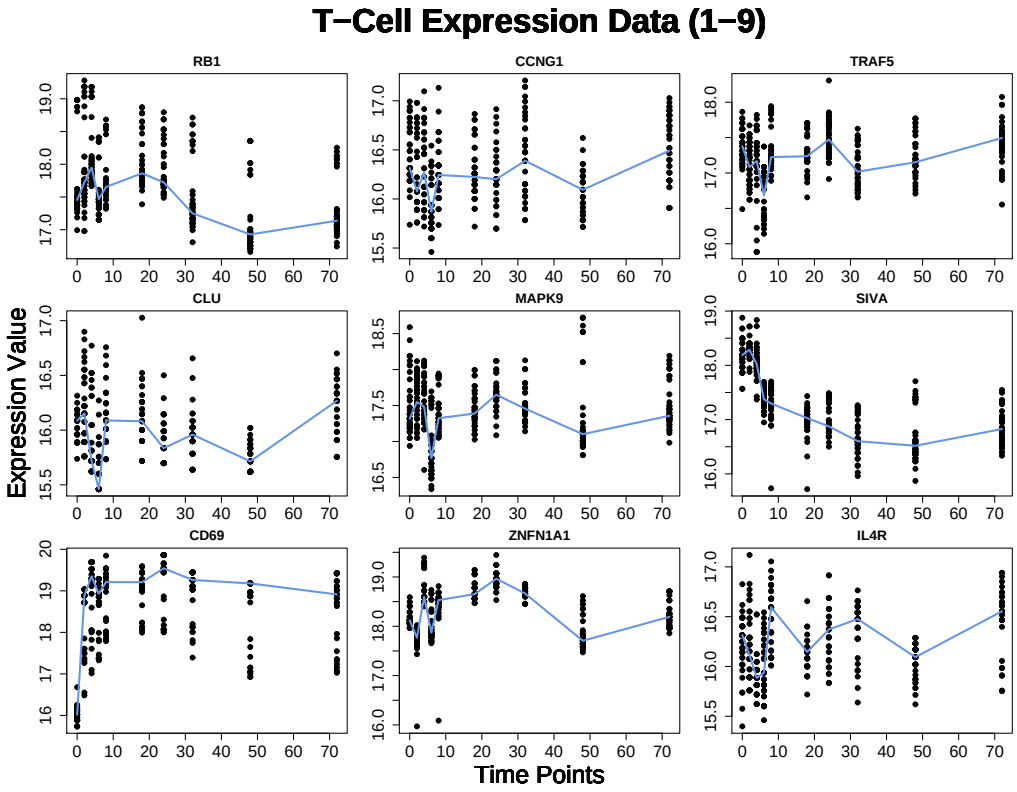


Figure 6.1. Expression measurements after pre-processing for the first nine genes in the T-cell activation data (Rangel et al., 2004): RB1, CCNG1, TRAF5, CLU, MAPK9, SIVA, CD69, ZNFN1A1, and IL4R. Grey dots represent the expression values at each time point for each of the 44 biological replicates, and blue lines are drawn at the median expression value across replicates for each time point.

tion about specific network edges, these data would incur a substantial computational burden in the context of the ABC-Net method, both in terms of time and memory.

Before the EBDBN method can be applied to the Rangel et al. (2004) data, we must determine an appropriate value for the hidden state dimension K. In previous work, this value was chosen to be $K = 9$ (Rangel et al., 2004) and $K = 14$ (Beal et al., 2005). Recall that the maximum number of nonzero singular values taken from the decomposition of the block-Hankel matrix H is $T - 1$, or in this case, 9 (see Section 4.1). For each of the 44 replicates, we construct H_r, apply the singular value

decomposition, and plot the singular values (see Figure 6.2). In doing so, we determine the optimal value for the hidden state dimension to be $K = 4$. We subsequently apply the EBDBN($K = 4$) and EBDBN(-) methods for 10 different sets of initial values for hyperparameters ψ, hidden states $\mathbf{x}_1$, and gene precisons $\mathbf{v}$. A 99.9% cutoff is used as a threshold for the z-scores of the edges, and only edges identified in 80% of the runs are retained for the final network structure. For the VBSSM and VAR methods, cutoff values of 99.9% for the z-scores and 80% for the local fdr correction are used, respectively.

For all methods compared here, the effect of each gene-to-gene interaction is determined by the sign of the significant elements of the Θ matrix (see Equations (2.2, (2.3), and (2.6)). That is, activations are represented by positive elements of Θ, and repressions by negative elements. The number of edges (positive, negative, and total) found by each method in the T-cell activation data is displayed in Table 6.1. The EBDBN method (both with and without hidden states) appears to identify more edges than the VAR and VBSSM method, although all methods identify a greater number of activations than repressions. Because the results of the methods under comparison seem to vary, we also consider their overlap in the four-way Venn diagram in Figure 6.3. We note that only 13 edges are selected by all four methods under consideration, largely due to the small number (15) of edges identified by the VAR method. There are 93 edges identified by at least three of the methods, and 371 by at least two of the methods. Several edges are identified by only a single method; there are 223, 210 and 185 such uniquely identified edges for the EBDBN(x), EBDBN(-), and VBSSM, respectively.

It is somewhat surprising that the results of the four methods under comparison differ so widely, particularly given that the approaches are all based on similar models. For instance, although the EBDBN(x) and VBSSM methods resemble each other quite closely with the exception of their respective estimation procedures, over 60% of the interactions identified by one (64% for the former, 68% for the latter) are not identified by the other in the T-cell activation data. Because these results differ by such a large

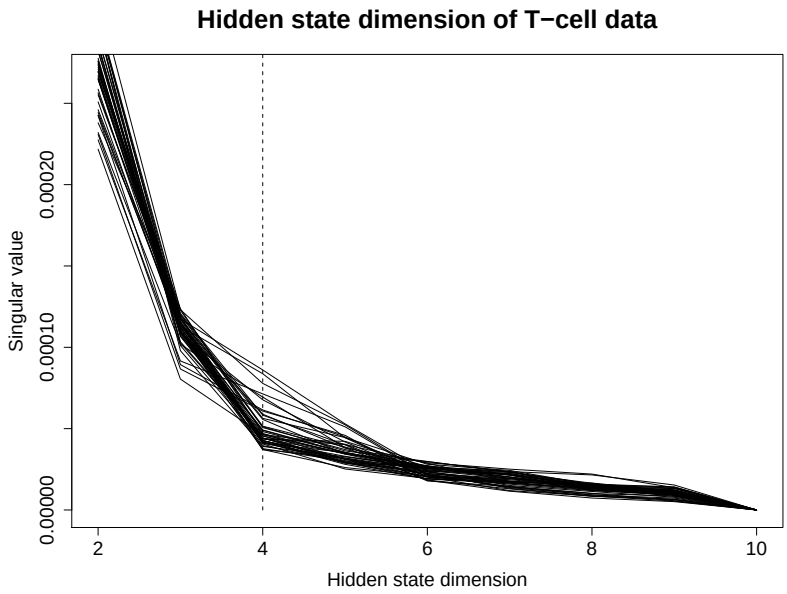


Figure 6.2. Singular values after decomposition of the block-Hankel matrices H_r for each replicate ($r = 1, \ldots, 44$) in the T-cell activation data (Rangel et al., 2004). Each line represents the singular values for one of the 44 biological replicates. As the "elbow" of the plot appears to fall at $K = 4$ for the majority of biological replicates, this value is chosen to be the hidden state dimension for the EBDBN method.

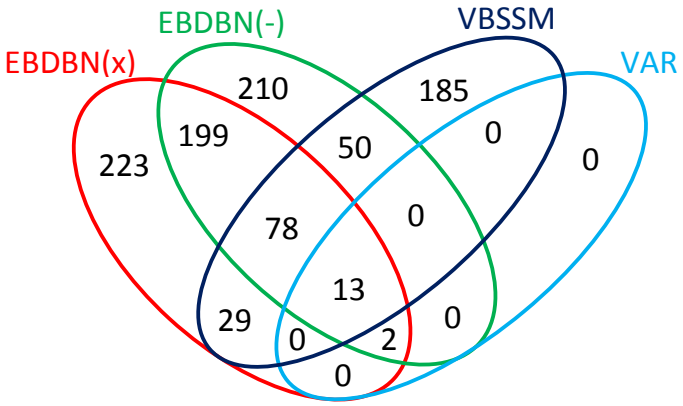


Figure 6.3. Four-way Venn Diagram of edges identified by the EBDBN(x), EBDBN(-), VBSSM, and VAR in the T-cell activation data of Rangel et al. (2004). Although there is some overlap among the four methods, there are a large number of edges that are uniquely identified by a single method (e.g., 223 edges are identified only by the EBDBN(x), 210 edges are identified only by the EBDBN(-), and 185 edges are identified only by the VBSSM.)

Table 6.1
Positive, negative, and total edges (and percent of all possible edges) found for T-cell activation data, by method: the Empirical Bayes Dynamic Bayesian Network with hidden states (EBDBN(x)), the Empirical Bayes Dynamic Bayesian Network method without hidden states (EBDBN(-)), the Variational Bayes State Space Model (VBSSM) of Beal et al. (2005), and the shrinkage Vector Autoregressive model (VAR) of Opgen-Rhein and Strimmer (2007).

Method	# Positive	# Negative	Total Edges (%)
EBDBN(x)	435	109	544 (16.2)
EBDBN(-)	338	214	552 (16.4)
VBSSM	233	122	355 (10.6)
VAR	9	6	15 (0.4)

Table 6.2
Description of seven most important regulator genes in the T-cell activation network, as determined by the consensus of the EBDBN(x) and VBSSM methods. Genes listed correspond to those in yellow in Figure 6.4.

Number	Gene Name	Description
7	CD69	CD69 antigen (p60, early T-cell activation antigen)
17	SMN1	Survival of motor neuron 1, telomeric
21	CCNC	Cyclin C
28	EGR1	Early growth response 1
38	CASP4	Caspase 4, apoptosis-related cysteine protease
41	GATA4	GATA-binding protein 3
46	IL2RG	Interleukin 2 receptor, gamma (severe combined immunodeficiency)
48	MPO	Myeloperoxidase

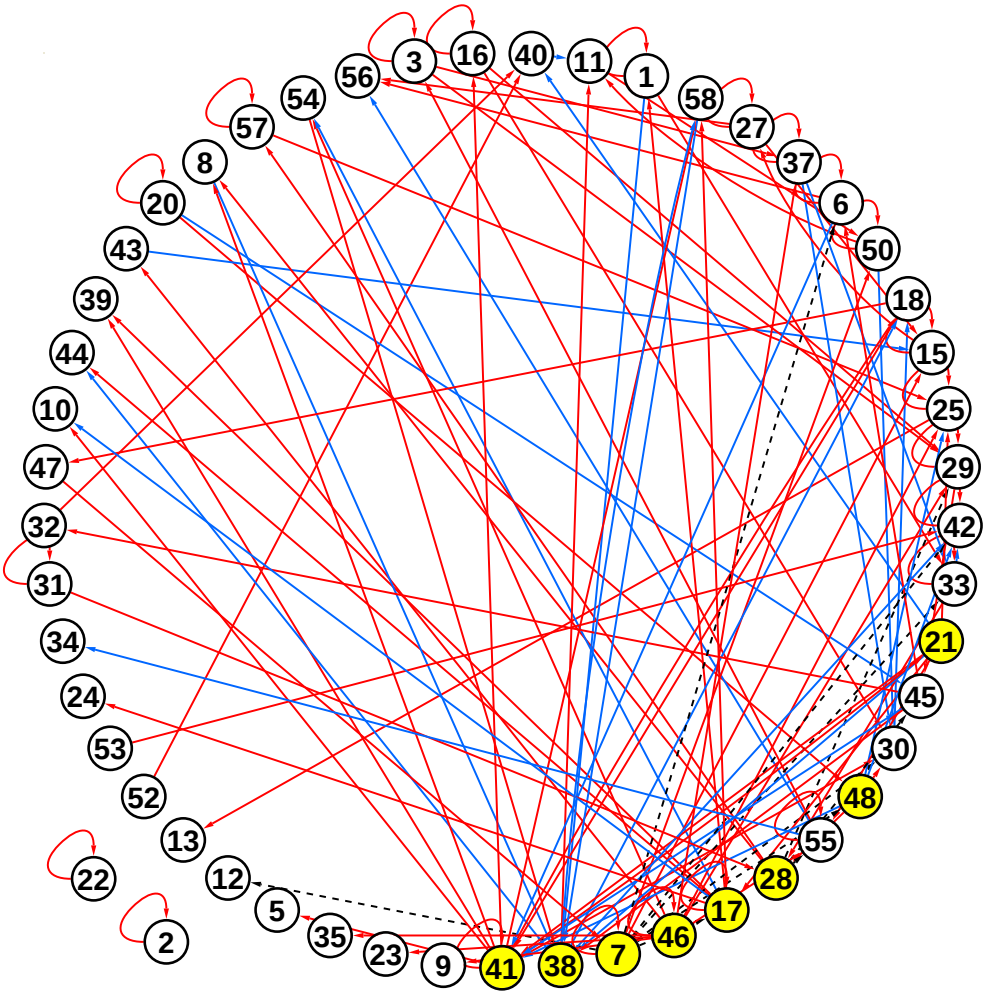


Figure 6.4. Edges inferred by both the EBDBN($K = 4$) and VBSSM methods for the T-cell activation data (Rangel et al., 2004). Nodes represent genes, with numbers corresponding to those found in the `R` package `GeneNet` of Schäfer et al. (2006). Blue solid lines represent inhibitory regulations, red solid lines activatory regulations, and black dotted lines edges with ambiguous regulation (i.e., disagreement between the EBDBN and VBSSM methods). Yellow nodes have five or more regulatory interactions with other genes, indicating important regulator genes in the network topology.

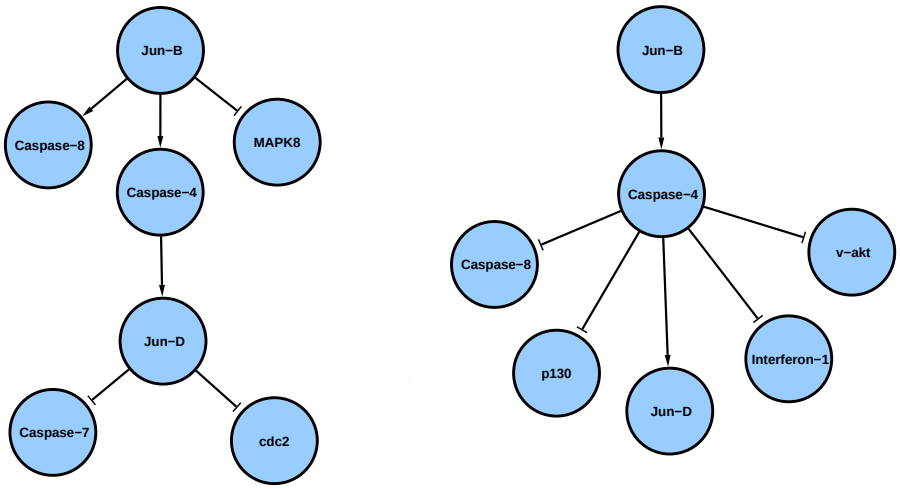


Figure 6.5. Sub-network found representing the interaction between genes in the Jun protein family and genes involved in programmed cell death. (Left) Subnetwork proposed in Beal et al. (2005). (Right) Subnetwork proposed by consensus of VBSSM and EBDBN($K = 4$).

extent, we focus on the common sub-network identified by both the EBDBN(x) and VBSSM methods, composed of a total of 120 edges among 51 genes (Figure 6.4). Of these edges, 26 represent repressions, 84 represent activations, and 10 have an ambiguous regulatory effect (i.e., the two methods do not agree with respect to the type of interaction). The structure of this so-called "consensus network" is visualized using the software application Cytoscape (Shannon et al., 2003), shown in Figure 6.4.

Further focusing on portions of the consensus network shown in Figure 6.4 can yield additional pertinent information. For example, eight genes (numbered 7, 17, 21, 28, 38, 41, 46, and 48, and colored in yellow in the figure) have a high degree of connectivity with other genes, with 5 or more outward-directed edges (see Table 6.2 for descriptions). In other words, the VBSSM and EBDBN provide concurring evidence that this set of eight genes are important regulator genes in the T-cell activation network, and could potentially be avenues of interest for future research. In work by Rangel et al. (2004), the gene FYB (gene 1) is found to occupy a crucial position in

the T-cell activation network, since it is involved in the highest number of outward connections. In the consensus network between the EBDBN and VBSSM methods (Figure 6.4), although it does not figure as prominently, FYB is present and directly connected to three other genes (genes 17, 33, and 38). In addition, an interleukin receptor gene, IL-2Rγ (gene 7), is one of the most connected genes, with nine outward connections and one feedback loop with itself.

A portion of the sub-network found by Beal et al. (2005) representing the interaction between two proto-oncogenes of the Jun protein family, Jun-B (gene 54) and Jun-D (gene 11), is also found in this consensus network (see Figure 6.5). This pathway is of particular interest, as Jun-B and Jun-D are thought to negatively regulate cell growth and inhibit programmed cell death, and thus are at the center of mechanisms controlling apoptosis and proliferation. Both sub-networks seem to support this hypothesis, as Jun-B appears to regulate the apopototic gene Caspase-4 (gene 38), and at least indirectly, Caspase-8 (gene 18). Despite their differences, both sub-networks support a central pathway in which Jun-B activates Caspase-4, which in turn activates Jun-D. These results agree with the current literature, and suggest an important role for Jun-B in T-cell activation and tumor promotion (Gurzov et al., 2008).

6.2 S.O.S. DNA Repair System in *Escherichia coli*

The S.O.S. DNA repair system in the *Escherichia coli* bacterium is a well-known gene network that is responsible for repairing DNA after damage. The full network is made up of about thirty genes working at the transcriptional level, and the behavior of genes in the network in the presence of DNA damage has been well characterized (Ronen et al., 2002). Specifically, under normal conditions a master repressor called lexA represses the expression of the genes responsible for DNA repair (Figure 6.7). However, when one of the S.O.S proteins (recA) senses DNA damage by binding to single-stranded DNA, it becomes activated and provokes the autocleavage of lexA.

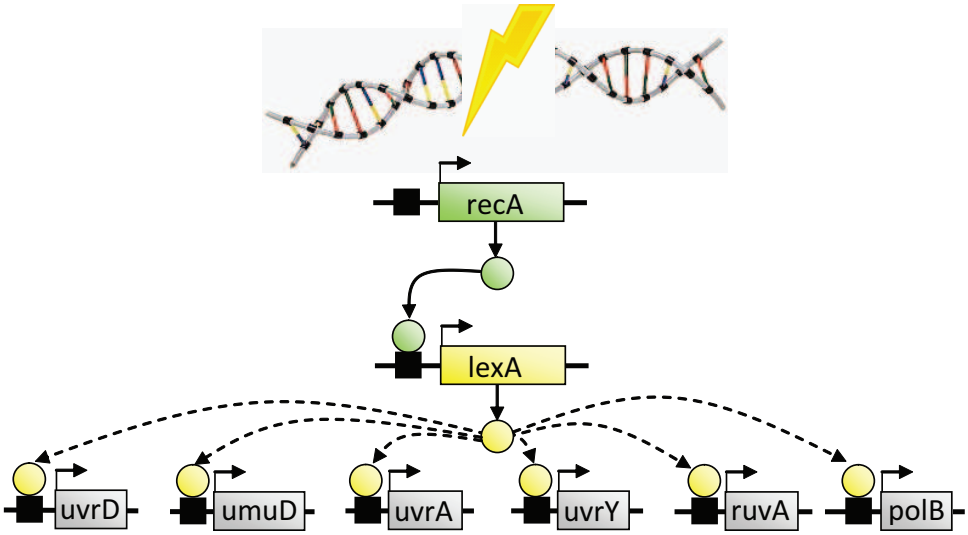


Figure 6.6. The S.O.S. DNA repair system of *E. coli*. Under normal conditions, the master repressor lexA represses the expression of the S.O.S. genes (uvrD, umuD, uvrA, uvrY, ruvA, and polB) responsible for DNA repair. When DNA damage is detected by the protein recA, it becomes activated and provokes the autocleavage of lexA. This in turn provokes the activation of the S.O.S. genes. After DNA damage is repaired, the level of recA drops, lexA reaccumulates in the cell, and the S.O.S genes return to their original state.

The subsequent drop in the levels of lexA suspends the repression of the S.O.S. genes, and these genes become activated. Once DNA damage has been repaired, the level of recA drops, which allows lexA to reaccumulate in the cell and subsequently repress the S.O.S. genes. At this point, the cells return to their original state. Although the network itself is quite small, its simple structure allows the cell to react in very sophisticated ways to conditions within the cell.

The S.O.S. DNA repair system is a good example to illustrate the utility of the ABC-Net algorithm, as it is a benchmark dataset in which specific regulatory interactions are well-characterized. The EBDBN method would not typically be applied to data from small, well-characterized networks such as the S.O.S. DNA repair system, as the gene-to-gene interactions are well understood and finding novel testable

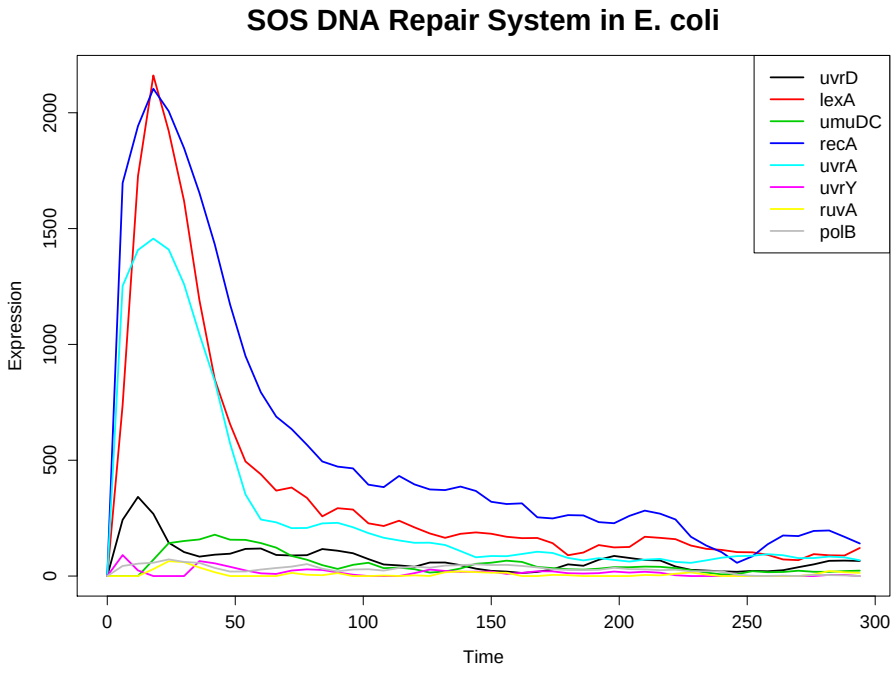


Figure 6.7. Data collected by Ronen et al. (2002) on eight genes in the S.O.S. DNA repair system in *Escherichia coli*: uvrD, lexA, umuD, recA, uvrA, uvrY, ruvA, and polB. The expression of these eight genes is measured at fifty equally spaced time points (every six minutes following ultraviolet irradiation of the cells to provoke DNA damage).

interactions is not of interest. However, applying both the EBDBN and the ABC-Net methods in this context can help illustrate the benefit of using several different methods in tandem. To this end, we draw on data collected by Ronen et al. (2002), which focused on a sub-network within the S.O.S. DNA repair system made up of eight genes: uvrD, lexA, umuD, recA, uvrA, uvrY, ruvA, and polB (Figure 6.7). Using green fluorescent protein (GFP) reporter plasmids, Ronen et al. (2002) measured the expression of the eight genes at fifty time points (every six minutes following ultraviolet irradiation of the cells to provoke DNA damage). The quantity of GFP is proportional to the quantities of the corresponding S.O.S. proteins, which are in turn proportional to the corresponding mRNA production rates (Perrin et al., 2003). As such, it is reasonable to assume that the data of Ronen et al. (2002) directly indicate

the expression levels of each of the S.O.S. genes. The data are directly available at the authors' website (`http://www.weizmann.ac.il/mcb/UriAlon`).

In addition, the study performed by Ronen et al. (2002) consisted of two different experiments for each of two different intensities of ultraviolet light (Experiments 1 and 2 at 5 Jm^{-2}, and Experiments 3 and 4 at 20 Jm^{-2}). One recent study by Charbonnier et al. (2010) found that Experiments 1 and 4 systematically led to poor results for network inference methods, although nothing should distinguish them from the other two experiments. As such, we focus the rest of our discussion on the data collected in Experiment 3, which was measured with the higher level of ultraviolet light.

Because the gene-to-gene interactions in the S.O.S. DNA repair system are well-defined and no hidden states are believed to be involved in the network, we apply the EBDBN method with a hidden state dimension of $K = 0$, where a 99.9% cutoff is used as a threshold for the z-scores of the edges. We also apply the ABC-Net method to these data. As before, we set the Gaussian proposal standard deviation in Equation (5.2) to $\sigma_\Theta = 0.5$, and we ran the algorithm for ten independent chains of length 1×10^6, with a thinning interval of 50. The VAR(1) simulator (Section 5.1) is used to generate simulated data $Y^\star$, and the prior bounds of $\pi(\Theta|G)$ are set to (-2,2). We use the Euclidean distance function, where the threshold ϵ is selected using the previously described heuristic method (Section 5.2), based on the 1% quantile of distances based on 5000 random networks. Due to the small size of the network, the maximum fan-in is constrained to 2 or less (i.e., each gene has a maximum of two regulators).

The gene-to-gene interactions identified by the EBDBN(-) method are shown in Figure 6.8, where blue and red solid edges represent "true positives" and "false postives", according to the previously described behavior of the S.O.S. network. We use these terms somewhat loosely, because even for well-understood networks such as the S.O.S. DNA repair system, the absence of a particular gene-to-gene interaction in the literature cannot indicate with absolute certainty that such a relationship is absent.

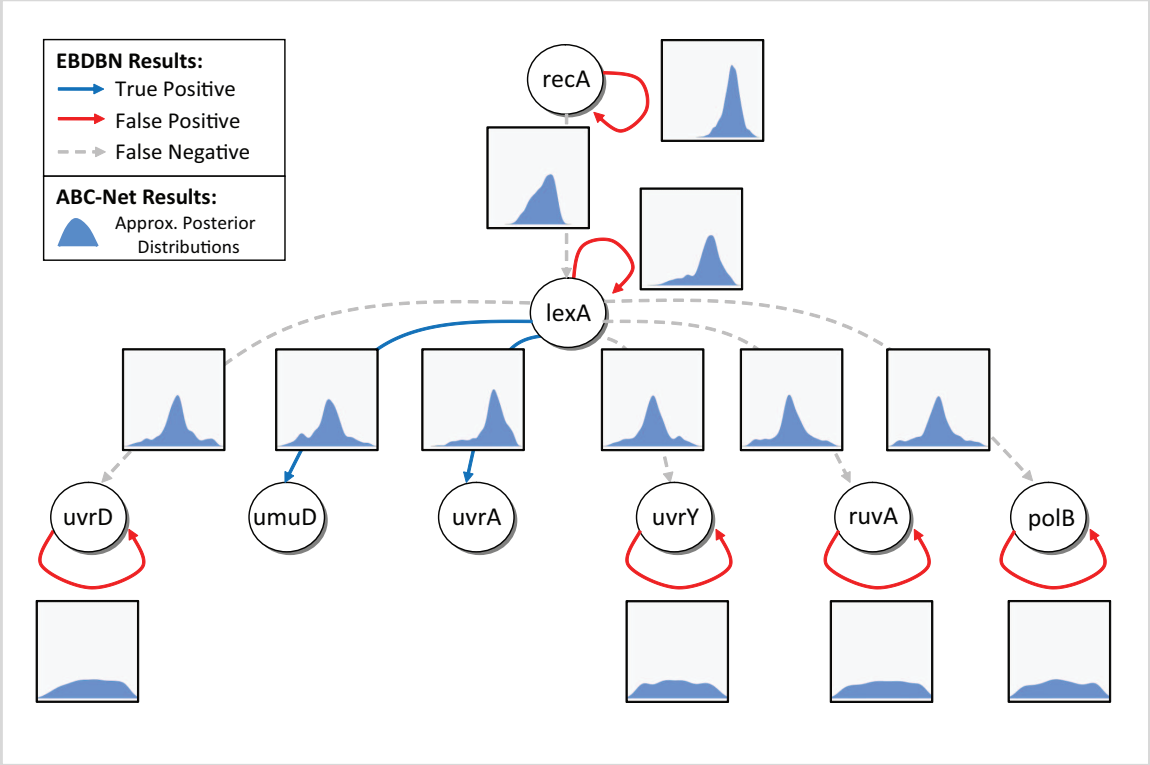


Figure 6.8. Results for the S.O.S DNA repair system for the EBDBN(-) and ABC-Net methods. Blue and red solid edges in the network represent gene-to-gene interactions identified by the EBDBN(-) method that are "true positives" and "false positives", according to the known behavior of genes in the S.O.S. network. Dotted gray lines represent gene-to-gene interactions supported by the literature that are not identified by the EBDBN(-) method. Blue-filled densities represent the marginal approximate posterior distributions found through the ABC-Net method. The feedback loops on the S.O.S. genes (uvrD, uvrY, ruvA, and polB) appear to flexible edges, while other identified edges exhibit greater rigidity.

Gray dotted lines represent gene-to-gene interactions supported by the literature that are not identified by the EBDBN(-) method. We also include the marginal approximate posterior distributions for each of these edges, as obtained by the ABC-Net method. As previously seen in the simulations of Section 5.7, these posterior distributions seem to fall into two categories: flexible edges (the feedback loops on uvrD, uvrY, ruvA, and polB) and rigid edges (the remaining edges). Edges identified by the

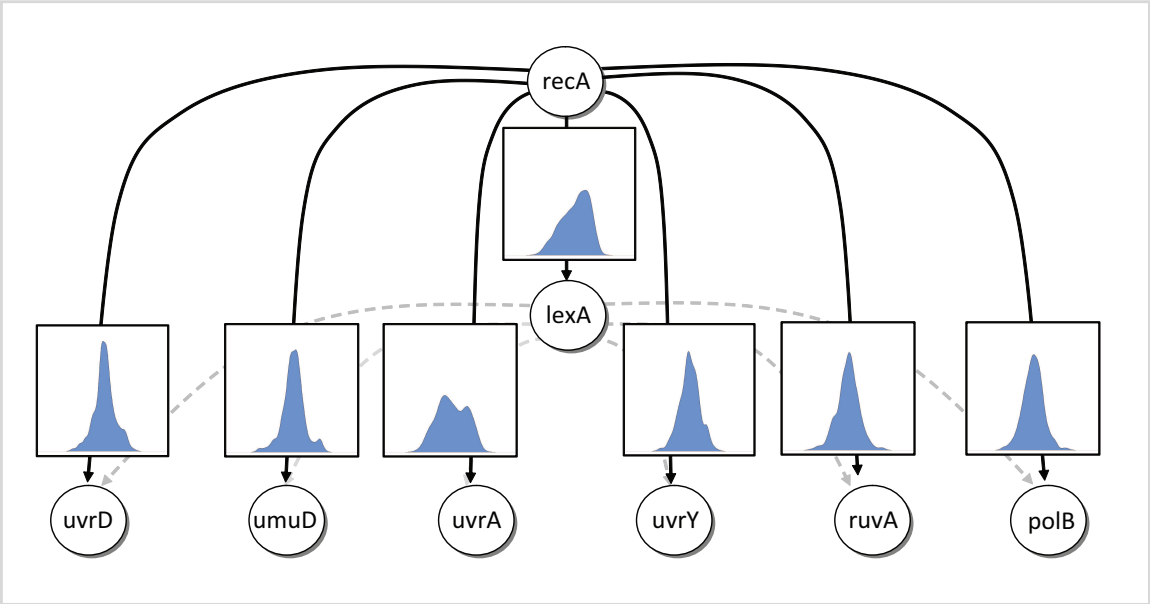


Figure 6.9. Edges exhibiting the highest rigidity in the S.O.S DNA repair system for the ABC-Net method. Dotted gray lines represent gene-to-gene interactions supported by the literature. Blue-filled densities represent the marginal approximate posterior distributions found through the ABC-Net method. The most rigid edges in the network connect the recA protein directly to the S.O.S. genes, bypassing the lexA master regulator.

EBDBN(-) with rigid approximate posterior distributions appear to be supported by substantial evidence, as those parameters are restricted to a smaller range of values in their posterior distributions. On the other hand, edges associated with flexible approximate posterior distributions may indeed represent false positives, since those parameters take on a wider range of values without negatively impacting the proximity of simulated and observed data in the ABC-Net algorithm. In this way, the distinctive results of the EBDBN and ABC-Net methods yield complementary information about specific gene-to-gene interactions, as well as the overall dynamics of a given biological system.

In addition to comparing the results of the EBDBN(-) and ABC-Net methods, we also examine the most rigid approximate posterior distributions, as identified by the

latter method (Figure 6.9). Interestingly, all of the most rigid edges in the S.O.S. DNA repair system are those directly connecting the recA protein to the other genes in the network, bypassing the lexA master regulator. This result can be explained by the one-step time delay inherent in the VAR(1) simulator of the ABC-Net method. More specifically, when DNA damage in the cell is detected by recA, the abundance of lexA decreases very rapidly and the remaining S.O.S. genes turn on almost immediately. However, time-delay models (like the VAR(1) simulator) are only able to identify gene-to-gene interactions that occur with a one-step time lag. The result of this is that in the results of the ABC-Net method, a strong link appears to occur directly between recA and the remaining genes in the network.

Finally, because the dynamics of the S.O.S. DNA repair system are constantly reacting to conditions within the cell (and thus changing over time), we also use the ABC-Net method to compare the approximate posterior distributions of gene-to-gene interactions during the first ten time points and during the last half of the experiment (Figure 6.10). The former subset makes up the portion of the data where DNA damage is at its peak, while most of the damage within the cell is repaired in the latter time interval. In examining the approximate posterior distributions, we note that for both subsets of data, flexible edges tend to stay flexible, and rigid edges tend to stay rigid. That is, in general the shape of the approximate posterior distributions is roughly the same for all edges between the two subsets of time points. There do appear to be slight differences in the approximate posterior distributions of edges emanating from the recA protein (column 4 in Figure 6.10). For these edges, densities associated with later time points appear to be slightly more peaked than for the earlier time points. In other words, at later time points the edges emanating from recA become increasingly rigid; biologically, this makes sense, since when DNA damage is repaired, the level of the recA protein must drop (and stay low) in order for lexA to reaccumulate and normal cell conditions to resume. After DNA damage has been repaired within the cell, recA has very little flexibility to interact with the other genes in the S.O.S. network.

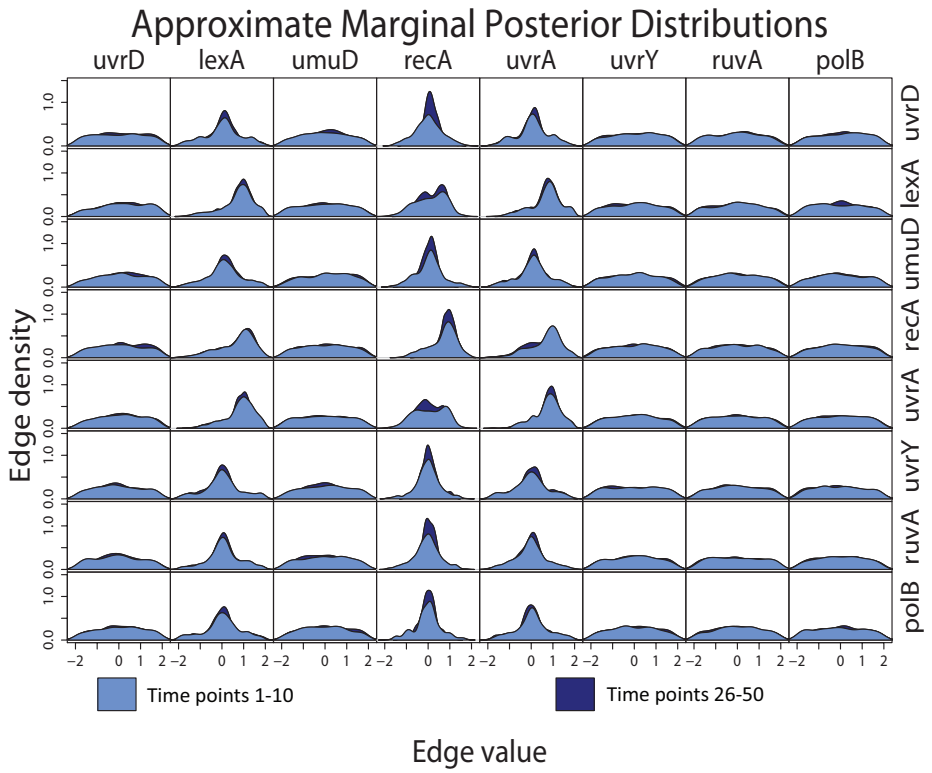


Figure 6.10. Graphical matrix of the marginal approximate posterior distributions from the ABC-Net algorithm for every edge in the S.O.S. DNA repair network, with data split into the first ten time points (light blue densities) and the last 25 time points (dark blue densities). Each element of the graphical matrix corresponds to the same element of Θ^{SOS}, i.e., the density in the second row and first column corresponds to Θ^{SOS}_{21} (uvrD→lexA). The x-axis of each plot represents the values of each parameter Θ^{SOS}_{ij}, and the y-axis represents the corresponding density. For the most part, different portions of the network share similar approximate posterior distributions, although those for recA are slightly more peaked at later time points.

6.3 Neurotrophin Signaling Pathway in Mouse

Although longitudinal studies of gene expression are not yet commonly performed using sequencing-based technologies, the decreasing cost of next generation sequencing technology suggests that such studies will likely be undertaken using next generation sequencing technologies in the near future. To date, no true longitudinal studies of gene expression based on sequencing technologies (e.g., SAGE and RNA-Seq) are widely available. However, to illustrate the flexibility and utility of the ABC-Net algorithm for longitudinal digital gene expression measures, we provide an example analysis using a series of datasets from the Mouse Atlas of Gene Expression Project (BC Cancer Agency, 2010).

The Mouse Atlas Project (BC Cancer Agency, 2010) is made up of 202 SAGE libraries from 198 tissues and various developmental stages in the common house mouse, *mus musculus*. We focus on a subset of eight SAGE libraries extracted from the whole brain, each at a different developmental stage: Theiler stage 20, Theiler stage 21, Theiler stage 23, Theiler stage 25, post-natal day 7, post-natal day 27, post-natal day 35, and 12 weeks post-birth. This particular collection of SAGE libraries was chosen as it represented the largest number of time points for a given tissue within the Mouse Atlas Project. Note that these data do not represent a true time series, as different mice were used at each time point, and the eight libraries are from eight different experiments. Nonetheless, this series of experiments provides a useful illustration of the extension of the ABC-Net algorithm to count data, and similar techniques could be employed for true longitudinal sequence-based studies when they become available.

The eight SAGE libraries under consideration all used wild-type, male mice of the same strain (C57BL/6J), based on the same SAGE protocol. Each SAGE library contained between 30,967 and 42,220 unique sequence tags. Data for the eight SAGE libraries were downloaded from the BC Cancer Agency (2010) website, and the tag counts at quality 0.95 cut-off were aligned to the annotated transcripts from the

NCBI Reference Sequence Project (RefSeq) database (National Library of Medicine and National Center for Biotechnology Information, 2010), found at `ftp://ftp.ncbi.nih.gov/genomes/M_musculus/RNA`. Alignment for each of the eight SAGE libraries was done using the SOAPaligner/soap2 algorithm with default settings in the Short Oligonucleotide Analysis Project (SOAP) of Li et al. (2008). Counts for each of the sequence tags were subsequently summed to form a digital expression measure for each gene. The number of unique genes in each library ranged from 12,021 (Theiler stage 21) to 14,449 (post-natal day 35) genes.

After summarizing counts at the gene level for each of the eight SAGE libraries, the data were further filtered in the following manner. Using the Kyoto Encylopedia of Genes and Genomes (KEGG) database (KEGG PATHWAY Database, 2010), which consists of a set of manually-drawn pathway graphs representing organism-specific networks of molecular interactions (Kanehisa et al., 2010, 2006; Kanehisa and Goto, 2000), we identified a smaller subset of genes for the task of network inference. In particular, we focused our attention on a gene network in the nervous system of the mouse involving the differentiation, development, and survival of neural cells, known as the neurotrophin signalling pathway (network identifier: mmu04722, see Figure 6.11). According to the KEGG PATHWAY Database (2010), the 144 genes within the neurotrophin pathway play an important role in neural development and higher-order activities like learning and memory. Because the eight SAGE libraries under consideration were extracted from the whole brain, this particular network is of interest as it plays an important role in the nervous system.

Only 16 of the 144 genes involved in the neurotrophin signalling pathway were mapped to by at least one sequence tag in all eight of the SAGE libraries. For this reason, we further focused on a sub-network of these 16 genes within the neurotrophin signalling pathway (see Table 6.3). We subsequently applied the ABC-Net method to these "longitudinal" counts using the extension described in Section 5.6. As before, we set the Gaussian proposal standard deviation in Equation (5.2) to $\sigma_\Theta = 0.5$ and the prior bounds of $\pi(\Theta|G)$ to (-2,2). We ran the ABC-Net algorithm for ten independent

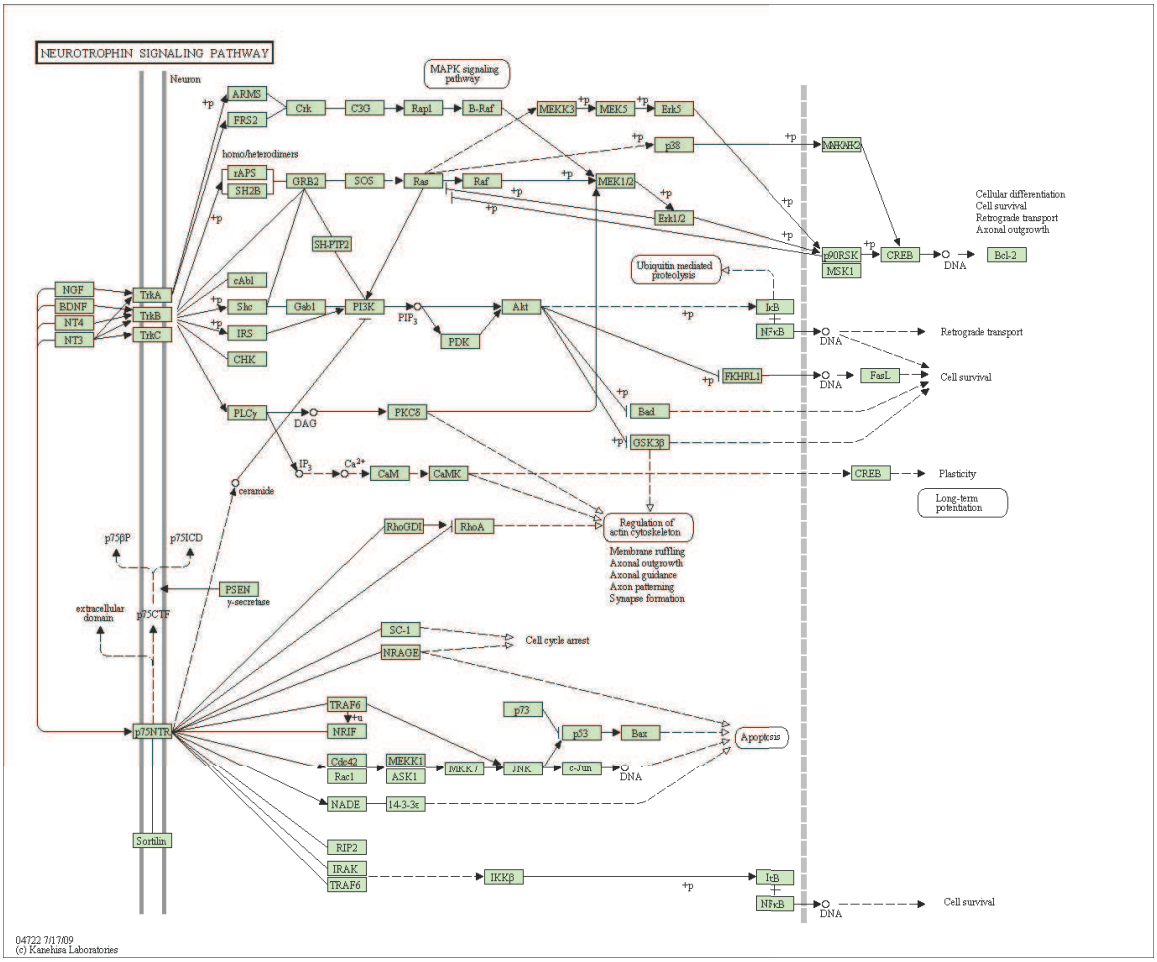


Figure 6.11. The neurotrophin signalling pathway, as described in the KEGG PATHWAY Database (2010). Boxes indicate gene products (e.g., proteins and RNA), small circles represent other molecules, arrows represent molecular interactions or relations, dotted lines indicate indirect effects, and solid lines represent bindings or associations. Phosphorylation is represented by +p, and ubiquitination by u+. For additional information, see the KEGG PATHWAY Database (2010).

chains of length 1×10^6, using a thinning interval of 50, a maximum fan-in of 5 (i.e., each gene has a maximum of 5 regulators), and the Euclidean distance function. At each iteration five independent datasets $Y^{\star(1)}, \ldots, Y^{\star(5)}$ were simulated for a proposed network, and the distances were averaged to yield a single distance for each $\Theta^\star$. The threshold ϵ was selected using the heuristic method of Section 5.2, based on the 1%

Table 6.3
Names and descriptions of a sub-network of sixteen genes in the neurotrophin signalling pathway (KEGG PATHWAY Database, 2010).

Gene	Description
Sos1	Son of sevenless homolog 1
Nras	Neuroblastoma ras oncogene
Raf1	V-raf-leukemia viral oncogene 1
Mapk3	Mitogen-activated protein kinase 3
Atf4	Activating transcription factor 4
Kidins220	Kinase D-interacting substrate 220
Calm1	Calmodulin 1
Calm3	Calmodulin 3
Rac1	RAS-related C3 botulinum substrate 1
Map3k1	Mitogen-activated protein kinase kinase kinase 1
Mapk10	Mitogen-activated protein kinase 10
Bax	BCL2-associated X protein
Prdm4	PR domain containing 4
Maged1	Melanoma antigen, family D, 1
Ywhaq	Tyrosine 3-monooxygenase/tryptophan 5-monooxygenase activation protein, theta polypeptide
Sort1	Sortilin 1

quantile of averaged distances of five simulated datasets for each of 5000 random networks. Note that the EBDBN algorithm could not be applied to this set of data, as gene expression is measured by counts and not continuous values.

The results of the ABC-Net algorithm can be seen by examining the marginal approximate posterior distributions shown in Figure 6.12. It can be seen that the majority of the edges in the network display flexible (diffuse) posterior distributions.

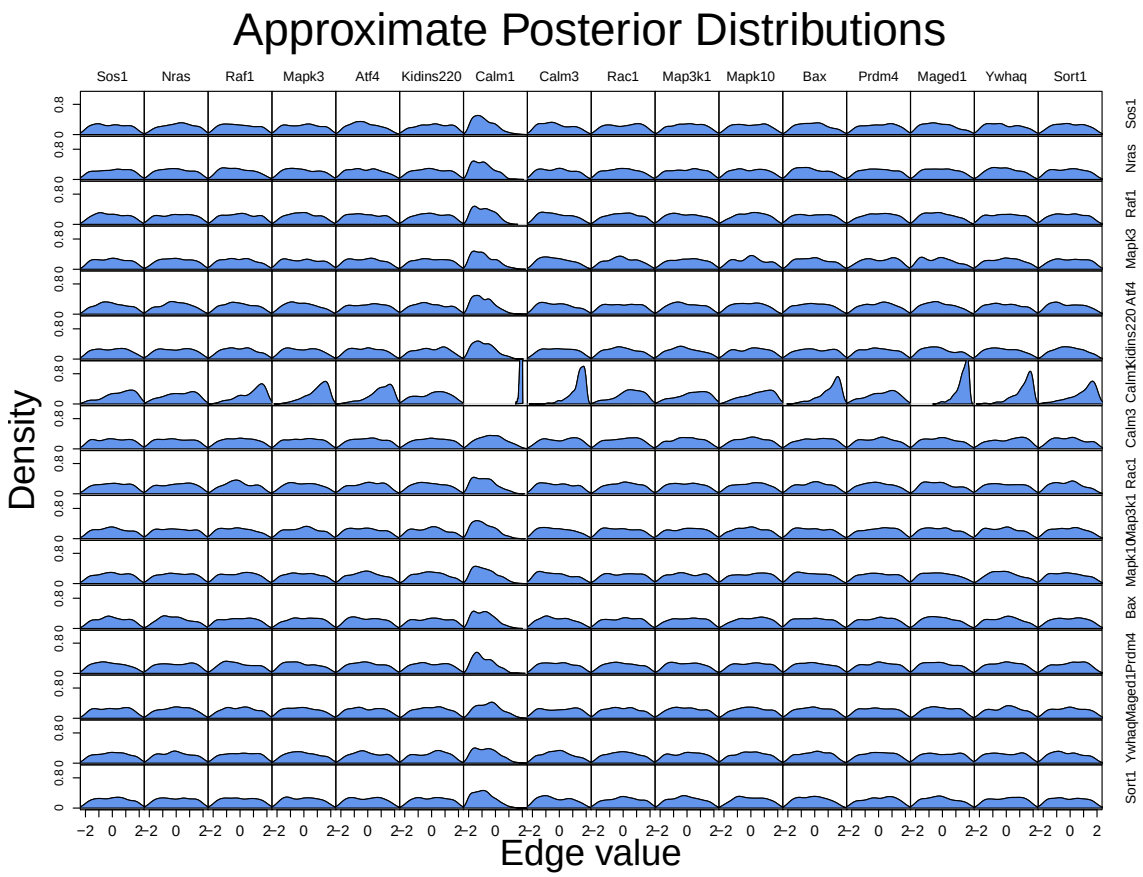


Figure 6.12. Graphical matrix of the marginal approximate posterior distributions for every potential edge in the neurotrophin signalling pathway. Each element of the graphical matrix corresponds to the same element of the adjacency matrix for the neurotrophin pathway, i.e., the density in the second row and first column corresponds to Sos1→Nras. The x-axis of each plot represents the values of each parameter, and the y-axis represents the corresponding density.

However, it appears that one gene, Calm1, plays a central role in the network, both in terms of inward and outward pointing edges. In particular, Calm3, Bax, Maged1, and Ywhaq all appear to be strong activators of Calm1 (with fairly rigid, predominantly positive approximate posterior distributions), while Calm1 itself appears to be a moderately strong repressor of the other genes in the sub-network. One of the most prominent features of these results is the very strong, positive feedback loop in

which Calm1 participates (Calm1→Calm1). This feedback loop displays a very rigid edge that is concentrated at the upper bound of the prior distribution support.

The discrepancies between these results and the known structure of the neurotrophin signalling pathway (Figure 6.11) may be explained by several factors. First, although the data in this example were chosen to be as similar as possible (e.g., mouse strain and SAGE protocol), they are in fact heterogeneous. Because the eight experiments were conducted individually on different mice at different developmental stages, the data are not truly longitudinal. In addition, the time points used in the analysis are unevenly spaced, and represent a wide spectrum of developmental stages, where some are collected at pre-natal stages (in embryos) and others after birth. This also affects the assumption of time-invariant gene-to-gene interactions, since it is unlikely that the behavior of the neurotrophin network remains constant through such a wide spectrum of unevenly spaced developmental stages. Finally, one additional consideration concerns the differences in magnitudes among the sixteen genes considered (Table 6.3). Calm1 displays both the largest magnitude and range of digital expression measures over the eight time points (varying between a count of 131 to 633 over the eight time points). For the most part, the other genes have counts of less than 100 over all time points, with 13 of the 16 genes displaying counts of 20 or less across all time points. This large heterogeneity may explain in part the strong influence of Calm1 on the remaining genes in the network (Figure 6.11), and may suggest that some sort of normalization is required.

7. SUMMARY AND FUTURE WORK

Reverse engineering the structure of gene regulatory networks from longitudinal expression data is an intrinsically difficult task, given the complexity of network architecture, the large number of potential gene-to-gene interactions in typical networks, and the small number of replicates and time points available in real data. In this dissertation, two different approaches are proposed that are based in approximate Bayesian methodology. Both make use of directed, graphical models of stochastic processes, known as dynamic Bayesian networks, to infer information about underlying gene regulatory networks from time-series gene expression data. In addition, both approaches explicitly model the joint behavior of a set of genes over time, thus maintaining the correlation structure of gene expression between adjacent time points, rather than examining time points independently. Although the two approaches are not directly comparable, they are complementary to one another and illustrate the need for a variety of network inference methods adapted for different contexts.

The first proposed approach, known as the Empirical Bayes Dynamic Bayesian Network (EBDBN) method (Rau et al., 2010), is based on an empirical Bayes estimation procedure using a linear Gaussian state space model. This approach was motivated by that of Beal et al. (2005), where variational Bayesian learning was used to estimate model parameters. The novelty of the EBDBN method lies in its efficient empirical Bayes estimation of hyperparameters within a hierarchial Bayesian framework and the resulting improvement in computational time. Model selection is performed using the singular value decomposition of the block-Hankel matrix, and hidden states and hyperparameters are iteratively estimated until convergence is attained. Z-scores based on the posterior distributions of network parameters are used to identify the presence or absence of edges in moderately sized networks. Due to its restrictive dis-

tributional assumptions, the EBDBN method is best suited to exploratory analyses of gene regulatory networks where little *a priori* biological information is known.

In the second approach, known as the ABC-MCMC for Networks (ABC-Net) approach, we apply a simulation-based Bayesian method to conduct a detailed analysis of small, well-characterized pathways under fewer model assumptions. Using approximate Bayesian computation and a first-order vector autoregressive model, the ABC-Net approach enables Bayesian inference for complex, high-dimensional networks for which the likelihood is difficult to calculate. By sampling from the approximate posterior distributions of parameters involved in gene regulatory networks, this method yields a wealth of information about the structure and inferability of complicated biological systems, particularly with respect to the flexibility and rigidity of network edges. In addition, the ABC-Net method has the flexibility to incorporate *a priori* biological knowledge into the prior distribution structure, as well as to incorporate longitudinal counts of gene expression (e.g., SAGE or RNA-Seq data). For the time being, the computing time required for the ABC-Net limits its application to small networks. However, as computing power continues to improve in terms of speed and memory, it is anticipated that the method will progressively be able to handle larger networks.

Throughout this work, we made use of extensive simulation studies to evaluate the performance of our two proposed approaches under different settings and in comparison to other published network inference methods. However, designing and implementing simulated data in this context is not always straightforward. We suggest the use of both "model-based" and "data-based' simulations when comparing the results of several methods. Model-based simulations, though an over-simplified version of the complexities of gene regulatory networks, offer greater flexibility in the choice of network parameters (e.g., number of genes, number of hidden states, percentage of edges present, amount of noise, etc.). More realistic data-based simulations using ordinary differential equations may better represent the complicated dynamics of gene regulatory networks. However, in general they are made up of much smaller

datasets than would typically be used for network inference. The development of a set of realistic, time-course benchmark datasets for comparisons among methods is a necessary addition to the growing collection of network inference techniques.

Both approaches proposed in this work do share some limitations. In particular, both techniques make the assumption of linear, time-invariant relationships among genes in a network. In addition, to avoid an explosion in model complexity, both also deal solely with first-order dynamics (i.e., the expression of genes at time t depends only on those at time $t-1$). Although these simplifications can be a good approximation to the general nature of complicated biological systems, more realistic models of gene regulatory interactions would undoubtedly capture complex relationships, such as nonlinear, time-variant, or higher-order interactions, more effectively.

7.1 Future Work

One of the major stumbling blocks for reverse engineering gene regulatory networks is highlighted by the fact that different methods, even those that are very similar to one another, often yield very different results for the same data. As one example, in the analysis of the T-cell activation data in Section 6.1, in the results of the EBDBN and VBSSM methods (which resemble each other quite closely), over 60% of the edges identified by each one are not identified by the other. In this work, we introduced the concept of a "consensus network", that is, one in which several different network inference methods are in agreement on the significance of a particular edge. Although somewhat unorthodox from a frequentist statistical point of view, this type of compromise (also referred to as "model averaging") may be more meaningful than applying a single method for network inference. However, more sophisticated statistical methods for combining results from different methods are needed.

Further work is also required to fully examine the components of the ABC-Net algorithm. In this work, we propose new network structures based on the application of one of three basic moves (add, delete, or reverse an edge). However, more

elaborate proposal schemes are possible, and may more effectively explore the approximate posterior distributions. Alternative data simulators must be proposed and explored, in addition to appropriate techniques to identify the optimal data simulator to be used for real data. Further work is also required on the interpretation of the marginal approximate posterior distributions. For instance, rather than relying solely on qualitative examinations of these posterior distributions, a hierarchical prior could be defined on the network adjacency G, and a local Bayes factor could be used to obtain a numerical assessment of the evidence for a particular gene-to-gene interaction. Finally, due to its computational burden, the ABC-Net algorithm would benefit from optimized and parallelized programs to best exploit computational resources.

The ability of the ABC-Net method to analyze time-series digital gene expression measures (e.g., RNA-Seq data) is particularly promising, as illustrated by the analysis in Section 6.3. In that example, we used an autoregressive simulator for Poisson distribution rates, and a Euclidean distance function to compare observed and simulated data. Additional work is required to determine whether other techniques for simulating time-series count data are better suited to gene expression data. In addition, because the data are non-continuous, it is likely that other distance functions are better adapted to compare simulated and observed data. Keeping these considerations in mind, simulation studies are required to evaluate the performance of this extension of the ABC-Net algorithm under a variety of simulation settings. By adapting the ABC-Net approach to forthcoming longitudinal sequence-based measures of gene expression, it will be well-poised at the forefront of methods to reverse engineer gene regulatory networks in the coming years.

7.2 Conclusions

This research demonstrates the applicability of an approximate Bayesian framework to the task of reverse engineering gene regulatory networks. Specifically, there are three major statistical and bioinformatic contributions from this work. First,

a novel empirical Bayes estimation procedure for linear state space models, which enables efficient identification of gene-to-gene interactions in moderately sized gene regulatory networks, was proposed. Second, an `R` package to implement this approach was made publicly available. Third, a novel application of approximate Bayesian computation methods to the inference of gene regulatory networks was proposed. These varied contributions highlight the advantage of developing a variety of network inference methods to provide greater insight into the complicated interactions occurring among genes in real biological systems.

LIST OF REFERENCES

LIST OF REFERENCES

454 Life Sciences (2010). `http://www.454.com`.

Affymetrix (2010). `http://www.affymetrix.com`.

Agilent Technologies (2010). `http://www.agilent.com`.

Agilent Technologies Newsroom (2010). `http://www.agilent.com/about/newsroom/lsca/imagelibrary/index_2004.html`.

Ahmed, A. and E. P. Xing (2009). Recovering time-varying networks of dependencies in social and biological studies. *PNAS 106*(29), 11878–11883.

Akaike, H. (1969). Fitting autoregressive models for prediction. *Annals of the Institute of Statistical Mathematics 21*, 243–247.

Alon, U. (2007). Network motifs: theory and experimental approaches. *Nature Genetics Reviews 8*, 450–461.

Aoki, M. and A. Havenner (1991). State space modeling of multiple time series. *Econometric Reviews 10*(1), 1–59.

Applied Biosystems (2010). `http://www3.appliedbiosystems.com`.

Arbeitman, M. N., E. E. M. Furlong, F. Imam, E. Johnson, B. H. Null, B. S. Baker, M. A. Krasnow, M. P. Scott, R. W. Davis, and K. P. White (2002). Gene expression during the life cycle of *Drosophila melanogaster*. *Science 297*, 2270–2275.

Auer, P. L. (2010). *Statistical Design and Analysis of Next-Generation Sequencing Data*. Ph.D. dissertation, Purdue University, West Lafayette, IN USA.

Auer, P. L. and R. W. Doerge (2010). Statistical design and analysis of RNA-Seq data. *Genetics 185*, 1–12.

Baggerly, K. A., L. Deng, J. S. Morris, and C. M. Aldaz (2004). Overdispersed logistic regression for SAGE: modelling multiple groups and covariates. *BMC Bioinformatics 5*(144).

Bar-Joseph, Z., G. Gerber, I. Simon, D. K. Gifford, and T. S. Jaakkola (2003). Comparing the continuous representation of time-series expression profiles to identify differentially expressed genes. *PNAS 100*(18), 10146–10151.

Basso, K., A. A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera, and A. Califano (2005). Reverse engineering of regulatory networks in human B cells. *Nature Genetics 37*(4), 382–390.

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions 53*, 370–418.

BC Cancer Agency (2010). Mouse Atlas of Gene Expression Project. `http://www.mouseatlas.org`.

Beal, M. J., F. Falciani, Z. Ghahramani, C. Rangel, and D. L. Wild (2005). A Bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics 21*(3), 349–356.

Beal, M. J., J. Li, Z. Ghahramani, and D. L. Wild (2005). *Introduction to Systems Biology*, Chapter "Reconstructing transcriptional networks using gene expression profiling and Bayesian state space models". Humana Press (Springer).

Beaumont, M. A., J.-M. Cornuet, J.-M. Marin, and C. P. Robert (2009). Adaptivity for ABC algorithms: the ABC-PMC. *Biometrika 96*(4), 983–990.

Beaumont, M. A., W. Zhang, and D. J. Balding (2002). Approximate Bayesian computation in population genetics. *Genetics 162*, 2025–2035.

Bilmes, J. A. (1997). A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical Report ICSI-TR-97-021, University of Berkeley.

Blum, M. G. B. and O. François (2010). Non-linear regression models for Approximate Bayesian Computation. *Statistics and Computing 20*(1), 63–73.

Bolstad, B. M., R. A. Irizarry, M. Astrand, and T. P. Speed (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics 19*(2), 185–193.

Bortot, P., S. G. Coles, and S. A. Sisson (2007). Inference for stereological extremes. *Journal of the American Statistical Association March*, 84–92.

Brandman, O. and T. Meyer (2008). Feedback loops shape cellular signals in space and time. *Science 322*, 390–395.

Bremer, M. and R. W. Doerge (2009). The KM-algorithm identifies regulated genes in time series expression data. *Advances in Bioinformatics 2009*, 284251.

Bremer, M. M. (2006). *Identifying Regulated Genes Through the Correlation Structure of Time Dependent Microarray Data*. Ph.D. dissertation, Purdue University, West Lafayette, IN USA.

Butte, A. J. and I. S. Kohane (2000). Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pacific Symposium on Biocomputing*, 418–429.

Cao, J. and H. Zhao (2008). Estimating dynamic models for gene regulation networks. *Bioinformatics 24*(14), 1619–1624.

Cappé, O., A. Guillin, and C. Robert (2004). Population Monte Carlo. *Journal of Graphical and Computational Statistics 13*, 907–929.

Carey, V. J., J. Gentry, E. Whalen, and R. Gentleman (2005). Network structures and algorithms in Bioconductor. *Bioinformatics 21*(1), 135–136.

Carlin, B. P. and T. A. Louis (2000). *Bayes and Empirical Bayes Methods for Data Analysis* (2nd ed.). Chapman and Hall/CRC.

Casella, G. (1985). An introduction to empirical Bayes data analysis. *The American Statistician 39*(2), 83–87.

Charbonnier, C., J. Chiquet, and C. Ambroise (2010). Weighted-LASSO for structured network inference from time course data. *Statistical Applications in Genetics and Molecular Biology 9*(15), 1–29.

Chiquet, J., A. Smith, G. Grasseau, C. Matias, and C. Ambroise (2009). SIMoNe: Statistical Inference for MOdular NEtworks. *Bioinformatics 25*(3), 417–418.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association 74*, 829–836.

Cloonan, N., A. R. R. Forrest, G. Kolle, B. B. A. Bardiner, F. G. J., M. K. Brown, D. F. Taylor, A. L. Steptoe, S. Wani, G. Bethel, A. J. Robertson, A. C. Perkins, S. J. Bruce, C. C. Lee, S. S. Ranade, H. E. Peckham, J. M. Manning, K. J. McKernan, and S. M. Grimmond (2008). Stem cell transcriptome profiling via massive-scaled mRNA sequencing. *Nature Methods 5*, 613–619.

Cowles, M. K. and B. P. Carlin (1994). Markov chain Monte Carlo convergence diagnostics: a comparative review. Technical Report 94-008, Division of Biostatistics, School of Public Health, University of Minnesota.

Crick, F. H. C. (1970). Central dogma of molecular biology. *Nature 227*, 561–563.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodological) 39*(1), 1–38.

D'haeseleer, P., S. Liang, and R. Somogyi (2000). Genetic network inference: from co-expression clustering to reverse engineering. *Bioinformatics 16*(8), 707–726.

Eckart, C. and G. Young (1936). The approximation of one matrix by another of lower rank. *Psychometrika 1*(3), 211–218.

Efron, B. (2003). Robbins, empirical Bayes and microarrays. *Annals of Statistics 31*(2), 366–378.

Efron, B. (2005). Local false discovery rates. Technical Report 2005-20B/234, Department of Statistics, Stanford University.

Efron, B. and C. Morris (1972). Limiting the risk of Bayes and empirical Bayes estimators – Part II: The empirical Bayes case. *Journal of the American Statistical Association 67*(337), 130–139.

Efron, B. and C. Morris (1973). Stein's estimation rule and its competitors – an empirical Bayes approach. *Journal of the American Statistical Association 68*(341), 117–130.

Efron, B. and C. Morris (1975). Data analysis using Stein's estimator and its generalizations. *Journal of the American Statistical Association 70*(350), 311–319.

Efron, B., R. Tibshirani, J. D. Storey, and V. Tusher (2001). Empirical Bayes analysis of a microarray experiment. *Journal of the American Statistical Association 96*(456), 1151–1160.

Eisen, M. B., P. T. Spellman, P. O. Brown, and D. Botstein (1998). Cluster analysis and display of genome-wide expression patterns. *PNAS 95*(25), 14863–14868.

EMBL Nucleotide Sequence Database (2010). `http://www.ebi.ac.uk/embl`.

Enders, W. (2004). *Applied Econometric Time Series Analysis*. Wiley.

Friedman, N. (2000). Using Bayesian networks to analyze expression data. *Journal of Computational Biology 7*(3/4), 601–620.

Friedman, N. (2004). Inferring cellular networks using probabilistic graphical models. *Science 303*(799), 799–805.

Fuhrman, S., M. J. Cunningham, X. Wen, G. Zweiger, J. J. Seilhamer, and R. Somogyi (2000). The application of Shannon entropy in the identification of putative drug targets. *Biosystems 55*(1-3), 5–14.

Fujita, A., J. R. Sato, H. M. Garay-Malpartida, R. Yamaguchi, S. Miyano, M. C. Sogayar, and C. E. Ferreira (2007). Modeling gene expression regulatory networks with the sparse vector autoregressive model. *BMC Systems Biology 1*(39).

Gardner, T. S., D. di Bernardo, D. Lorenz, and J. J. Collins (2003). Inferring genetic networks and identifying compound mode of action via expression profiling. *Science 301*, 102–105.

Gelman, A., J. Carlin, H. Stern, and D. Rubin (2004). *Bayesian Data Analysis* (2nd ed.). Chapman and Hall.

Gelman, A. and D. B. Rubin (1992). Inference from iterative simulation using multiple sequences. *Statistical Science 7*, 457–511.

Geyer, C. J. (1992). Practical Markov chain Monte Carlo. *Statistical Science 7*, 473–511.

Giessen Research Center in Infectious Diseases (2010). `http://www.uniklinikum-giessen.de/grid/micarray.html`.

Gilks, W. R., S. Richardson, and D. J. Spiegelhalter (Eds.) (1996). *Markov Chain Monte Carlo in Practice: Interdisciplinary Statistics*. Chapman and Hall/CRC.

Grelaud, A. (2009). *Méthodes sans vraisemblance appliquées à l'étude de la sélection naturelle et à la prédiction de structure tridimensionnelle des protéines*. Ph. D. thesis, Université Paris Dauphine, Paris, France.

Griffiths, A., J. Miller, D. Suzuki, R. Lewontin, and W. Gelbart (2008). *Introduction to Genetic Analysis*. W. H. Freeman and Co.

Gurzov, E. N., L. Bakiri, J. M. Alfaro, E. F. Wagner, and M. Izquierdo (2008). Targeting c-Jun and JunB proteins as potential anticancer cell therapy. *Oncogene 27*, 641–352.

Hartemink, A. J., D. K. Gifford, T. S. Jaakkola, and R. A. Young (2001). Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. *Pacific Symposium on Biocomputing 6*, 422–433.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika 57*(1), 97–109.

Huang, D. and W. Pan (2006). Incorporating biological knowledge into distance-based clustering analysis of microarray gene expression data. *Bioinformatics 22*(10), 1259–1268.

Husmeier, D. (2003). Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. *Bioinformatics 19*(17), 2271–2282.

Husmeier, D., R. Dybowski, and S. Roberts (Eds.) (2005). *Probabilistic Modeling in Bioinformatics and Medical Informatics*. Springer.

Illumina (2010). `http://www.illumina.com`.

Imoto, S., T. Goto, and S. Miyano (2002). Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. *Pacific Symposium on Biocomputing 7*, 175–186.

JIC Genome Laboratory (2010). `http://revgenukorders.jic.ac.uk/services/microarrays/affymetrix-genechip/genechip-system.html`.

Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer.

Kaiser, H. (1960). The application of electronic computers to factor analysis. *Educational and Psychological Measurement 20*(1), 141–151.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME - Journal of Basic Engineering 82*, 35–45.

Kanehisa, M. and S. Goto (2000). KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucelic Acids Research 28*, 27–30.

Kanehisa, M., S. Goto, M. Furumichi, M. Tanabe, and M. Hirakawa (2010). KEGG for representation and analysis of molecular networks involving diseases and drugs. *Nucleic Acids Research 38*(9), D355–D360.

Kanehisa, M., S. Goto, M. Hattori, K. F. Aoki-Kinoshita, M. Itoh, S. Kawashima, T. Katayama, M. Araki, and M. Hirakawa (2006). From genomics to chemical genomics: new developments in KEGG. *Nucelic Acids Research 34*, D354–357.

KEGG PATHWAY Database (2010). `http://www.genome.jp/kegg/pathway.html`.

Keller, M. P., Y. Choi, P. Wang, D. B. Davis, M. E. Rabaglia, A. T. Oler, D. S. Stapleton, C. Argmann, K. L. Schueler, S. Edwards, H. A. Steinberg, E. C. Neto, R. Kleinhanz, S. Turner, M. K. Hellerstein, E. E. Schadt, B. S. Yandell, C. Kendziorski, and A. D. Attie (2008). A gene expression network model of type 2 diabetes links cell cycle regulation in islets with diabetes susceptibility. *Genome Research 18*, 706–716.

Kerr, M. K. (2003). Design considerations for efficient and effective microarray studies. *Biometrics 59*, 822–828.

Kerr, M. K. and G. A. Churchill (2001). Statistical design for gene expression microarray data. *Genetical Research 77*, 123–128.

Leclerc, R. D. (2008). Survival of the sparsest: robust gene networks are parsimonious. *Molecular Systems Biology 4*(213).

Lee, I., S. V. Date, A. T. Adai, and E. M. Marcotte (2004). A probabilistic functional network of yeast genes. *Science 306*, 1555–1558.

Leiden Genome Technology Center (2010). `http://www.lgtc.nl/equipment.php`.

Leuenberger, C. and D. Wegmann (2009). Bayesian computation and model selection without likelihoods. *Genetics 183*, 1–10.

Li, R., Y. Li, K. Kristiansen, and J. Wang (2008). SOAP: short oligonucleotide alignment program. *Bioinformatics 24*(5), 713–714.

Liang, Y. and A. Kelemen (2007). Bayesian state space models for inferring and predicting temporal gene expression profiles. *Biometrical Journal 49*(6), 801–814.

Lipschutz, R. J., S. P. A. Fodor, T. R. Gingeras, and D. J. Lockhart (1999). High density synthetic oligonucleotide arrays. *Nature Genetics 21*, 20–24.

Lockhart, D. J. and E. A. Winzeler (2000). Genomics, gene expression, and DNA arrays. *Nature 405*, 827–836.

Lönnstedt, I. and T. Speed (2002). Replicated microarray data. *Statistica Sinica 12*, 31–46.

Lund, R. and B. Li (2009). Revisiting climate region definitions via clustering. *American Meteorlogical Society 22*, 1787–1800.

Luo, W., K. D. Hankenson, and P. J. Woolf (2008). Learning transcriptional regulatory networks from high throughput gene expression data using continous three-way mutual information. *BMC Bioinformatics 9*.

Man, M. X., X. Wang, and Y. Wang (2000). POWER_SAGE: comparing statistical tests for SAGE experiments. *Bioinformatics 16*, 953–959.

Mardia, K. V., J. T. Kent, and J. M. Bibby (1980). *Multivariate Analysis*. Academic Press.

Mardis, E. R. (2008). Next-generation DNA sequencing methods. *Annual Reviews of Genomics and Human Genetics 9*, 387–402.

Marioni, J. C., C. E. Mason, S. M. Mane, M. Stephens, and Y. Gilad (2008). RNA-Seq: An assessment of technical reproducibility and comparison with gene expression arrays. *Genome Research 18*(9), 1509–1517.

Marjoram, P., J. Molitor, V. Plagnol, and S. Tavaré (2003). Markov chain Monte Carlo without likelihoods. *PNAS 100*(26), 15324–15328.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics 21*(6), 1087–1092.

Meyer, P. E., F. Lafitte, and G. Bontempi (2008). minet: a R/Bioconductor package for inferring large transcriptional networks using mutual information. *BMC Bioinformatics 9*(461).

Microarray Core Facility (2010). http://core.iddrc.org/molecular-genetics/?page_id=7.

Morozova, O., M. Hirst, and M. A. Marra (2009). Applications of new sequencing technologies for transcriptome analysis. *Annual Review of Genomics and Human Genetics 10*, 135–151.

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

Mullis, K. B., F. Ferre, and R. A. Gibbs (1994). *The polymerase chain reaction* (1st ed.). Birkhäuser.

Murphy, K. and S. Mian (1999). Modelling gene expression data using dynamic Bayesian networks. Technical report, Computer Science Division, University of California, Berkeley, CA.

National Center for Biotechnology Information Entrez Genome (2010). http://www.ncbi.nlm.nih.gov.

National Human Genome Research Institute (2010). http://www.genome.gov/glossary/index.cfm?id=53.

National Library of Medicine and National Center for Biotechnology Information (2010). The NCBI handbook: Chapter 18, The Reference Sequence (RefSeq) Project. http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Books.

Niida, A., A. D. Smith, S. Imoto, S. Tsutsumi, H. Aburatani, M. Q. Zhang, and T. Akiyama (2008). Integrative bioinformatics analysis of transcriptional regulatory programs in breast cancer cells. *BMC Bioinformatics 9*(404).

NimbleGen (2010). http://www.nimblegen.com.

Ong, I. M., J. D. Glasner, and D. Page (2002). Modelling regulatory pathways in *E. coli* from time series expression profiles. *Bioinformatics 18*, S241–S248.

Opgen-Rhein, R. and K. Strimmer (2006). Inferring gene dependency networks from genomic longitudinal data: a functional data approach. *Revstat 4*(1), 53–65.

Opgen-Rhein, R. and K. Strimmer (2007). Learning causal networks from systems biology time course data: an effective model selection procedure for the vector autoregressive process. *BMC Bioinformatics 8(Suppl 2)*.

Pan, W. (2006). Incorporating gene functions as priors in model-based clustering of microarray gene expression data. *Bioinformatics 22*(7), 795–801.

Park, T., S.-G. Yi, S. Lee, S. Y. Lee, D.-H. Yoo, J. I. Ahn, and Y.-S. Lee (2003). Statistical tests for identifying differentially expressed genes in time-course microarray experiments. *Bioinformatics 19*(6), 694–703.

Parzen, E. (1962). On estimation of a probability density function and mode. *Annals of Mathematical Statistics 33*(3), 1065–1076.

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

Perrin, B.-E., L. Ralaivola, A. Mazurie, S. Bottani, J. Mallet, and F. d'Alché Buc (2003). Gene networks inference using dynamic Bayesian networks. *Bioinformatics 19*(Suppl. 2), ii138–ii148.

Pritchard, J. K., M. T. Seielstad, A. Perez-Lezann, and M. W. Feldman (1999). Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Molecular Biology and Evolution 16*, 1791–1798.

Quach, M., N. Brunel, and F. d'Alché Buc (2007). Estimating parameters and hidden variables in non-linear state-space models based on ODEs for biological networks inference. *Bioinformatics 23*(23), 3209–3216.

R Development Core Team (2009). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. ISBN 3-900051-07-0.

Rangel, C., J. Angus, Z. Ghahramani, M. Lioumi, E. Southeran, A. Gaiba, D. L. Wild, and F. Falciani (2004). Modeling T-cell activation using gene expression profiling and state-space model. *Bioinformatics 20*(9), 1361–1372.

Rangel, C., J. Angus, Z. Ghahramani, and D. L. Wild (2005). *Probabilistic Modeling in Bioinformatics and Medical Informatics*, Chapter "Modeling Genetic Networks with State-Space Models", pp. 269–293. Springer.

Ratmann, O., O. Jorgensen, T. Hinkley, M. Stumpf, S. Richardson, and C. Wiuf (2007). Using likelihood-free inference to compare evolutionary dynamics of the protein networks of *H. pylori* and *P. falciparum*. *PLoS Computational Biology 3*(11), 2266–2278.

Rau, A., F. Jaffrézic, J.-L. Foulley, and R. W. Doerge (2010). An empirical Bayesian method for estimating biological networks from temporal microarray data. *Statistical Applications in Genetics and Molecular Biology 9*(9), 1–28.

Rice, J. J., Y. Tu, and G. Stolovitzky (2005). Reconstructing biological networks using conditional correlation analysis. *Bioinformatics 21*, 765–773.

Robbins, H. (1955). An empirical Bayes approach to statistics. In *Proceedings of the Third Berkeley Symposium Mathematical Statistics and Probability I*, Berkeley, pp. 157–164. University of California Press.

Robert, C. and G. Casella (2004). *Monte Carlo Statistical Methods*. Springer.

Rogers, S. and M. Girolami (2005). A Bayesian regression approach to the inference of regulatory networks from gene expression data. *Bioinformatics 21*(14), 3131–3137.

Romualdi, C., S. Bortoluzzi, and G. Danieli (2001). Detecting differentially expressed genes in multiple tag sampling experiments: comparative evaluation of statistical tests. *Human Molecular Genetics 10*, 2133–2141.

Ronen, M., R. Rosenberg, B. I. Shraiman, and U. Alon (2002). Assigning numbers to the arrows: parameterizing a gene regulation network by using accurate expression kinetics. *PNAS 99*(16), 10555–10560.

Rubin, D. B. (1984). Bayesienly justifiable and relevant frequency calculations for the applied statistician. *Annals of Statistics 12*, 1151–1172.

Ruijter, J. M., A. H. C. V. Kampen, and F. Baas (2002). Statistical evaluation of SAGE libraries: consequences for experimental design. *Physiological Genomics 11*, 37–44.

Saccharomyces Genome Database (2010). `http://www.yeastgenome.org`.

Sachs, K., O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan (2005). Causal protein-signaling networks derived from multiparameter single-cell data. *Science 308*(5721), 523–529.

Schäfer, J., R. Opgen-Rhein, and K. Strimmer (2006). Reverse engineering genetic networks using the GeneNet package. *R News 6*, 50–53.

Schäfer, J. and K. Strimmer (2005). An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics 21*(6), 754–764.

Schena, M., D. Schalon, R. W. Davis, and P. O. Brown (1995). Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science 270*(5234), 467–470.

Schena, M., S. J. Smith, and P. O. Brown (1996). A DNA microarray system for analyzing complex DNA samples using two-color fluorescent probe hybridization. *Genome Research 6*(7), 639–645.

Schlitt, T. and A. Brazma (2007). Current approaches to gene regulatory network modelling. *BMC Bioinformatics 8(Suppl 6)*(S9), 1–22.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics 6*(2), 461–464.

Serial Analysis of Gene Expression (2010). `http://www.sagenet.org`.

Shannon, P., A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker (2003). Cytoscape: A software environment for integrated model of biomolecular interaction networks. *Genome Research 13*, 2498–2504.

Shaw, C. and J. Tollett (2001). Image analysis and quantitation of cDNA microarray images. `http://www.bcm.edu/microarray/imageanalysis.pdf`.

Shimamura, T., S. Imoto, R. Yamaguchi, A. Fujita, M. Nagasaki, and S. Miyano (2009). Recursive regularization for inferring gene networks from time-course gene expression profiles. *BMC Systems Biology 3*(41).

Shumway, R. H. and D. S. Stoffer (1982). An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis 3*(4), 253–264.

Shumway, R. H. and D. S. Stoffer (2000). *Time Series Analysis and Its Applications*. Springer-Verlag.

Sisson, S. A., Y. Fan, and M. M. Tanaka (2007). Sequential Monte Carlo without likelihoods. *PNAS 104*, 1760–1765.

Smyth, G. K. (2004). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology 1*(3), 1–26.

Smyth, G. K. and T. Speed (2003). Normalization of cDNA microarray data. *Methods 31*, 265–273.

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

Stein, C. M. (1981). Estimation of the mean of a multivariate Normal distribution. *Annals of Statistics 9*(6), 1135–1151.

Tegnér, J., M. K. S. Yeung, J. Hasty, and J. J. Collins (2003). Reverse engineering gene networks: Integrating genetic perturbations with dynamcal modeling. *Genetics 100*(10), 59445949.

Toni, T., D. Welch, N. Strelkowa, A. Ipsen, and M. P. H. Stumpf (2009). Approximate Bayesian Computation scheme for parameter inference and model selection in dynamic systems. *Journal of The Royal Society Interface 6*(31), 187–202.

Troyanskaya, O., M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman (2001). Missing value estimation for DNA microarrays. *Bioinformatics 17*(6), 520–252.

Tseng, G. C. (2007). Penalized and weighted K-means for clustering with scattered objects and prior information in high-throughput biological data. *Bioinformatics 23*(17), 2247–2255.

Tĭno, P. (2009). Basic properties and information theory of Audic-Claverie statistic for analyzing cDNA arrays. *BMC Bioinformatics 10*(310).

Velculescu, V. E., L. Zhang, B. Vogelstein, and K. W. Kinzler (1995). Serial analysis of gene expression. *Science 270*, 484–487.

Velculescu, V. E., L. Zhang, W. Zhou, J. Vogelstein, M. A. Basrai, D. E. Bassett, P. Hieter, B. Vogelstein, and K. W. Kinzler (1997). Characterization of the yeast transcriptome. *Cell 88*.

Watson, J. D. and F. H. C. Crick (1953). A structure for deoxyribose nucleic acid. *Nature 171*, 737–738.

Werhli, A. V. and D. Husmeier (2007). Reconstructing gene regulatory networks with Bayesian networks by combining expression data with multiple sources of prior knowledge. *Statistical Applications in Genetics and Molecular Biology 6*(1), Article 15.

Wessels, L. F. A., E. P. Van Someren, and M. J. T. Reinders (2001). A comparison of genetic network models. *Pacific Symposium on Biocomputing 6*, 508–519.

Wilkinson, D. J. (2009). Stochastic modelling for quantitative description of heterogeneous biological systems. *Nature Reviews Genetics 10*, 122–133.

Wu, F. X., W. J. Zhang, and A. J. Kusalik (2004). Modeling gene expression from microarray expression data with state-space equations. *Pacific Symposium on Biocomputing 9*, 581–592.

Yang, Y. H., S. Dudoit, P. Luu, and T. P. Speed (2001). Normalization for cDNA microarray data. *SPIE BiOS*.

Yaragatti, M., T. Sandler, and L. Ungar (2009). A predictive model for identifying mini-regulatory modules in the mouse genome. *Bioinformatics 25*(3), 353–357.

Yeung, M. K. S., J. Tegner, and J. J. Collins (2002). Reverse engineering gene networks using singular value decomposition and robust regression. *PNAS 99*(9), 6163–6168.

Zak, D. E., F. J. I. Doyle, G. Gonye, and J. S. Schwaber (2001). Simulation studies for the identification of genetic networks from cDNA array and regulatory activity data. *Proceedings of the 2nd International Conference on Systems Biology*, 231–238.

Zak, D. E., G. E. Gonye, J. S. Schwaber, and F. J. Doyle (2003). Importance of input perturbations and stochastic gene expression in the reverse engineering of genetic regulatory networks: Insights from an identifiability analysis of an in silico network. *Genome Research 13*, 2396–2405.

Zou, M. and S. D. Conzen (2005). A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. *Bioinformatics 21*(1), 71–79.

VITA

VITA

Andrea Rau was born November 29, 1982 in Minot, North Dakota. She received a B.A. in French and Mathematics with a Concentration in Statistics from Saint Olaf College in Northfield, Minnesota in 2005 and a M.S. in Applied Statistics from Purdue University in 2007. She has been a graduate student in the Statistics Department at Purdue University since 2005.