Bayesian Methods for Inferring Gene Regulatory Networks

William Chad Young

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2016

Reading Committee:

Adrian E. Raftery, Chair

Vladimir Minin

Ka Yee Yeung

Program Authorized to Offer Degree:
Statistics

University of Washington

Abstract

Bayesian Methods for Inferring Gene Regulatory Networks

William Chad Young

Chair of the Supervisory Committee:
Professor Adrian E. Raftery
Department of Statistics

The recent explosion in the availability of gene expression data has opened up new possibilities in advancing our understanding of the fundamental processes of life. To keep up with the increasing size of the datasets, new models and inference methods must be developed that can handle tens of thousands of genes efficiently, provide a systematic framework for the integration of multiple data sources, and yield robust, accurate, and compact gene regulatory networks. In this thesis, I develop methods for different types of gene expression data that are able to handle large datasets and produce meaningful inference of relationships between genes. These methods also incorporate information from other data sources in the form of an informative prior when available. These methods are applied to gene expression data from yeast and humans, as well as synthetic benchmark data. I also look at data artifacts present in big data in biology and propose a model-based clustering method for addressing these issues and correcting the data and show how the improved data lead to improved subsequent analysis. Finally, I propose a model for inferring network information from steady-state data. I prove some theoretical results about a constrained version of the model and explore results from applying it to synthetic benchmark data.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ARACNE: algorithm for the reconstruction of accurate cellular networks.

BMA: Bayesian model averaging.

DBN: dynamic Bayesian network.

EM: expectation-maximization.

CLR: context likelihood of relatedness.

DREAM: dialogue on reverse-engineering assessment and methods.

LASSO: least absolute shrinkage and selection operator.

LINCS: Library of Integrated Network-based Cellular Signatures.

LR: likelihood ratio.

MAE: mean absolute error.

MCMC: model-based clustering with data correction.

MLE: maximum likelihood estimate.

MRNA: messenger RNA.

MRMR: minimum redundancy maximum relevance.

NIH: National Institutes of Health.

SAM: significance analysis of microarrays.

SEM: structural equation modeling.

T&J: TRANSFAC and JASPAR.

VAR: vector auto-regressive.

ACKNOWLEDGMENTS

This thesis is the result of years of work, but also would not have been possible without the contributions and support of so many people around me.

First, I would like to thank my advisors Adrian Raftery and Ka Yee Yeung. They provided me with engaging research projects from my first year at the University of Washington. They have helped me become a better researcher with their insight and feedback. I would also like to thank my committee members Roger Bumgarner and Vladimir Minin. They have helped shape my time here through classes as well as comments and questions about my work.

I am so thankful for family and friends who have been and continue to be encouraging and supportive. My parents, Paul and Kim Young, have supported me and my family in this endeavor far beyond I would ever think to ask. This never would have happened if my wife, Michelle, had not been open to taking the risk of moving our family and leaving the known so that I could pursue something that is better for me. Michelle, I am humbled by your faith in me and all the ways you make sure our family thrives.

Finally, I am extremely grateful for the financial support of the ARCS Foundation; the Department of Statistics and the Center for Statistics and Social Sciences (CSSS) at the University of Washington; the National Institute of Health (NIH; Grants U54-HL127624, R01-HD054511 and R01-HD070936)

DEDICATION

To my amazing wife, Michelle, and my awesome children Gavin, Miriel, and Arianna.

Chapter 1

INTRODUCTION

1.1 Gene Regulatory Networks

In 1958 (Crick, 1958), Francis Crick, along with James Watson a discoverer of the structure of DNA, first stated the central dogma of molecular biology. Refined in (Crick and others, 1970), this is now most simply stated as "DNA makes RNA makes protein". This reflects the understanding that within cells, the transfer of information flows from the cell's DNA to transcribe RNA for that cell. The RNA then facilitates the creation of proteins via translation. These proteins are the structural building blocks of the cell and are also required for the various functions of the cell, including facilitation of the production of RNA. This intricate dance within the cells of all living things enables the various functions of the cells. Understanding the complexities of these interactions has been the focus of much research, with the knowledge that learning about the genetic interactions within cells will lead to a much better understanding of how organisms develop and function at the cellular level.

In computational and systems biology, the central dogma is often framed in terms of gene regulatory networks. A gene regulatory network is a set of regulators within the cell which interact with each other and govern the amount of RNA and protein that is produced. In general, a gene from a cell's DNA is expressed by being transcribed into messenger RNA (mRNA), which is then used to create a protein. This protein then interacts with the cell or its environment according to the protein's structure. This may include enhancing or inhibiting the expression of other genes in that cell.

One way to learn about these vast, complicated networks is to focus on just a subset of the actors in the network. For instance, some researchers focus only on protein-protein interactions in order to shed some light in that area (Jeong et al., 2001; Rual et al., 2005).

In this dissertation, I focus on RNA-RNA regulation from gene expression data (Segal et al., 2003). RNA abundance in a cell is an indication of the activity level of the associated genes in the DNA as well as a proxy for protein abundance.

1.1.1 Gene Expression Data

Gene expression technologies measure the abundance of mRNA in a cell or group of cells. One common method for measuring this mRNA is with the use of microarrays (Schena et al., 1995; Lockhart et al., 1996; Ball et al., 2002). This works by manufacturing sequences of DNA that are complementary to the mRNA from a particular gene. The mRNA is made up of a sequence of bases, each of which attaches preferentially to its complementary base. The complementary sequences of DNA are labelled with a fluorescent marker which fluoresces when attached, or hybridized, to the correct mRNA. Microarrays consist of making a chip with many spots, each containing the complementary DNA for the mRNA of a single gene. A solution of cells, having been prepared in order to allow the mRNA to be available for hybridization, is washed over the chip. The mRNA attaches to the available DNA in each spot, and the spot fluoresces in proportion to the concentration of its gene's mRNA level in the solution. The fluorescence at each spot is then measured, resulting in expression levels for the set of measured genes.

Recent improvements in gene expression measurement technologies, such as microarrays and RNAseq (Wang et al., 2009), have greatly increased the amount of gene expression data available. Gene expression experiments have been done extensively on many different organisms, from yeast to human cells. One of the most important features of gene expression data is that it can be done under a variety of experimental conditions. Drug perturbations are done by introducing a chemical into a culture of cells and recording how the gene expression levels change in response to the drug. In knockdown and over expression experiments, a gene is targeted to control its expression level, either inhibiting its expression or encouraging it, and again the expression levels of the genes are measured in response.

1.2 Causality

One question to be addressed when using gene expression data to infer relationships between genes is that of causality (Pearl, 2009). Many methods focus on inferring relationships between genes that do not include a directional feature, resulting in networks that lack directionality (Basso et al., 2005; Margolin et al., 2006; Faith et al., 2007; Meyer et al., 2007). I am interested in learning which gene is the influencing and which is the responding gene. This can be difficult in part due to the vast number of genes that are potentially interacting. The human genome, for example, contains over 20,000 genes, and in many experiments only a subset of the genes are measured. Because of this, many direct relationships cannot be observed, and we may only be able to make inference of indirect or mediated relationships between genes. In a given data set we may observe a relationship between gene A and gene B, for example, but we cannot say that this is a direct relationship. It could be the case that gene C, not observed in the experiment, is influenced by gene A and in turn affects gene B.

Further complicating the picture is that in most gene-expression datasets the number of genes measured far exceeds the number of observations. This requires that additional assumptions be made about the data and limits inference (Wainwright, 2009; Verzelen, 2012).

Causal inference also depends on the type of data being analyzed. In many cases, knockout experiments provide the most reliable information regarding regulatory relationships among genes (Pinna et al., 2010). In a knockout experiment, a single gene is targeted and suppressed such that it is not expressed in the cell. If we compare the resulting gene expression levels of all other genes with a baseline measurement of the expression levels when the target gene is knocked-out, we can see directly what the impact of that gene is on the network. The inferred relationships still may not reflect direct interactions between the knocked out gene and the other genes. However, if knockout experiments are performed on many genes, the resulting network can be improved by using an edge reduction technique to prune extraneous edges.

Unfortunately, it is not possible to perform knockout experiments on all genes in a cell.

There are many genes that are crucial to keeping the cells alive, and removing them would kill the cell. It is more practical to perform knockdown or overexpression experiments, where a target gene is partially inhibited or induced to express at higher than normal levels. The efficacy of these experiments can vary greatly in the resulting level of expression, but there is still the concept of a causal gene being experimentally controlled.

Drug perturbation experiments are similar to knockdown or overexpression experiments in that many drugs affect cells by targeting and affecting the expression of a small set of genes. These targets are not well known for all drugs, and often these experiments are done in order to identify the targets. If the targets of a drug are known, then causal inference can be done in a similar manner to knockdown or overexpression experiments, often with a small set genes identified as the causal genes.

Another popular form of gene expression experiment produces time-series data (Ernst and Bar-Joseph, 2006). A culture of cells is prepared in a culture and at intervals a sample of the cells is taken and the gene expression levels are measured. These time-series experiments show the dynamic behavior of cells when exposed to some perturbation, whether drug or controlled gene expression. Time-series data provide a rich set of information about the interactions between genes but do not provide the same strong causality as knockdown or overexpression experiments.

1.3 Dissertation Outline

The remainder of this dissertation is structured as follows. In Chapter 2 I consider time-series gene expression data and develop ScanBMA, an improved algorithm for Bayesian model averaging. ScanBMA is applied to time-series data from yeast as well as simulated data from the DREAM4 competition, and comparison is made with other common methods for analyzing time-series gene expression data. This chapter is closely based on (Young et al., 2014), published in *BMC Systems Biology*.

Chapter 3 introduces a simple Bayesian approach for analyzing gene knockdown experiments. Based on a simple linear regression model, the method calculates posterior probabil-

ities of relationships between the knocked-down gene and each other gene measured in the experiments. The posterior probability method is compared with other methods appropriate for knockdown data on data from the LINCS L1000 experiments. The LINCS L1000 data is a large collection of gene expression experiments on human cell lines under a variety of experimental conditions. This chapter is closely based on (Young et al., 2016b), published in *Mathematical Biosciences and Engineering*.

In Chapter 4, I show how the data-processing pipeline of the L1000 data introduces artifacts in the data. A method is proposed and developed for identifying and correcting these artifacts, model-based clustering with data correction (MCDC). MCDC is applied to simulated data as well as the L1000 data, and the posterior probability method of Chapter 3 is applied to the corrected data, showing the improvements in subsequent analysis. This chapter is closely based on (Young et al., 2016a), to be published in *Annals of Applied Statistics*.

In the LINCS L1000 data, there is a plethora of baseline data, used as a reference against which to find differences in expression in the various experiments. Chapter 5 introduces a way to use this baseline, static data to learn about the underlying network. In particular, I look at the equilibrium distribution of a vector autoregressive model (VAR model) to see under what circumstances inference can be made about the network edges. A proof is presented for consistency and efficiency of the MLE in a restricted case, and a likelihood-ratio test is developed for the existence of a single edge. Simulation results are presented, and the method is applied to a small subnetwork in yeast.

Chapter 6 concludes with a summary of key contributions as well as a discussion of possible future work.

Chapter 2

SCANBMA APPLIED TO TIME SERIES DATA

2.1 Motivation - Time Series Data

One common type of gene expression data is time-series data - obtaining and analyzing samples from a culture of cells at multiple different times. This gives insight into how the genes are expressed dynamically over time. Many of these datasets from microarrays and next generation sequencing quantify the expression levels of all genes in a given genome. Genome-wide time-series data, in principle, allow reverse engineering of the gene regulatory relationships by studying the temporal patterns of regulators and target genes. However, this can be a difficult problem due to the large number of genes (*i.e.* variables) being measured, typically far exceeding the number of observations. Also, the number of actual regulators for a particular gene is only a small fraction of the number of possible regulators. In this case the true signal can be difficult to find among the plethora of noisy, potentially confounding variables.

A popular method for inferring gene regulatory networks from time series data uses Dynamic Bayesian Networks (DBN) to model the network holistically (Murphy et al., 1999; Kim et al., 2003, 2004; Zou and Conzen, 2005; Lèbre et al., 2010; Zhu et al., 2010; Ghanbari et al., 2015). DBN methods estimate a probabilistic graphical model, given the time-series data. DBN methods work well, but the network size that can be handled in practice is limited because of their computational cost.

Ordinary differential equations (ODE) are an alternative method for constructing networks (D'haeseleer et al., 1999; Bansal et al., 2006). These methods are deterministic rather than statistical, meaning that quantification of uncertainty is often missing. ODE methods can, however, be combined with statistical methods. Li et al. (2011), for example, use DBN

on local networks within a larger ODE model inference method.

Another class of methods is based on regression models, where the expression level of a target gene is modeled as a function of the exression level of that gene's regulators. Vector autoregressive models have been proposed for inferring causal links between genes. Often, this takes the form of a variable or model selection problem, and methods such as LASSO (Tibshirani, 1996; Tibshirani et al., 2005; Gustafsson et al., 2009; Men'endez et al., 2010), elastic net (Zou and Hastie, 2005; Friedman et al., 2010), and Bayesian model averaging (BMA) (Raftery et al., 1997; Hoeting et al., 1999) have been used (Gustafsson et al., 2009; Men'endez et al., 2010; Shojaie and Michailidis, 2010; Yeung et al., 2011; Lo et al., 2012; Young et al., 2014) . Morrissey et al. (2010) implemented a MCMC sampler for a fully Bayesian formulation of the autoregressive model.

Mutual information methods have been used extensively on genetics data (Basso et al., 2005; Margolin et al., 2006; Faith et al., 2007; Meyer et al., 2007), but usually for steady-state rather than time-series data. These methods are typically non-directional. Zoppoli et al. (2010) and Lopes et al. (2011) have recently extended mutual information methods to analyze time-series data and produce directed networks. Mutual information methods have the advantage of being able to identify nonlinear relationships.

2.2 Method - ScanBMA

I developed a new approach using Bayesian Model Averaging (BMA) for variable selection from time-series gene expression data. The new method offers the following advances:

- I transform the time-series data to reduce spurious correlations.

- I develop a new algorithm called ScanBMA that searches the model space more efficiently and thoroughly than previous implementations. It is much faster than previous implementations of BMA for a large number of predictors, resulting in running time comparable to that of LASSO. It allows inference for networks of thousands of genes to be completed in minutes on a standard laptop computer.

- I integrate external information as a prior on regulatory relationships.

I applied my approach, ScanBMA, in two data contexts. The first was the time-series data measuring the gene expression levels of 97 haploid yeast segregants perturbed with the drug rapamycin. The second consisted of simulated time-series data from the DREAM4 (Dialogue for Reverse Engineering Assessment and Methods) challenge. I compared the performance of my method, ScanBMA, to other network construction methods in the literature, namely LASSO, the mutual information methods MRNET (Maximum Relevance/Minimum Redundancy), CLR (Context Likelihood or Relatedness) and ARACNE (Algorithm for the Reconstruction of Accurate Cellular Networks), and also Dynamic Bayesian Networks when the latter were computationally feasible. For the yeast dataset, I found that my method outperformed the competitors and previous analyses in recovering known regulatory relationships previously reported in the literature. For the DREAM4 data, for which no prior information was available, ScanBMA performed comparably to other methods, while producing more compact networks. Finally, the ScanBMA algorithm presents a significant improvement in running time over previous implementations of BMA.

2.2.1 Bayesian Model Averaging

In regression-based methods for network inference, I infer regulators (parent nodes) for each target gene. Hence, network inference can be formulated as a series of variable selection problems. I use the following multiple linear regression model for network inference from time-series data:

$$X_{i,t,s} = \beta_{0,i} + \sum_{h \in H} \beta_{hi} X_{h,t-1,s} + \varepsilon_{i,t,s},$$

where $X_{i,t,s}$ is the expression level for gene i at time t for strain or replicate s, H is the set of potential regulators, and $\varepsilon_{i,t,s} \overset{\text{iid}}{\sim} N(0, \sigma_\varepsilon^2)$ is the error term, for $i = 1, \ldots, n$, $t = 2, \ldots, T$ and $s = 1, \ldots, S$. I am particularly interested in whether $\beta_{hi} \neq 0$, indicating a regulatory relationship from gene h to gene i.

One difficulty is that for gene network time series data there are typically far more potential regulators than observations. To address this problem, as well as to take into account uncertainty in model selection, I use BMA to obtain the posterior probability that each regulator is in the model (Raftery et al., 1997; Hoeting et al., 1999). BMA takes model uncertainty into account by averaging over the posterior distributions of a quantity of interest based on multiple models, weighted by their posterior model probabilities. Thus the posterior probability that $\beta_{hi} \neq 0$, also called the posterior inclusion probability of β_{hi}, is

$$\Pr(\beta_{hi} \neq 0|D) = \sum_{k=1}^{K} \mathbf{1}(\beta_{hi} \neq 0|D, M_k)\Pr(M_k|D),$$

where $\Pr(M_k) \propto \prod_{h \in M_k} \pi_{hi}$, the models considered are denoted by $M_1, \ldots, M_K$, D denotes the data, $\Pr(M_k)$ is the prior model probability of model M_k, and π_{hi} is the prior probability that gene h regulates gene i.

An additional issue is that the number of possible models is too large to enumerate in a reasonable amount of time. MCMC approaches have been applied where the number of potential regulators is large (Bottolo et al., 2010), but they are expensive computationally. Iterative BMA has also been applied to large genetics datasets, but the number of regulators was restricted prior to analysis (Yeung et al., 2005, 2011).

2.2.2 Data Transformations

One concern when identifying regulatory relationships among genes is that there is a great deal of variation in gene expression levels that does not come from these interactions. For example, in the yeast data, many of the genes experience a sharp change in expression level over the first few time points caused by the application of the drug. This common trajectory is not important for inference and can produce many large correlations that do not correspond to actual interactions. In addition, I have found that removing the effect of a gene on itself can improve inference. In doing this I remove excess variation in order to gain accuracy in inferring relationships.

In light of these observations, I transform the data in two steps to remove these extra sources of variation. First, I use time-adjusted data by subtracting the mean expression level for each gene i at timepoint t across strains:

$$X^*_{i,t,s} = X_{i,t,s} - \bar{X}_{i,t,\cdot},$$

where $\bar{X}_{i,t,\cdot} = S^{-1} \sum_{s=1}^{S} X_{i,t,s}$. This removes the overall effect of the drug.

Second, I take the residuals from regressing the gene on itself at the previous timepoint:

$$X^*_{i,t,s} = X_{i,t,s} - \hat{\alpha}_i X_{i,t-1,s},$$

where α_i is the regression coefficient in the simple linear regression model of the expression level of gene i at time t on its expression level at time $t-1$:

$$X_{i,t,s} = \alpha_i X_{i,t-1,s} + \delta_{i,t,s},$$

where $\delta_{i,t,s} \overset{\text{iid}}{\sim} N(0, \sigma_i^2)$. Then $\hat{\alpha}_i$ is the least squares estimate of α_i based on all $S(T-1)$ relevant observations for gene i.

2.2.3 Specifying the Prior

An important feature of this method is a way to incorporate external information. The Bayesian approach requires the specification of prior distributions and so includes this directly.

BMA requires two types of prior information: the prior edge probability, π_{hi}, that potential regulator h regulates target gene i, for each h and i, and the prior distribution of the parameter vector for each model considered.

For the prior edge probabilities π_{hi}, I considered two different prior distributions. The first is based on the empirical finding of Guelzim et al. (2002) that each target gene is regulated by a small number of transcription factors, estimated as 2.76 per gene (Lo et al., 2012). I implement this by setting $\pi_{hi} = 2.76/6000$ for all h and i. This prior distribution does not incorporate any gene-specific information. Denote this the Guelzim prior.

The second prior edge distribution I consider is that of Lo et al. (2012), which uses external information in the form of expression data, binding-site data, curated literature and other sources to come up with a prior probability for each individual possible regulatory relationship. This leads to values of π_{hi} specific to each regulator-target pair. I refer to this as the informative prior. Integration of multiple information sources has been shown to be beneficial in network construction (Zhu et al., 2008; Yip et al., 2010).

2.2.4 Zellner's g-prior and EM Optimization

As a prior for the model parameters, corresponding to the strengths of the relationships, I use Zellner's g-prior (Zellner, 1986), as in Clyde and George (2004). The prior distribution of the parameter vector $(\beta_{0,i}^{[k]}, \beta^{[k]}, \sigma^{2[k]})$ of model M_k is

$$\begin{aligned} \beta^{[k]}|\sigma^{2[k]}, M_k, g &\sim N(0, g\sigma^{2[k]}(X_k^T X_k)^{-1}), \\ \Pr(\beta_{0,i}^{[k]}, \sigma^{2[k]}|M_k) &\propto 1/\sigma^{2[k]}, \end{aligned}$$

where X_k is the design matrix for model M_k and $g > 0$ controls the prior variance of the regression parameters. This prior yields an analytic form for the posterior model probabilities $p(M_k|D)$, namely

$$2\log\left(p(M_k|D)/p(M_0|D)\right) = (n-1)\log(1+g(1-R_k^2)) -$$

$$(n-d_k-1)\log(1+g) - 2\sum_{h\in\mathcal{M}_k} \log\left(\frac{\pi_{hi}}{1-\pi_{hi}}\right),$$

where M_0 is the null model with no regulators, d_k is the number of regulators present in model M_k, and R_k^2 is the R^2 value for M_k. This is an alternative to using BIC to approximate the posterior model probabilities, as has been done previously (Raftery, 1995; Yeung et al., 2005).

The parameter g controls the expected size of the regression parameters β_{hi}, and is approximately equal to the prior variance of $\beta_{hi}/\mathrm{SE}(\hat{\beta}_{hi})$, where $\hat{\beta}_{hi}$ is the OLS estimator. This suggests that g should be at least 1, otherwise the individual β_{hi}'s would be expected to be nearly indistinguishable from the noise even under ideal conditions. Also, the effective

number of data points in the prior is n/g. Using $g = n$ corresponds to a unit information prior and yields similar results to using the BIC approximation. Raftery (1999) argued that the prior should be no more spread out than the unit information prior, suggesting a choice of g in the range $1 \leq g \leq n$.

I estimate g from the data by maximum marginal likelihood, where the likelihood is summed over the model space. I do this using an EM algorithm (Dempster et al., 1977; McLachlan and Krishnan, 2007), where the "missing data" are the γ_{hi}'s indicating whether regulator gene h is in the model for target gene i. First, I run BMA with a selected value of g, and from this I obtain the posterior probabilities of the models used. I then maximize

$$\sum_{k=1}^{K} \Pr(M_k|D) \left((n - d_k - 1) \log(1 + g) - (n - 1) \log \left[1 + g(1 + R_k^2) \right] \right)$$

with respect to g. use the new value of g in the next iteration of ScanBMA. I do this until convergence.

2.2.5 ScanBMA Algorithm - Searching the Model Space

To find the models to be included in the BMA summation, I use the Occam's window principle, according to which models that are substantially worse than the best model found (by a factor of C in posterior probability) are discarded (Madigan and Raftery, 1994). I used $C = 100$ in the present work. To find the models in Occam's window, I propose a new, fast algorithm called ScanBMA.

The main idea of ScanBMA is to keep an active set of models around which to search. All models which can be created by adding or removing a single predictor from those in the active set are then evaluated and, if they fall within Occam's window of the current best model, are added to the active set to expand the search. The method is outlined in Algorithm 1.

Because ScanBMA does not average over every model in the model space, the algorithm yields posterior inclusion probabilities of either 100% or 0% for many regulators. These extreme posterior inclusion probabilities are only approximations, and I can refine them. To

Algorithm 1: ScanBMA

Initialize $\mathcal{M}_{keep}, \mathcal{M}_{next} = \{\}$

Initialize $\mathcal{M}_{active} = \{null\ model\}$, $bestScore = 0$

while $\mathcal{M}_{active}$ *not empty* **do**

  for *model* m_{new} *in* NeighborsOf*(*$\mathcal{M}_{active}$*)* **do**

    $mScore$ = EvaluateModelScore(m_{new})

    if $mScore$ *in* OccamsWindow*(*$bestScore$*)* **then**

      add m_{new} to $\mathcal{M}_{next}$

      $bestScore$ = BestModelScore($bestScore$, $mScore$)

    end

  end

  Trim models from $\mathcal{M}_{keep}$ according to $bestScore$

  Add good models from $\mathcal{M}_{active}$ to $\mathcal{M}_{keep}$

  $\mathcal{M}_{active}$ = good models from $\mathcal{M}_{next}$

end

return $\mathcal{M}_{keep}$

estimate the posterior inclusion probability of a predictor X_h with an approximate posterior probability of 100%, I calculate the ratio $O_{-h,i}$ of the posterior probability of the best model to that of the best model with the predictor X_h removed. I then approximate the posterior probability of predictor h by $O_{-h,i}/(1 + O_{-h,i})$, which will be less than 100%.

Similarly, to estimate the posterior inclusion probability of a predictor X_h with approximate posterior probability 0%, I compute $O_{h,i}$, the ratio of the posterior probability of the best model to that of the best model with the predictor X_h added. Then the approximate posterior inclusion probability of predictor h is $(1 + O_{h,i})^{-1}$, which will be greater than 0%.

This post-processing step yields a unique ordering among the predictors, which is useful in evaluation.

A variant of ScanBMA that that I found to greatly improve computational efficiency without degrading performance is to restrict the number, $nvar$, of potential regulators h considered for each target gene i to those with the highest prior probabilities, π_{hi}. Guelzim et al. (2002) found that the number of transcription factors per gene has approximately an exponential distribution with mean 2.76; they did not find any of the target genes that they considered to have more than 13 regulators. As a result, I consider $nvar = 20$, which leaves some margin over the Guelzim maximum. I compare this with considering all genes with observed variation in expression as potential regulators, which amounts to setting $nvar = 3556$.

2.3 Competing Methods

To evaluate the performance of ScanBMA, I compared it with LASSO (Tibshirani, 1996), as well as with the mutual information methods MRNET, CLR and ARACNE. LASSO is a variable selection based on a linear regression model, and thus can be used as an alternative to BMA. It is known for being computationally efficient. I used the implementation of LASSO in the `glmnet` package in `R` (Friedman et al., 2010), with the shrinkage penalty parameter, lambda, chosen by cross-validation.

Mutual information methods have been used extensively in identifying relationships

among genes (Basso et al., 2005; Margolin et al., 2006; Faith et al., 2007) . Since mutual information methods are non-directional, I used a modified version inspired by Shimamura et al. (2009), including the response as the first column of the matrix given to mutual information and the predictors as the other columns. I then took the first column from the resulting mutual information matrix as the measure of the directed relationship from the predictors to the target gene. MRNET (Maximum Relevance/Minimum Redundancy), CLR (Context Likelihood or Relatedness) and ARACNE (Algorithm for the Reconstruction of Accurate Cellular Networks) are different methods for using the mutual information to infer a weighted adjacency matrix between genes. All are implemented in the `minet` package in `R` (Meyer et al., 2008).

I have also investigated the possibility of using Dynamic Bayesian Networks (DBN). Methods based on DBNs have been implemented in several R packages. These include the `GeneNet` package (Sch afer et al., 2001), but this is not designed for multiple time-series and so was not a comparable method on the datasets. The `ebdbnet` (Empirical Bayes Estimation of Dynamic Bayesian Networks) R package (Rau et al., 2010), and the `G1DBN` R package (Smith et al., 2004), do allow for multiple time-series, but they are not designed to handle networks of the size of the yeast data. They are more appropriate for networks of sizes up to a few hundred genes. I was able to use the `ebdbnet` package for the much smaller DREAM4 networks.

2.4 Data

2.4.1 Data Description

To validate the ScanBMA method, I applied it to time-series data from a gene expression experiment on yeast as well as simulated time-series datasets from the DREAM4 challenge. The DREAM challenges provide a framework within which to benchmark gene inference methods by making data available, whether synthetic or from biological experiments, where the underlying truth of the network is known. As different methods are applied to the data,

this allows for a solid basis for comparison and benchmarking (Marbach et al., 2010, 2009; Prill et al., 2010).

The yeast data come from an experiment on 97 strains of yeast crossed from two parent strains (Yeung et al., 2011). Each strain was subjected to a treatment of the macrolide drug rapamycin, chosen to cause changes in gene transcription across the genome. Gene expression levels were measured for approximately 6,000 genes every 10 minutes from 0 to 50 minutes after the administration of the drug, using Affymetrix microarrays. The data were then filtered to remove genes that did not show significant variation over the measurements, leaving 3,556 genes. These yeast data are publicly available from ArrayExpress with accession number E-MTAB-412.

The simulated DREAM4 In Silico Network Challenge provided 5 networks each of size 10 and 100 genes (`https://www.synapse.org/#!Synapse:syn3049712/wiki/74628`). Time-series data for each network are produced by artificially perturbing a portion of the genes in the network and simulating the response of the network over time. For each network, 10 time series consisting of 21 time points were provided from which to make inference.

Since my focus is on time-series data, I did not use the other data sources provided by the DREAM4 challenge. In particular, the DREAM4 challenge provided the results of simulated gene knock-out experiments for all genes, and in fact the winning entry in the competition used only the knock-out data, and ignored the time series data (Pinna et al., 2010). In practice, however, this is unrealistic, since it is typically feasible to do knock-out experiments for only a small proportion of the genes. Time series data do have the potential to provide some information about all potential regulatory relationships, and my goal is to develop methods for doing this, so I have ignored the knock-out data here.

2.4.2 YEASTRACT - Assessment Data

A number of metrics have been used to evaluate the quality of inferred networks. I focus on a few that compare the inferred network with a gold-standard network of true edges. One measure that I use is the precision of the inferred network, equal to the number of true

positives divided by the total number of edges in the inferred network. Precision is important to researchers because an experiment to verify relationships identified in exploratory work can be expensive. Thus, the more confident I can be when identifying relationships, the better. In light of this, I also look at the area under the precision-recall curve (AUPRC). This gives a more comprehensive view of network quality and does not require that a threshold be chosen for the posterior probability of an edge or for the number of edges included. I also look at the area under the ROC curve (AUROC), which is widely used to assess the quality of networks.

In the case of yeast, the full regulatory network is not known, so I used a partial assessment based on the YEASTRACT database (Teixeira et al., 2006). This is a literature-curated repository of regulatory relationships between known transcription factors and target genes in yeast. Among the 3,556 genes in the filtered yeast time-series data, this database contains documented regulatory relationships for 127 transcription factors (TFs), encompassing a total of 17,173 edges. I therefore evaluate only inferred relationships from these 127 transcription factors, and inferred regulatory relationships from other genes are not used in evaluation. Due to incomplete knowledge in yeast biology, the lack of an edge in YEASTRACT is not hard evidence of the absence of a relationship between two genes, although it is used as such in my evaluation.

In contrast, the true underlying networks from the simulated DREAM4 data are known, so that an absence of a true edge is a false negative and a presence of a non-existing edge is a false positive.

2.5 Results

2.5.1 Yeast Results

Table 2.1 summarizes the assessment results for the methods applied to the yeast dataset. The preferred ScanBMA method with $nvar = 20$ had a precision of 0.39, much higher than any other method. The area under the ROC and precision-recall curves (AUROC and

AUPRC) for ScanBMA were also much better than for the competing methods. Note that for random guessing, the expected AUROC is 0.5 and the expected AUPRC is 0.038. For these data, the mutual information methods CLR and MRNET produced extremely large networks — nearly half of the number of possible edges, which is not concordant with what is known about regulatory networks of this type.

Table 2.2 compares the preferred version of ScanBMA with four other versions, each of which lacks one of the components of the final method. This table shows that each component contributes to the accuracy of the final network. As shown by Lo et al. (2012), the incorporation of the informative prior yields the largest increase in performance, but the other components also contribute. The use of the g prior reduces the number of false positives, while the data transformation substantially reduces the size of the inferred network.

The precision-recall curves for the different methods on the yeast data are shown in Figure 2.1. This figure shows the quality of the ScanBMA network even beyond the 95% posterior inclusion probability cutoff. The precision stays high through a large range of recall, whereas for the other methods it quickly drops to the level of random guessing. The figure also shows that $nvar = 20$ performs better than $nvar = 3556$.

When analyzing gene networks where the number of nodes is in the thousands, computation time can be an important consideration. I compared ScanBMA with the other methods on the yeast data by running each method on 20 target genes under controlled conditions to find the average cpu time per gene. All the methods compared are linear in the number of genes analyzed. Table 2.3 shows that ScanBMA with $nvar = 3556$, i.e. considering all other genes whose expression varied as potential regulators, is within a factor of 3 of LASSO, the fastest of the other methods. Some of the mutual information methods, on the other hand, are much slower, with MRNET taking about 50 times longer than ScanBMA. Dynamic Bayesian network methods were not included in the comparison because they analyze the entire network at once and do not scale well enough to be run on large network datasets such as the yeast data.

Table 2.1: Performance of different methods on the yeast data. AUROC is the area under the ROC curve, AUPRC is the area under the precision-recall curve, and TP and FP are the number of true positive and false positive edges inferred, respectively. Thus TP+FP is the number of edges in the inferred network and Precision = TP/(TP+FP). ScanBMA was applied to the transformed data using the informative edge prior and Zellner's g prior for the model parameters. The superscript indicates the value of $nvar$. Expected precision and AUPRC from random guessing is 0.0380.

Method	**Precision**	**AUROC**	**AUPRC**	**TP**	**FP**
LASSO	0.046	0.506	0.0416	996	20,469
ARACNE	0.205	0.502	0.0399	69	268
CLR	0.039	0.510	0.0435	8,879	220,942
MRNET	0.039	0.513	0.0442	8,737	214,757
ScanBMA$^{[20]}$	**0.391**	0.601	**0.0747**	227	353
ScanBMA$^{[3556]}$	0.274	**0.629**	0.0740	127	336

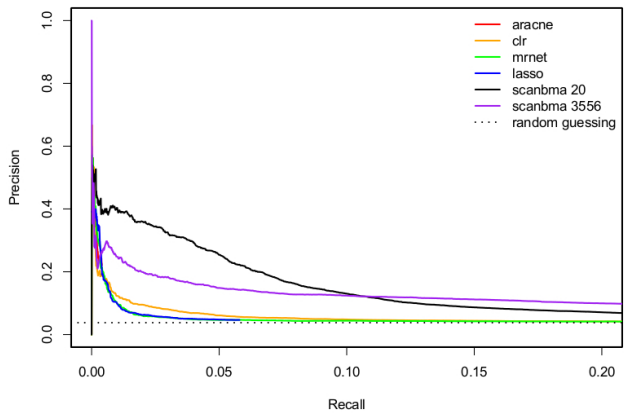

Figure 2.1: Precision-Recall curves for different methods on the yeast data. ScanBMA was run using the g-prior, transformed data, and informative prior with nvar=20 and 3556.

2.5.2 DREAM4 Results

Table 2.4 summarizes the results of the competing methods for the DREAM4 10-gene networks. For the DREAM4 networks, I was able to add the Dynamic Bayesian Networks

Table 2.2: Performance of different versions of ScanBMA on the yeast data, either on the original scale (Orig) or transformed (Trans). For the priors, inform refers to the informative prior, while Guelzim refers to the prior probability of 2.76/6000 for all possible relationships.

Data	**nvar**	**Priors**	**Precision**	**AUROC**	**AUPRC**	**TP**	**FP**
Trans	20	g, inform	**0.391**	0.601	**0.0747**	227	353
Trans	3556	g, inform	0.274	**0.629**	0.0740	127	336
Trans	20	BIC, inform	0.244	0.590	0.0616	200	619
Orig	20	g, inform	0.274	0.586	0.0680	552	1460
Trans	20	g, Guelzim	0.175	0.499	0.0395	34	160

method, as implemented in the `ebdbnet` R package (Rau et al., 2010), to the comparison because the networks are small. ScanBMA again performed best among the methods, particularly in terms of the areas under the ROC and precision-recall curves, even though no external information was available. However, the extent of ScanBMA's superiority to other methods, notably LASSO, was smaller in this case than for the yeast data, reflecting the lack of external information.

The precision-recall curves from the various methods are shown for the first of the five 10-gene networks in Figure 2.2. Figure 2.4 shows the actual networks returned by each method in comparison with the true network. The ScanBMA network exhibits high level of resemblance to the true network. In particular, the compactness of the ScanBMA network is apparent, particularly when compared with LASSO and MRNET. The small number of false positives is invaluable in focusing the attention of the biologist on edges of high interest when searching for new regulatory relationships.

The results of the methods for the DREAM4 100-gene networks are summarized in Table 2.5. For these networks, the mutual information methods MRNET and CLR performed best in terms of the area under curve measures. ScanBMA was not quite as good by these measures, but its precision was much higher than that of any other method. Figure 2.3

Table 2.3: Average CPU time per target gene of different methods for the yeast data. ScanBMA was run with g prior, transformed data, informative prior. Superscript indicates value of $nvar$.

Method	**Running Time Per Gene (s)**
LASSO	4.1
ARACNE	70.4
CLR	7.9
MRNET	>500
ScanBMA$^{[3556]}$	11.2
ScanBMA$^{[20]}$	0.04

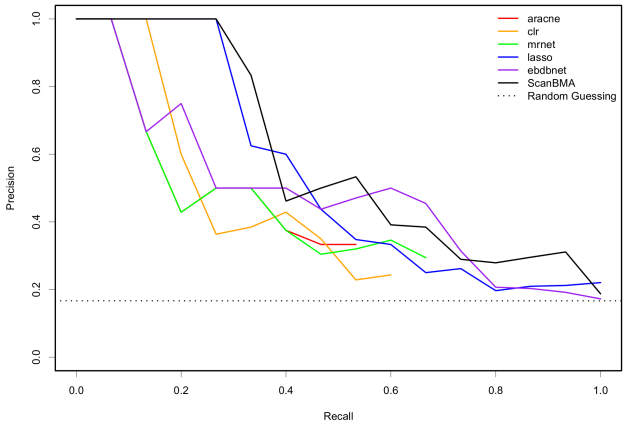

Figure 2.2: Precision-Recall curves for various methods on network 1 of the 10-gene networks from the DREAM4 competition. This network has 15 true edges.

illustrates the precision-recall curves for the various methods, showing the increased precision across a broader range for ScanBMA.

2.6 Discussion

In this chapter, I presented a Bayesian Model Averaging method for inferring gene regulatory networks from time series data. It incorporates external information in a principled way via

Table 2.4: Average performance of different methods on the DREAM4 10-gene networks. ScanBMA was run with the original data. The true positive (TP) and false positive (FP) columns are totaled across all 5 networks. There are 71 true edges across the 5 networks.

Method	**Precision**	**AUROC**	**AUPRC**	**TP**	**FP**
LASSO	0.190	0.731	0.487	62	265
ebdbnet	**0.509**	0.704	0.438	28	27
ARACNE	0.304	0.668	0.388	35	80
CLR	0.215	0.681	0.397	50	183
MRNET	0.215	0.709	0.409	53	193
ScanBMA	0.432	**0.740**	**0.505**	35	46

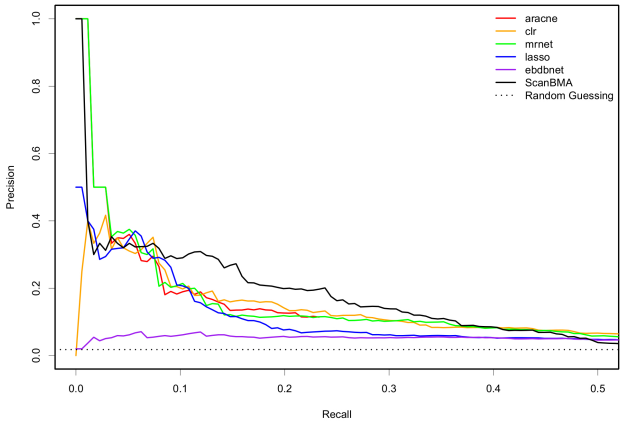

Figure 2.3: Precision-Recall curves for various methods on network 1 of the 100-gene networks from the DREAM4 competition.

the prior edge probabilities, transforms the data to reduce spurious correlations, and uses Zellner's g-prior for model parameters with g estimated from the data. I have introduced a new algorithm, ScanBMA, to search the model space efficiently. This method infers compact networks with high precision, important features for further analysis in searching for new regulatory relationships.

I found that ScanBMA outperformed previous methods as well as LASSO and mutual

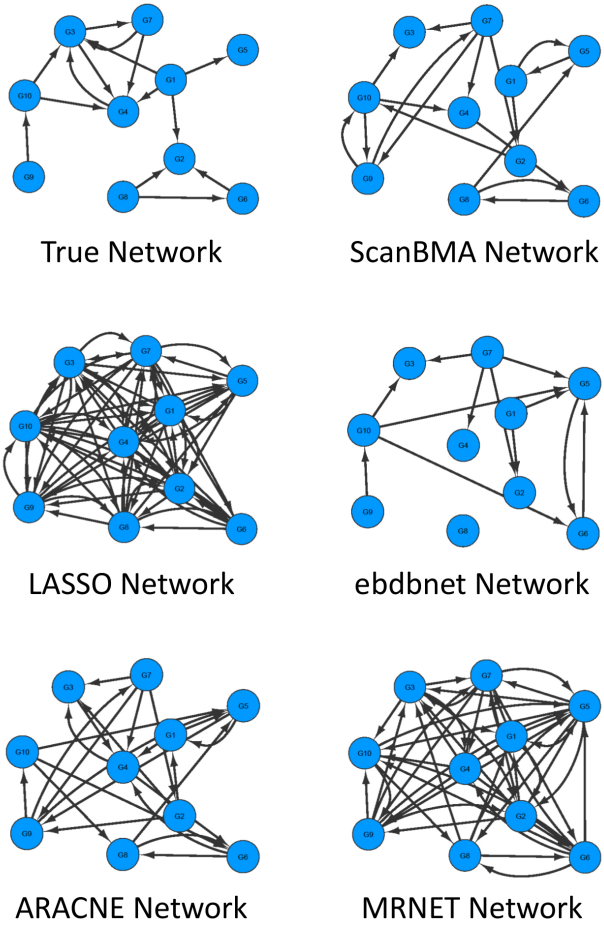

Figure 2.4: DREAM4 10-gene network visual comparison.

Table 2.5: Average performance of different methods on the DREAM4 100-gene networks. ScanBMA run with original data. The true positive (TP) and false positive (FP) columns are totaled across all 5 networks. There are 1,024 true edges across the 5 networks.

Method	**Precision**	**AUROC**	**AUPRC**	**TP**	**FP**
LASSO	0.035	0.643	0.073	571	15757
ebdbnet	0.054	0.643	0.043	182	3201
ARACNE	0.114	0.589	0.106	208	1621
CLR	0.035	0.699	0.123	678	18669
MRNET	0.035	**0.701**	**0.130**	689	18784
ScanBMA	**0.153**	0.657	0.101	193	1062

information methods on yeast time-series data. In addition, the method performed comparably to competing methods, including Dynamic Bayesian Networks, on simulated data from the DREAM4 challenge, even in the absence of prior information. The networks from ScanBMA are also similar in size to the target networks.

Chapter 3

POSTERIOR PROBABILITIES FOR PERTURBATION DATA

3.1 Motivation - Perturbation Data

In this chapter, I propose a simple, fast method for inferring gene regulatory relationships from just knockdown data. This method uses a simple linear regression model focusing on single regulator-target gene pairs. This method allows the incorporation of prior knowledge about the relationships and generates posterior probabilities which can be used to form a ranked edgelist or as a part of a more expansive analysis.

In most cases the number of genes measured far exceeds the number of observations, as is typical in microarray or sequencing experiments. Any method for analyzing such data must take this fact into account. Often this is done by enforcing a sparsity constraint, either via an added penalty on non-sparseness or via priors placed on the model. Even with these constraints, the ability to make valid inference is limited in these high-dimensional regimes as the number of genes grows compared to the number of observations (Wainwright, 2009; Verzelen, 2012).

When looking for regulatory relationships, it has been found that the information available from knockout experiments, where a single gene is fully suppressed, can be highly informative since they give a way to identify a causal pathway, direct or indirect. In the DREAM4 in silico network challenge (Marbach et al., 2009, 2010), `https://www.synapse.org/#!Synapse:syn3049712/wiki/74628`, one of the top scoring submissions used only the data from the knockout experiments to infer the true networks, ignoring the time-series data entirely (Pinna et al., 2010). In real biological experiments, full knockout experiments are not possible for many essential genes, but knockdown experiments, where the target gene is partially suppressed, are often available.

Many methods have been developed for inferring relationships between genes from knockdown data. Mutual information methods are often used, as well as correlation-based approaches (Salleh et al., 2015) and implicit latent variable scoring (Yoo et al., 2002). Significance Analysis of Microarrays (Tusher et al., 2001) is a differential expression method used to identify genes whose expression levels change due to perturbation experiments. Another approach is to model the network holistically using Bayesian networks (Friedman et al., 2000; Pe'er et al., 2001; Rogers and Girolami, 2005; Fröhlich et al., 2007; Scutari, 2010; Cho et al., 2016). This yields good interpretability, but often does not scale well and is difficult to apply at the whole-genome level. Regression-based methods, where the expression level of a target gene is modeled as a function of the exression level of that gene's regulators, can be applied to much larger sets of genes but lack the same overarching model of the entire network. Inference for these models generally becomes a statistical variable selection or model selection problem (Bühlmann et al., 2010). Common methods for this include previously mentioned methods, such as LASSO and BMA. In addition, there has been some work in combining steady-state data with knockdown data to improve results (Christley et al., 2009; Shojaie et al., 2014).

3.2 L1000 Data

My data come from the Library of Integrated Network-based Cellular Signatures (LINCS) program, `http://lincsproject.org/`, funded by the Big Data to Knowledge (BD2K) Initiative at the National Institutes of Health (NIH). The aim of this program is to generate genetic and molecular signatures of cells in response to various perturbations. This program includes gene expression, protein-protein interaction, and cellular imaging data (Vempati et al., 2014). These data allow researchers to gain new insights into cellular processes. For example, (Vidović et al., 2013) used the LINCS data to understand drug action at the systems level, and (Shao et al., 2013) studied kinase inhibitor induced pathway signatures. Both (Chen et al., 2015) and (Liu et al., 2015) looked at associating chemical compounds with gene expression profiles.

One thrust of this program is the large-scale generation of gene expression profiles using L1000 technology. The LINCS L1000 data is a vast library of gene expression profiles that include over one million experiments covering more than seventy human cell lines. These cell lines are populations of cells descended from an original source cell and having the same genetic makeup, kept alive by growing them in a culture separate from their original source. The L1000 data include experiments using over 20,000 chemical perturbagens, namely drugs added to the cell culture to induce changes in the gene expression profile. In addition, there are genetic perturbation experiments targeting a single gene to control its expression level, either suppressing it (knockdown) or enhancing it (overexpression). The LINCS L1000 data is publicly available for download from `http://lincscloud.org` and from the Gene Expression Omnibus (GEO) database with accession number GSE70138 `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE70138`. An interactive tool for exploring the L1000 data is available at `http://www.maayanlab.net/LINCS/LCB/` (Duan et al., 2014).

Each of the L1000 experiments measures the expression levels of 1000 landmark genes in the human genome. These genes were selected specifically to cover as much of the gene expression variation in cellular expression as possible, since all 20,000+ genes cannot be measured. These experiments have measured cellular response to more than 20,000 different chemical perturbagens. In addition, knockdown and over-expression experiments, where a single gene is targeted to control its expression level, have been performed on thousands of individual genes, both within and outside of the 1000 landmark genes.

The L1000 experiments were performed using Luminex Bead technology (Dunbar, 2006), in which color-coded microspheres were coded to attach to specific RNA sequences corresponding to a landmark gene and fluoresce according to the level of that gene's expression. Sets of beads for measuring the 1000 landmark genes were added to the solution for a single experiment along with the perturbing agent. The experiment was left for a specified period of time and then the gene expression levels were measured.

Experiments were done in sets on a single plate having individual wells for 384 experiments. This minimizes some external sources of error, such as environmental conditions,

across these experiments. A small set of these experiments were used as controls with no perturbation. This gives a baseline distribution of expression level for each gene from which to measure deviations in other experiments. A common approach in this setup is to look at deviations in the perturbation experiments from the controls on the same plate, again recognizing that experiments on the same plate are likely to be more similar than those on different plates. Multiple plates, typically three or four, are prepared and analyzed together as a batch. These plates are prepared as technical replicates, with a given perturbation being prepared and then put into the same well of each plate. This gives additional power in removing systematic biases. Any given perturbation also is performed in multiple different batches, resulting in biological replicates since the sets of experiments were not prepared together.

3.3 Method - Posterior Probability

I want to use the L1000 data to infer gene regulatory networks. This means that I need a method for inferring causality. One way to do this would be to use a causal time-series model, but the L1000 data include a very small number of time points (drug perturbation experiments include only one to two time points). Instead, I use knockdown experiments to identify a single gene as a putative causal agent. Although this limits the amount of information I can gain from a single experiment, it allows us to use a straightforward model with a clearly defined regulatory gene.

When looking at the knockdown experiments, it is important to understand that not all experiments are equally useful. The efficacy of the perturbation varies between target genes and even between experiments for the same target. The experimental setup of the LINCS data is helpful in identifying these differences. I use the control experiments on a plate to get an estimate of the normal variability of a gene. This eliminates some of the variability due to effects I cannot control or even measure, including differences in experimental conditions such as ambient temperature and the scientist performing the experiment, since these are captured in plate-level effects.

To take advantage of this aspect of the LINCS data, I calculate plate-level z-values for each gene in a knockdown experiment. To do this, I first calculate the baseline mean, $\bar{x}_{hp}$, and standard deviation, s_{hp}, for each gene h across all control experiments on plate p. Then the z-value for gene h in knockdown experiment i on plate p is

$$x^*_{hi} = \frac{x_{hi} - \bar{x}_{hp}}{s_{hp}}.$$

Once I have transformed the data in this way, I use a simple linear regression model to model the change in a target gene t as dependent on the change in the knockdown gene h:

$$x^*_{ti} = \beta_0 + \beta_1 x^*_{hi} + \epsilon_i, \quad i = 1, \ldots, n_h \tag{3.1}$$

$$\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2). \tag{3.2}$$

Here, n_h is the number of available knockdown experiments for gene h. This model specifies a linear relationship between the z-score of the knockdown gene h and that of the target gene t. This is a simplification of the true process underlying the relationship between genes h and t, but it can still be effective for discovering relationships.

I estimate this model with a Bayesian approach using Zellner's g-prior (Zellner, 1986) for the model parameters $(\beta_0, \beta_1, \sigma^2)$.

$$\begin{aligned} \beta_1 | \sigma^2, g &\sim N\left(0, \frac{g\sigma^2}{\sum_i x^{*2}_{hi}}\right), \\ \Pr(\beta_0, \sigma^2) &\propto 1/\sigma^2. \end{aligned}$$

The parameter g specifies the expected size of the regression parameter β_1 relative to the standard error of the OLS estimate of β_1. The choice $g = 1$ indicates that the regression parameter is expected to be nearly indistinguishable from the noise, and thus gives a lower bound for g. Also, n_h/g is the effective number of data points in the prior, with $g = n_h$ corresponding to a unit information prior and giving similar results to BIC. I do not want a prior that has more spread than a unit information prior (Raftery, 1999), and thus I choose g in the range $1 \leq g \leq n_h$. In this case, I used $g = \sqrt{n_h}$. I have found this to be a good

compromise between the extremes. I found that when I estimated g using an Expectation-Maximization algorithm (Dempster et al., 1977; Young et al., 2014), the estimated value was often close to $\sqrt{n_h}$.

The regression model with the g-prior allows us to quickly calculate the posterior probability that gene h regulates gene t (Clyde and George, 2004). I first calculate the ratio of the likelihood that h regulates t given the data $\mathbf{x}$, $\Pr(h \to t|\mathbf{x})$, versus the likelihood that there is no regulatory relationship, $\Pr_0$. Further, I can incorporate a prior probability of regulation, π_{ht}, which reflects prior information regarding a regulatory relationship between genes h and t. This gives us

$$T_{ht} \equiv \frac{\Pr(h \to t|\mathbf{x})}{\Pr_0} = \frac{\pi_{ht}}{1-\pi_{ht}} \exp\left[(n_h - 2)\log(1+g)/2 - (n_h - 1)\log(1 + g(1-R^2))/2\right],$$

where R^2 is the coefficient of determination for the simple linear regression model (5.2). From this I can get the posterior probability that h regulates t, or posterior edge probability:

$$p_{ht} = \frac{\Pr(h \to t|\mathbf{x})}{\Pr(h \to t|\mathbf{x}) + \Pr_0} = \frac{T_{ht}}{1 + T_{ht}}.$$

I use this posterior edge probability to rank potential edges and find likely edges for further investigation.

Two advantages of this method are its speed and simplicity. To compute the z-scores, I first get plate-level means and standard deviations, which can be done in a single read through the baseline data by keeping track of sums and sums of squares. From there, standardization of the knockdown data is quick and I need only to calculate correlations between the knockdown gene and each other gene to get the posterior edge probabilities. Additionally, including external knowledge through the prior edge probability can provide a significant boost in accuracy and precision (Young et al., 2014). Finally, these posterior probabilities have a straightforward interpretation, namely the probability that a given regulator-target pair is a true relationship given the data.

3.4 Results

I computed posterior probabilities for edges on the LINCS data for cell line A375. Cell line A375 is a human skin melanoma cell line with over 100,000 experiments in the LINCS data. That includes approximately 15,000 knockdown experiments on landmark genes. This gives a good set of data to evaluate my method. I set the prior probability of any edge being present to 0.0005, reflecting the average expected number of regulators (parents) for each node determined by Guelzim et al. (Guelzim et al., 2002) for yeast and the assumption that transcription factors will regulate approximately the same proportion of target genes regardless of the total number of genes available.

3.4.1 TRANSFAC & JASPAR Assessment Dataset

To assess the results, I need a reference standard. In this case, I looked at the Enrichr website, `http://amp.pharm.mssm.edu/Enrichr/` (Chen et al., 2013a), which has collected numerous gene-set libraries, including some that list gene regulatory relationships. I used the TRANSFAC and JASPAR lists of edges; these list transcription factors as well as putative binding sites on other genes using a position weight matrix (Wingender et al., 2000; Sandelin et al., 2004). This is not a comprehensive gold standard for assessment since these regulatory relationships are limited to well-studied transcription factors. However, assessment of gene networks in the mammalian systems is non-trivial due to incomplete knowledge.

The TRANSFAC and JASPAR (T&J) dataset includes 37 transcription factors that overlap with the LINCS landmark genes. Thus I limit my assessment to only those genes as potential regulators. For these 37 transcription factors, the T&J dataset has 4,303 regulation-target pairs where both the transcription factor and the target are among the landmark genes. For each transcription factor in T&J, there are a variable number of knockdown experiments in the L1000 data targeting that particular gene, generally between 10 and 20 per gene. These are used to calculate posterior probabilities for about 42,000 pairs of genes to be compared with the T&J reference dataset.

3.4.2 Comparison methods

To further evaluate the method, I compared the results with results from applying Significance Analysis of Microarrays (SAM) (Tusher et al., 2001) and mutual information methods to the data. SAM is an adaptable method for identifying significant changes in gene expression level while estimating the false discovery rate. It is widely used to evaluate microarray data and is available as an `R` package, `samr` (Tusher et al., 2001). Mutual information methods, based in information theory, have also been used extensively to identify relationships among genes. I used the `minet` package in `R` (Meyer et al., 2008) to analyze the data with three different mutual information methods: Context Likelihood of Relatedness (CLR) (Faith et al., 2007), Algorithm for the Reconstruction of Accurate Cellular Networks (ARACNE) (Margolin et al., 2006), and minimum redundancy - maximum relevance (MRMR) (Ding and Peng, 2005; Meyer et al., 2007). These three methods offer differing approaches for identifying relationships between genes.

Each method produces a list of gene pairs along with some measure of the strength of their relationship. SAM returns p-values for each relationship, the mutual information methods produce weights indicating the strength of the relationship, and my method gives posterior probabilities. I can sort these to produce a ranked list and evaluate these lists against the reference dataset.

3.4.3 2x2 tables

First, I looked at two-by-two tables from each method. For the posterior probability method, I used probability cutoffs of 0.5 and 0.95 to define found edges. SAM provides a list of relationships found to be significant. The mutual information methods do not define a particular cutoff for significance, and so all relationships returned with non-zero weight were included. These two-by-two tables are produced in Table 3.1.

To assess whether the lists and the T&J dataset are related, I also report approximate (non-Bayesian) p-values by using the probability of getting at least the number of true

positives found using a binomial(n, p) distribution, where n is the number of pairs in the inferred list and p is the probability of selecting a true edge from the total number of possible edges. Note that these p-values are not the same as the posterior probability thresholds, and inference is based on the posterior probability thresholds. The p-values are equal to 0.02 for both thresholds, indicating that the posterior edge probabilities are related to the T&J results at conventional levels of significance. The competing methods are not able to accurately identify a small number of edges as true, returning many more than the posterior probability method.

Table 3.1: Assessment results showing 2x2 tables for cell line A375 using knockdown experiments for finding edges via posterior probability calculation and compared to approximately 4,200 edges from TRANSFAC and JASPAR across 37 transcription factors. PP indicates the posterior probability method at a certain threshold. When looking at edges with posterior probability of 0.5 or greater (top left table), 41 of the 292 candidate edges are found in TRANSFAC and JASPAR, and 14 of the 76 candidate edges at a cutoff of 0.95 (top center table) are true edges. Binomial approximate p-values are 0.02 (left) and 0.02 (right). Lower p-values indicate better results. The competing methods return many more edges as true but are not as precise, resulting in higher p-values.

PP 0.5

		Yes	No
T&J	Yes	41	4262
	No	251	37566

p-value: 0.02

PP 0.95

		Yes	No
T&J	Yes	14	4289
	No	62	37755

p-value: 0.02

SAM

		Yes	No
T&J	Yes	1193	3110
	No	11151	26666

p-value: 0.98

CLR

		Yes	No
T&J	Yes	1651	2652
	No	14671	23146

p-value: 0.67

ARACNE

		Yes	No
T&J	Yes	34	4269
	No	910	36907

p-value: 1.00

MRMR

		Yes	No
T&J	Yes	1530	2773
	No	13533	24284

p-value: 0.60

I expect errors from the assessment results in the form of both false postives and false negatives due not only to limitations of this method, but also due to the nature of the data and the TRANSFAC and JASPAR reference standard by which I evaluate the edges. This is in part because the T&J reference standard is not specific to a certain cell line and the relationships inferred could be specific to cell line A375. Additionally, false positives might arise because the expression levels of target genes change due to indirect effects. The path from the transcription factor to the target gene may go through intermediate genes. In fact, since only about 5% of the human genes are measured by the LINCS experiments, there are likely to be many genes in relevant pathways that are not measured. If I had measurements for all 22,000 genes, using a link removal procedure could be very useful (Klamt et al., 2010; Pinna et al., 2010). I also expect false negatives since the T&J dataset is not a set of confirmed regulatory relationships. Rather, it is informed from attributes of the transcription factor as well as the target gene. This means that many of the true relationships as designated by the T&J dataset may not in fact reflect true interaction at the cellular level. In general, I do not expect a transcription factor to affect 10% of the possible targets, which is what the T&J dataset reports, so it is likely that the T&J data is overestimating the number of regulatory relationships.

3.4.4 Precision-recall curves

Another way of looking at the results is via the precision-recall curve. Precision and recall are both calculated by truncating the ranked list of edges and looking only at those proposed edges. Precision is the proportion of the proposed edges which are true edges. Recall is the proportion of true edges which are in the proposed set. The precision-recall curve takes a ranked list of edges from a procedure and shows how the precision varies as more and more edges are included from that list. High precision at low recall indicates that the procedure is good at identifying true edges at the highest probability. This is important in many cases, particularly genetic studies, because it gives researchers good information on where to focus their efforts in subsequent studies.

Figure 3.1 shows the precision-recall curves generated by the different methods. I do see that the edges most highly ranked by posterior probability yield better results than expected from random guessing by a factor of 1.5 to 2. The precision declines as I add more edges, returning to hover near random guessing. The MRMR and ARACNE results fare worse than random guessing, and although CLR ranks a few true edges highly, it returns to random much faster than the posterior probability edges. The ranked list from SAM performs comparably to the posterior probability method, but it is unable to differentiate between the edges at the very top of its list, with 168 edges yielding the same lowest p-value. From a scientific point of view, it is important to have high precision among the edges ranked most highly, since there are limited resources for designing and executing experiments investigating particular edges more closely. Of course it would be preferable to see even better precision, but previous discussion has shown why that may not be achievable with this dataset and standard.

3.5 Discussion

I have demonstrated a straightforward approach to inferring gene regulatory network edges from knockdown data. This approach is simple to apply to large datasets and includes the ability to incorporate prior information when available. This approach is able to find confirmable regulatory relationships between genes from the L1000 data. I showed that my method performs comparably to or better than popular approaches for identifying important regulatory relationships as found in the TRANSFAC & JASPAR evaluation dataset.

One key benefit of this approach is that it can be applied to extremely large datasets, requiring only one read through the data to compile all sufficient statistics for computing the posterior probabilities. There is no need to retain all the data after reading it and no iterative methods, such as Expectation-Maximization or Markov Chain Monte Carlo, are used. Methods which model the entire network (Lèbre et al., 2010; Scutari, 2010; McGeachie et al., 2014; Sanchez-Castillo et al., 2013) may yield more comprehensive network results but are also generally restricted to a smaller set of genes due to computational constraints. My approach is complementary to these other approaches in potentially narrowing down the set

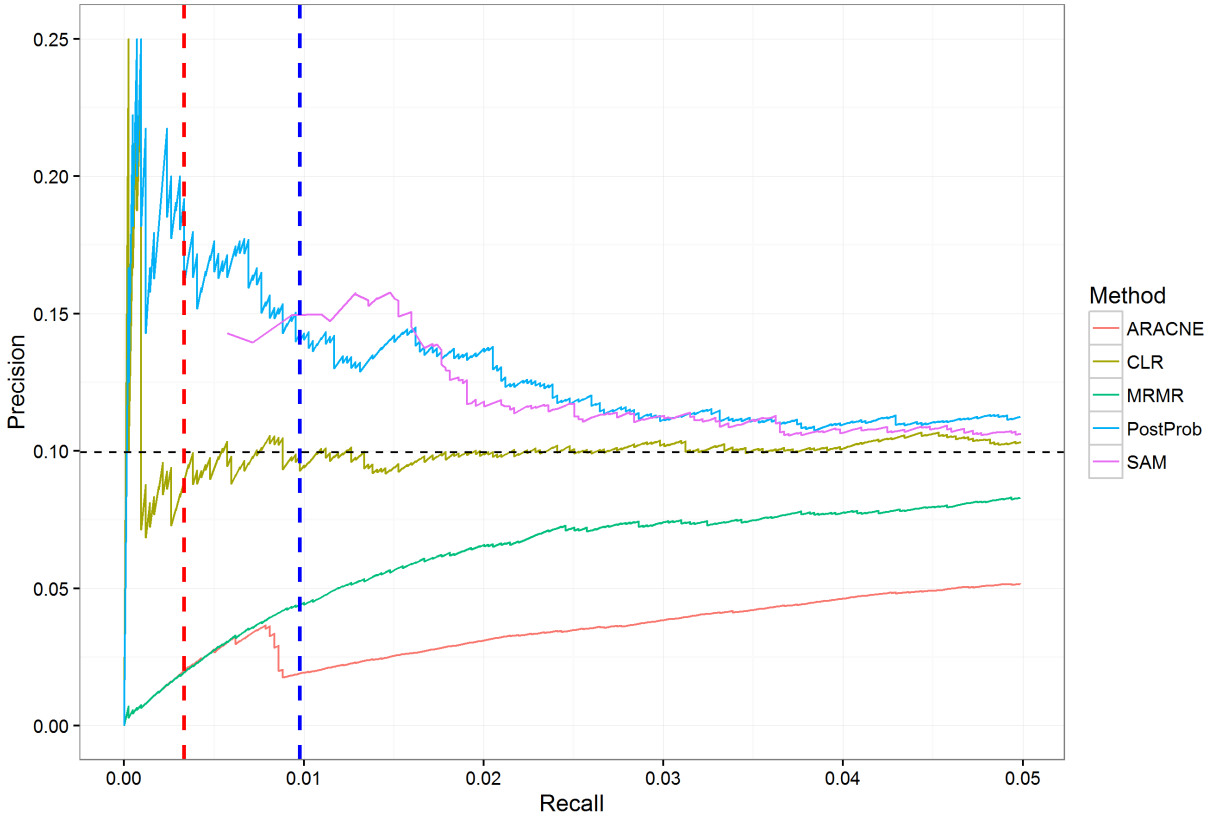

Figure 3.1: Precision-recall curves for cell line A375 using knockdown experiments for finding edges via posterior probability calculation evaluated against approximately 4,200 edges from TRANSFAC and JASPAR across 37 transcription factors. To the left of the red line are those edges with a posterior probability of at least 0.95, while the blue line shows the cutoff for edges with a posterior probability of 0.5. The horizontal dashed line shows the expected precision of 0.1 that would result from randomly ranking edges.

of edges for further investigation.

I have applied the method to knockdown data in order to identify causal regulatory relationships. This method can also be applied to over-expression data or even steady-state data, although for steady-state data the resulting edges would lack directionality (Michailidis and d'Alché Buc, 2013). This method could also be used to infer differential expression for a perturbation such as a drug treatment, as in Grechkin et al. (2016). This could be done using a 2-class model where the predictor variable indicates whether the expression measurements

come from a perturbation experiment or a control experiment. An implementation of my method is currently available as an `R` package, `BayesKnockdown`, in the beta branch of Bioconductor (Gentleman et al., 2004), including functions for both knockdown and perturbaion data. The package will be included in the next general release of Bioconductor.

I considered using an edge reduction technique, such as that used by Pinna et al. (Pinna et al., 2010), but ultimately decided against it. Since I only used 37 genes as regulators due to the assessment data, the resulting networks did not tend to have multiple pathways from one gene to another. In cases where the resulting networks are much more rich in multi-gene pathways, using an edge reduction method may be appropriate.

Another possible use of this method is to use the resulting edge probabilities as an informed prior for another method utilizing a different type of data. This allows the integration of multiple data sources and may increase the usefulness of knockdown data that may expected to only provide a small amount of evidence within a larger experimental context.

Chapter 4

MODEL-BASED CLUSTERING WITH DATA CORRECTION (MCDC)

4.1 Motivation - Artifacts in L1000 Data

The performance of any method is limited by the quality of the data being used. This is true for data from gene expression experiments due to a number of factors, from variability in environmental conditions to uncertainties inherent in the measurement technologies themselves (Liu and Rattray, 2010). The data used for inference have usually gone through a preprocessing pipeline to adjust the data to be more amenable to analysis (Sebastiani et al., 2003). Examples of preprocessing steps include log transformation of raw fluorescence values and quantile normalization. Although these techniques are often helpful, they can sometimes introduce artifacts into the data (Blocker et al., 2013; Lehmann et al., 2013). It is important to identify these additional sources of variation and correct them if possible, or if not to account for them in the assessment of variability and uncertainty.

This chapter is motivated by the L1000 gene expression dataset from the NIH Library of Integrated Network-based Cellular Signatures (LINCS) program. In this dataset, genes are paired in the experimental setup and this leads to multiple issues in the processed data, including clustering, switched expression values of the two genes, and assignment of the same expression value to the two genes. I develop a new method to fix these issues. The method is an extension of model-based clustering that explicitly incorporates the expression level swaps while simultaneously addressing the other problems in the data. I call it model-based clustering with data correction, or MCDC. I show that this method works well on simulated datasets. I also show that it improves the gene expression data, both in terms of agreement with an external baseline and in subsequent inference.

Section 4.2 describes the motivating data for this method. Section 4.3 outlines the method, MCDC, as well as a practical EM algorithm for implementation. In Section 4.4 I present a simulation study, showing the the method is able to identify and correct points which have been altered. Section 4.5 shows how MCDC can be applied to the motivating data to improve the data overall as well as improve subsequent analyses. Section 4.6 concludes with a discussion.

4.2 Data

4.2.1 Experimental design of the L1000 data

Each individual L1000 experiment measures the expression levels of approximately 1,000 landmark genes in the human genome. The goal of the LINCS project is to capture the cells' response to perturbations. Therefore, the project was designed to include a very large number of experiments, but this came at the cost of measuring only a limited number of selected landmark genes. These landmark genes were selected to cover as much of the variation in cellular gene expression as possible. In each experiment, the selected perturbation was applied and the cells were allowed to culture for a specified period of time before the gene expression levels were measured.

The L1000 experiments were performed using the Luminex Bead technology (Peck et al., 2006; Dunbar, 2006), in which color-coded microspheres are produced to attach to specific RNA sequences corresponding to a landmark gene and fluoresce according to the amount of RNA produced as that gene is expressed. To perform a single experiment, a perturbing agent such as a chemical compound is added to the solution. Additionally, around 100 beads are added for each gene to be measured. The beads for measuring a particular gene share a color that can be uniquely identified using lasers. To process an experiment, the beads in the solution are sampled and analyzed to determine which gene they are measuring. Additionally, their fluorescence level is measured to determine the expression level of the gene. With many beads per gene, a good estimate of the overall expression level is obtained.

The L1000 experiments used only 500 bead colors to measure the expression levels of the 1,000 landmark genes. This means that each bead color had to do double duty, accounting for a pair of genes. Thus, when an experiment is processed, a given bead color will have some observations from one gene and the rest from another gene. These gene pairs were selected to have different levels of expression, and the beads for a pair were mixed in approximately a 2:1 ratio. This means that, ideally, when the beads are sampled a histogram of fluorescence levels corresponding to gene expression is created with two peaks, one of which has twice the number of observations as the second peak.

4.2.2 L1000 Data Preprocessing

In order to facilitate statistical analysis on the L1000 data, the raw bead fluorescence measurements are combined and transformed. First, the measurements from many beads of the same color were deconvolved to assign expression values to the appropriate pair of genes. The data then went through multiple normalization steps (Liu et al., 2015; Bolstad et al., 2003). First, a set of genes were identified as being stable across cell lines and perturbations, and these were used to inform a power law transformation of all gene values. The expression values were then quantile-normalized across sets of experiments to make the distribution of expression levels the same for all experiments. These steps are illustrated in Figure 4.1.

Although these data processing steps result in data that are more amenable to statistical analysis, I have found that the deconvolution step in particular introduces artifacts in the data. This can be seen when looking at multiple experiments on the same cell line with the same experimental conditions. If we look at a pair of genes that share a bead color and form a scatterplot of their values across many experiments, we see that these artifacts can take several forms.

First of all, two genes that are paired on the same bead color may not be expressed at levels such that they are easily distinguished. This can lead to assigning both genes the same value, resulting in a clustering of data directly on the $x = y$ diagonal. Secondly, the deconvolution step, which uses a simple k-means algorithm, can be misled if there are

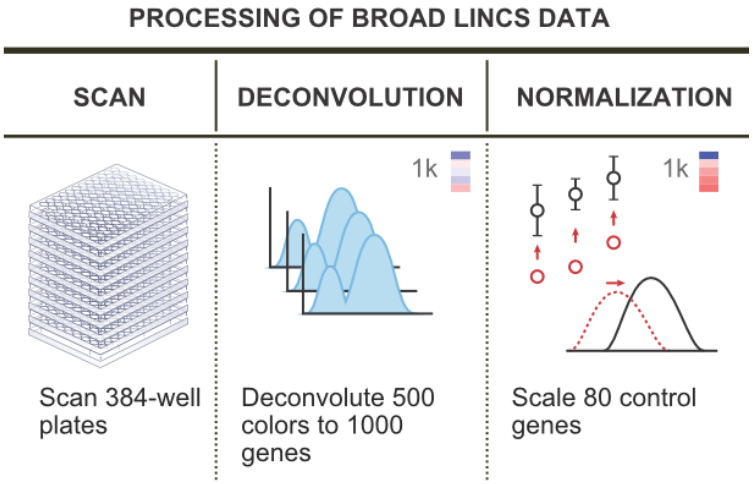

Figure 4.1: The L1000 data preprocessing pipeline. The raw data are first measured from the beads in the experiments. Next, the data from each color of bead are deconvolved to assign expression values to the two genes which share that bead color. Finally, the data are normalized to yield directly comparable data across experiments. Figure adapted from an image on the Broad Institute LINCS cloud website (http://lincscloud.org/l1000).

many beads sampled with very low fluorescence values. This, combined with the quantile normalization step, can lead to additional clusters that are not at the true expression value. Finally, the deconvolution step can result in assigning the expression levels of the genes incorrectly. That is, the expression level of gene A of the pair on the same bead color is sometimes assigned to gene B instead, and vice versa. Figure 4.2 shows examples of the raw bead data of two gene pairs and illustrates the difficulty of the deconvolution step.

Figure 4.3 shows examples of these three types of artifact in the L1000 data. The figure shows the expression values for two paired genes, CTLC and IKZF1. Each point shows the values measured in a single experiment. All 630 experiments in this dataset are on the same cell line, A375, and are untreated, used as controls. As such, one would expect a single cloud of observations centered around the point defined by the true expression values of the two genes. Instead, several clusters of observations are present, as well as points lying on or very near the diagonal. Note in particular the two circled sets of points. These appear to be a single cluster in which some of the points were flipped, with the expression values assigned

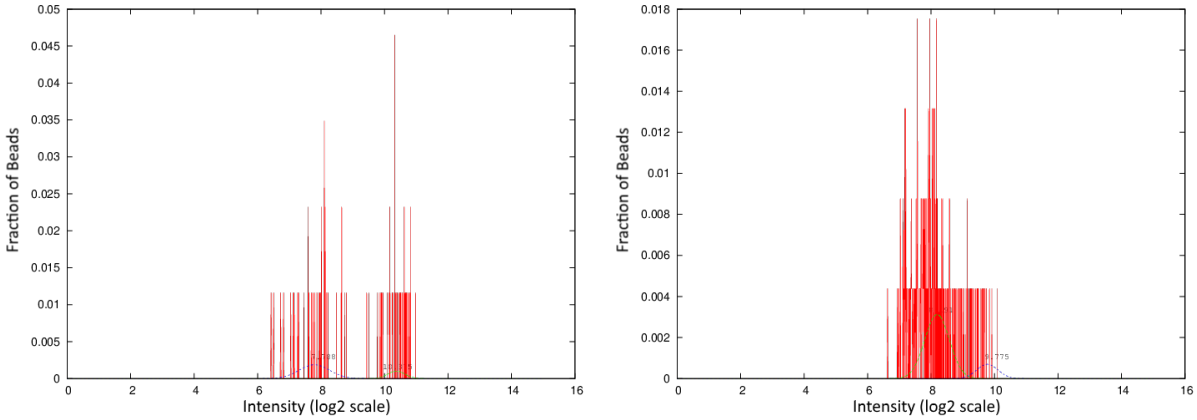

Figure 4.2: Example histograms of raw bead fluorescence values for single bead colors. The plot on the left shows an example where the two peaks corresponding to the genes sharing the bead color are relatively easy to distinguish. The right plot shows an example where the deconvolution is difficult. Dotted lines show possible inferred densities for two clusters.

to the wrong genes. If one set of points is flipped across the diagonal, it falls directly on the other set.

Looking more broadly at the untreated A375 experiments, artifacts such as those in Figure 4.3 occur across gene pairs. Every pair of genes has at least a few points lying on or near the diagonal, with an average of about 30 per pair. Additionally, in comparing the number of points on either side of the diagonal, excluding those close to it, half of the gene pairs have at least 10% of the points on the wrong side of the diagonal (the side with fewer points), and a quarter of the pairs have at least 25% on the wrong side. This may be partially explained by gene pairs that have similar expression values, but the gene pairs were initially selected so as to avoid that problem.

4.3 Method

I propose a method to detect and correct all three kinds of artifact present in the LINCS L1000 data: the multiple clusters introduced by the preprocessing pipeline, the erroneous assignment of the same expression value to paired genes, and flipping of the expression levels

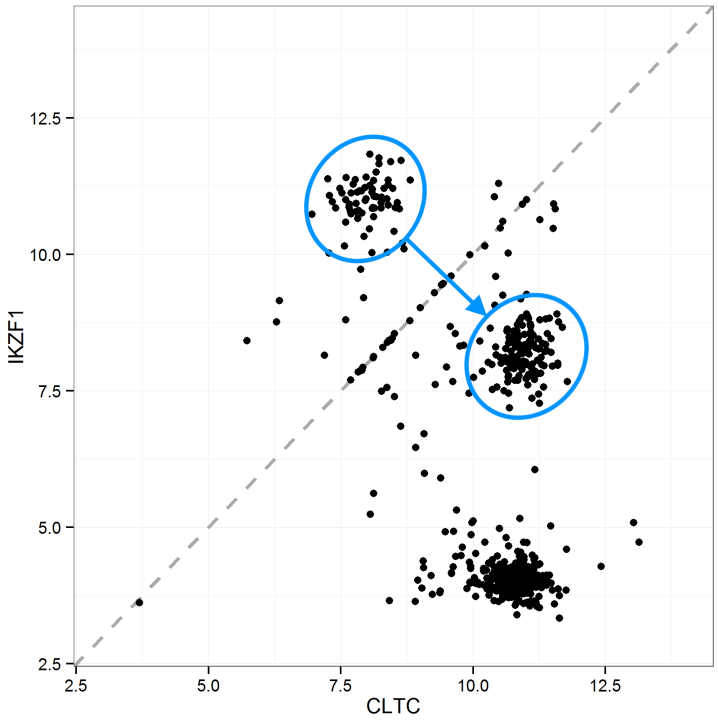

Figure 4.3: Expression levels on a log-base-2 scale for two paired genes, CLTC and IKZF1, measured on the same bead, in the L1000 data. Each point represents one experiment of 630. Data artifacts include points directly on or very near to the diagonal, multiple clusters rather than a single one as may be expected, and flipping between the two circled clusters of points, with the CLTC value incorrectly assigned to IKZF1, and vice versa. The blue arrow shows the effect of data correction using MCDC.

of paired genes. This improves the quality of the data and leads to better downstream analysis. I do this by addressing issues present in the preprocessed data rather than reprocessing the raw data, as in Liu et al. (2015). I adopt this approach because some of the artifacts are likely to persist even if the deconvolution method is improved for individual gene pairs in isolation.

This method is an extension of model-based clustering (Wolfe, 1970; Banfield and Raftery, 1993; McLachlan and Peel, 2000; Fraley and Raftery, 2002), which is a model-based method for finding clusters by fitting a multivariate Gaussian mixture model to the data. It has

been found useful in many different contexts, including geochemical analysis (Templ et al., 2008), chemometrics (Fraley and Raftery, 2007), and health studies (Flynt and Daepp, 2015). This method is well adapted to estimating the expression levels because sometimes there are small groups of points not in the main cloud around the true value, such as the points on the diagonal in Figure 4.3. Model-based clustering can identify these groups as clusters and remove or downweight them, thus preventing them from contaminating the estimation of the gene expression levels.

However, while model-based clustering is able to identify the clusters as well as identifying outliers, it does not have a mechanism for identifying particular points as flipped. Here I extend the model-based clustering method to detect and take into account the flipping in the data. More generally, it can be used for data with any invertible transformation applied to a subset of the data. This extension allows the use of an Expectation-Maximization (EM) algorithm commonly used to estimate finite mixture models (Dempster et al., 1977; McLachlan and Krishnan, 2007).

4.3.1 Model

Suppose multivariate data, $\{\mathbf{x}_i : \ i = 1, \ldots, N\}$, is observed, generated by a finite mixture of G distributions f_k, $k = 1, \ldots, G$ with probabilities $\tau_1, \ldots, \tau_G$:

$$f(\mathbf{x}) = \sum_{k=1}^{G} \tau_k f_k(\mathbf{x}|\theta_k).$$

Suppose further that $\mathbf{x}_i$ is not observed, but rather $\mathbf{y}_i$, a possibly transformed version of $\mathbf{x}_i$, where the probability of a data point having been transformed can depend on the mixture component k that $\mathbf{x}_i$ is drawn from:

$$\mathbf{y}_i = \begin{cases} \mathbf{x}_i & \text{with probability } \pi_k, \\ \mathbf{T}\mathbf{x}_i & \text{with probability } (1 - \pi_k). \end{cases}$$

Here, $\mathbf{T}$ is any invertible transformation that preserves the domain of $\mathbf{x}$. Often, this may be represented as a matrix, but it may also be a functional transformation (i.e. a component-

wise monotonic transformation). In the case of the L1000 data, this is just the 2×2 matrix with zeros on the diagonals and ones on the off-diagonals, switching the two values.

Given the transformation $\mathbf{T}$, the distribution of $\mathbf{y}_i$ can be written as follows:

$$f(\mathbf{y}_i|\boldsymbol{\tau}, \boldsymbol{\pi}, \boldsymbol{\theta}) = \sum_{k=1}^{G} \tau_k \left[\pi_k f_k(\mathbf{y}_i|\theta_k) + (1-\pi_k) f_k(\mathbf{T}^{-1}\mathbf{y}_i|\theta_k)\right].$$

To simplify the notation, define $f_{ik} \equiv f_k(\mathbf{y}_i|\theta_k)$ and $f_{ik}^{-} \equiv f_k(\mathbf{T}^{-1}\mathbf{y}_i|\theta_k)$. Then the distribution of $\mathbf{y}_i$ can be written

$$f(\mathbf{y}|\boldsymbol{\tau}, \boldsymbol{\pi}, \boldsymbol{\theta}) = \prod_{i=1}^{n} \sum_{k=1}^{G} \tau_k \left[\pi_k f_{ik} + (1-\pi_k) f_{ik}^{-}\right].$$

4.3.2 EM Algorithm

This model is estimated by maximum likelihood using the EM algorithm. One can formulate this as a missing data problem where the complete data are $\{\mathbf{y}_i, \mathbf{z}_i, \xi_i\}$. Here, $\mathbf{z}_i = (z_{i1}, \ldots, z_{iG})$ and ξ_i are unobserved labels, with

$$z_{ik} = \begin{cases} 1 & \text{if } \mathbf{y_i} \text{ belongs to group } k, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\xi_i = \begin{cases} 0 & \text{if } \mathbf{y_i} \text{ has been transformed,} \\ 1 & \text{otherwise.} \end{cases}$$

Then the complete-data log-likelihood is

$$\ell(\boldsymbol{\theta}, \boldsymbol{\tau}, \mathbf{z}, \boldsymbol{\pi}, \boldsymbol{\xi}|\mathbf{y}) = \sum_{i=1}^{n} \sum_{k=1}^{G} z_{ik} \left[\, \xi_i \log\left(\pi_k \tau_k f_{ik}\right) + (1-\xi_i) \log\left((1-\pi_k)\tau_k f_{ik}^{-}\right)\right].$$

The joint distribution of $\mathbf{z}_i$ and $\boldsymbol{\xi}_i$ given $\mathbf{y}_i$ and $\boldsymbol{\theta}$ can be written as:

$$f(\mathbf{z}_i, \boldsymbol{\xi}_i|\mathbf{y}_i, \boldsymbol{\theta}) = \frac{1}{f(\mathbf{y}_i|\boldsymbol{\tau}, \boldsymbol{\pi}, \boldsymbol{\theta})} \prod_{k=1}^{G} \left[(\tau_k \pi_k f_{ik})^{\xi_i} \cdot (\tau_k(1-\pi_k) f_{ik}^{-})^{(1-\xi_i)}\right]^{z_{ik}}. \tag{4.1}$$

For the E-step of the algorithm, first calculate the expected complete-data log-likelihood, namely

$$\begin{aligned} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^*) &= E[\ell(\boldsymbol{\theta}, \boldsymbol{\tau}, \mathbf{z}, \boldsymbol{\pi}, \boldsymbol{\xi}|\mathbf{y})|\mathbf{y}, \boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*], \\ &= \sum_{i=1}^{n}\sum_{k=1}^{G} E[z_{ik}\xi_i|\mathbf{y}, \boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*] \log(\pi_k \tau_k f_{ik}) + \\ &\quad E[z_{ik}(1-\xi_i)|\mathbf{y}, \boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*] \log\left((1-\pi_k)\tau_k f_{ik}^-\right). \end{aligned}$$

From Equation (4.1),

$$\begin{aligned} E[z_{ik}\xi_i|\mathbf{y}, \boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*] &= \frac{\tau_k^* \pi_k^* f_{ik}}{f(\mathbf{y}_i|\boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*)}, \\ E[z_{ik}(1-\xi_i)|\mathbf{y}, \boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*] &= \frac{\tau_k^* (1-\pi_k^*) f_{ik}^-}{f(\mathbf{y}_i|\boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*)}. \end{aligned}$$

$z_{ik}\xi_i + z_{ik}(1-\xi_i) = z_{ik}$ and $\sum_{k=1}^{G} z_{ik} = 1$. This leads to the following updates of the estimates of z_{ik} and ξ_i, which make up the E-step:

$$\begin{aligned} \hat{z}_{ik} &= \frac{\tau_k^*[\pi_k^* f_{ik} + (1 - \pi_k^* f_{ik}^-)]}{f(y_i|\boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*)}, \\ \hat{\xi}_i &= \frac{\sum_{k=1}^{G} \tau_k^* \pi_k^* f_{ik}}{f(y_i|\boldsymbol{\tau}^*, \boldsymbol{\pi}^*, \boldsymbol{\theta}^*)}. \end{aligned}$$

The M-step is then as follows:

$$\begin{aligned} \hat{\tau}_k &\leftarrow \frac{n_k}{n}, \\ \hat{\pi}_k &\leftarrow \frac{\sum_{i=1}^{n} \hat{z}_{ik}\hat{\xi}_i}{n_k}, \\ \hat{\boldsymbol{\mu}}_k &\leftarrow \frac{\sum_{i=1}^{n} \hat{z}_{ik}(\hat{\xi}_i \mathbf{y}_i + (1-\hat{\xi}_i)\mathbf{T}^{-1}\mathbf{y}_i)}{n_k}, \\ n_k &\equiv \sum_{i=1}^{n} \hat{z}_{ik}. \end{aligned}$$

To get the variance of the clusters, follow the steps from Celeux and Govaert (1995), modifying the scattering matrix W_k of cluster k as well as the within-cluster scattering

matrix W to be

$$\begin{aligned} W_k &= \sum_{i=1}^{n} \hat{z}_{ik} \left[\hat{\xi}_i (\mathbf{y}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{y}_i - \hat{\boldsymbol{\mu}}_k)' + (1 - \hat{\xi}_i)(\mathbf{T}^{-1}\mathbf{y}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{T}^{-1}\mathbf{y}_i - \hat{\boldsymbol{\mu}}_k)' \right], \\ W &= \sum_{k=1}^{G} W_k. \end{aligned}$$

These can then be used to calculate Σ_k under different variance models.

Iterate the EM steps until convergence. This leads to a local optimum of the log-likelihood (Wu, 1983). Although this is not guaranteed to be the global optimum, choosing starting values intelligently or doing multiple restarts have been shown to lead to good solutions (Fraley and Raftery, 1998; Biernacki et al., 2003).

The model allows the cluster-specific variance matrices to differ between clusters. I select the best number of clusters by running MCDC with the number of clusters ranging from 1 to some maximum number of clusters (9 in this case) and then comparing the BIC values for the resulting estimated models (Fraley and Raftery, 2002).

For the gene expression data, estimate the expression levels of a pair of genes as the mean of the largest cluster (the cluster with the most points assigned to it) found using the chosen model. This method ignores the artifactual clusters that result from the preprocessing of the data in estimating the expression levels of the genes.

4.4 Simulation Study

I now describe a simulation study in which data with some of the key characteristics of the LINCS L1000 data were simulated. I simulated datasets with no clustering (i.e. one cluster), but where some of the observations were flipped. I also simulated datasets with clustering (two clusters), where some of the observations were flipped.

Finally, I simulated a dataset where no observations were flipped, but instead some observations were rotated and scaled. This is to show that the method can be effective when some of the data are perturbed in ways other than flipping.

4.4.1 Simulation 1: One Cluster With Flipping

Figure 4.4 is an example dataset from the first simulation. This simulation represents what is seen in the LINCS L1000 data in the best case, with no clustering or diagonal values (i.e. a single cluster), but with some flipping. For the simulation, I generated 100 datasets with 300 points each from the single cluster model with flipping probabilities $(1 - \pi)$ of 0.05 to 0.45 in increments of 0.05, resulting in 900 simulated data sets in total. I applied MCDC to each simulated dataset and counted the number of times the correct number of clusters (one) was selected as well as the percentage of the points correctly identified as flipped or not. Finally, I looked at the inferred gene means compared to taking the mean of all the observations without applying MCDC. I refer to this as the unaltered mean.

The correct number of clusters (i.e. one) was selected for all but one of the 900 datasets, and in the one erroneous dataset out of 900 only a few points were in a second cluster. In 858 of the 900 datasets no errors were made – all the points were correctly identified as being flipped or not flipped. In three datasets, all with the highest probability of flipping, namely 0.45, all the points were misidentified by being flipped to the wrong side, while in one dataset (again with $1-\pi = 0.45$), a single large cluster with no flipping was identified. The remaining 38 datasets had one to three points out of 300 misidentified. All these misidentifications make sense, since I expect rare cases where a point crosses the $x = y$ line as well as cases where more points are flipped when using a flipping probability near 0.5.

Figure 4.5 and Table 4.1 show the mean absolute error in inferred mean using MCDC versus the unaltered data. For each flipping probability, I calculated the mean absolute error of the inferred mean from the true mean. MCDC did much better than taking the unaltered mean in all cases, improving on the unaltered data by a factor of 5 to 36, depending on the probability of flipping.

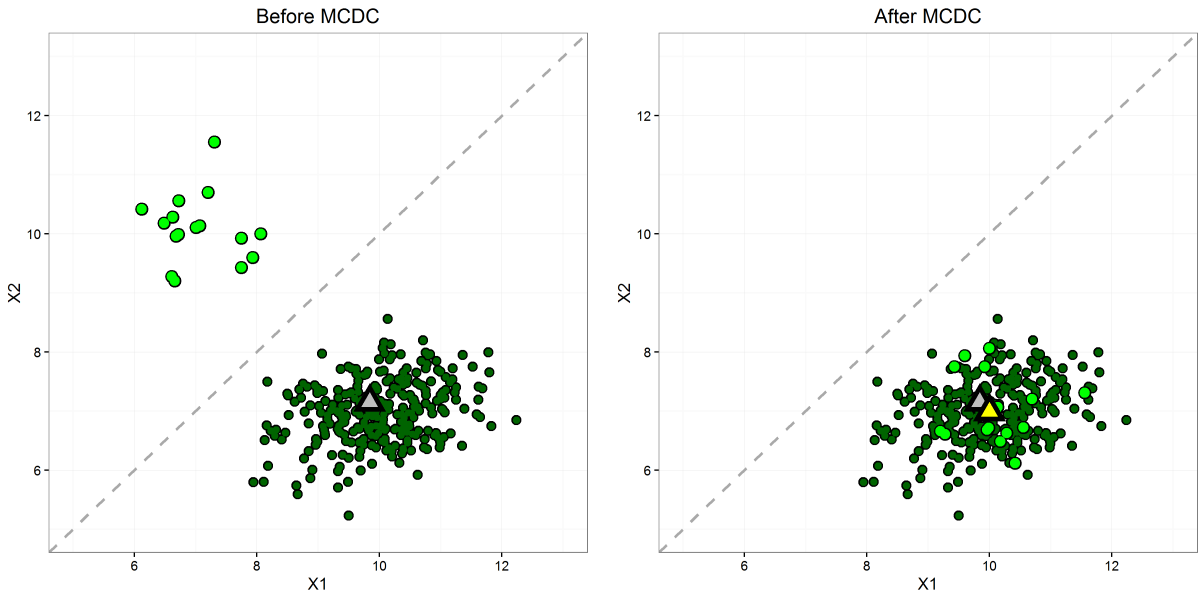

Figure 4.4: One Dataset from Simulation 1: One Cluster with Flipping. $1 - \pi$, the fraction of data flipped, is chosen to be 0.05. Left panel: Original data with flipped data points. Right panel: Data after correction by MCDC. MCDC identified and corrected all the flipped points. The grey triangle is the mean of all the data, and the yellow triangle is the mean of the data after correction by MCDC.

4.4.2 Simulation 2: Two Clusters with Flipping

For the second simulation, I added a second cluster on the diagonal, as demonstrated in Figure 4.6. This reflects a common issue I see in the L1000 data. When the data processing pipeline has trouble differentiating between the two gene expression levels, it can end up assigning them both the same value. For these data, I wanted to see how well MCDC identified the "good" points (not the diagonal cluster). Again, I used the mean of the largest cluster as the inferred mean. For the simulation, I generated 100 datasets with 400 points each for the two-cluster model with flipping probabilities $(1 - \pi)$ of 0.05 to 0.45, and probability τ of a point being in the true cluster 0.55 to 0.95, in increments of 0.05.

Figure 4.7 shows the comparison of mean absolute error in the inferred mean. The MCDC results are better across the board, although as τ decreases, it is more likely to find

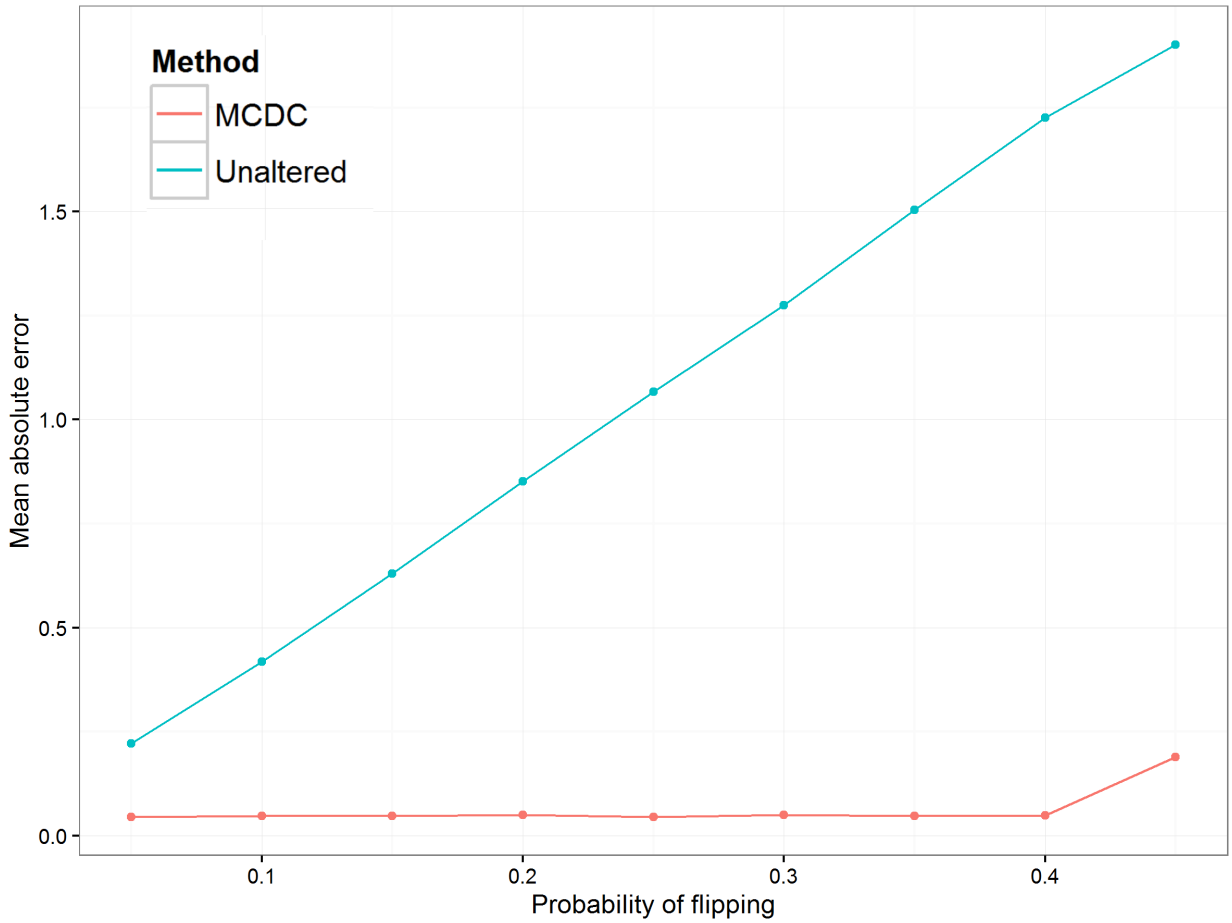

Figure 4.5: Simulation 1: Mean Absolute Error in Inferred Mean. The blue line is using unaltered data, while the red line is using the mean of the largest cluster found by MCDC.

the diagonal cluster as the largest one. Figure 4.8 shows that MCDC does well in identifying which points are flipped or not. In Figure 4.9, the correct number of clusters is not generally identified as well as is preferable. This may be due to poor initialization of the algorithm and may be corrected with multiple random initializations.

4.4.3 Simulation 3: Three Clusters With Rotation and Scaling

To show that MCDC can be applied to other kinds of data errors than flipping, I also generated a dataset with three clusters where the error process rotates and scales the data

Table 4.1: Simulation 1: Mean Absolute Error (MAE) in Inferred Mean for Unaltered data and MCDC-corrected Data, as the probability of flipping increases. The MAE ratio is the ratio of mean absolute error using the unaltered data divided by the MAE using the MCDC-corrected data. Values greater than 1 indicate improvement by using MCDC.

Probability of Flipping	**Unaltered MAE**	**MCDC MAE**	**MAE Ratio**
0.05	0.22	0.04	5
0.10	0.42	0.05	9
0.15	0.63	0.05	13
0.20	0.85	0.05	17
0.25	1.07	0.05	24
0.30	1.27	0.05	25
0.35	1.50	0.05	32
0.40	1.73	0.05	36
0.45	1.90	0.19	10

points affected. This transformation is not motivated by the L1000 data, but rather serves to demonstrate the potential for MCDC to be used in other situations. In this more complex situation, I used $n = 1000$ points split among the three clusters with varying probabilities of transformation. MCDC selected the correct number of clusters and correctly classified all the points. Figure 4.10 shows the original and MCDC-corrected data. One caveat is that here, as in the flipping situation, the data error process was known to the MCDC algorithm.

4.5 Application to LINCS L1000 Data

I applied MCDC to a portion of the LINCS L1000, namely the data from cell line A375, a human skin malignant melanoma cell line. I chose this cell line because it has good coverage in the L1000 data in terms of the number of different perturbations applied. I only considered the transformation $\mathbf{T}$ that switches the expression levels of the paired genes. I looked at

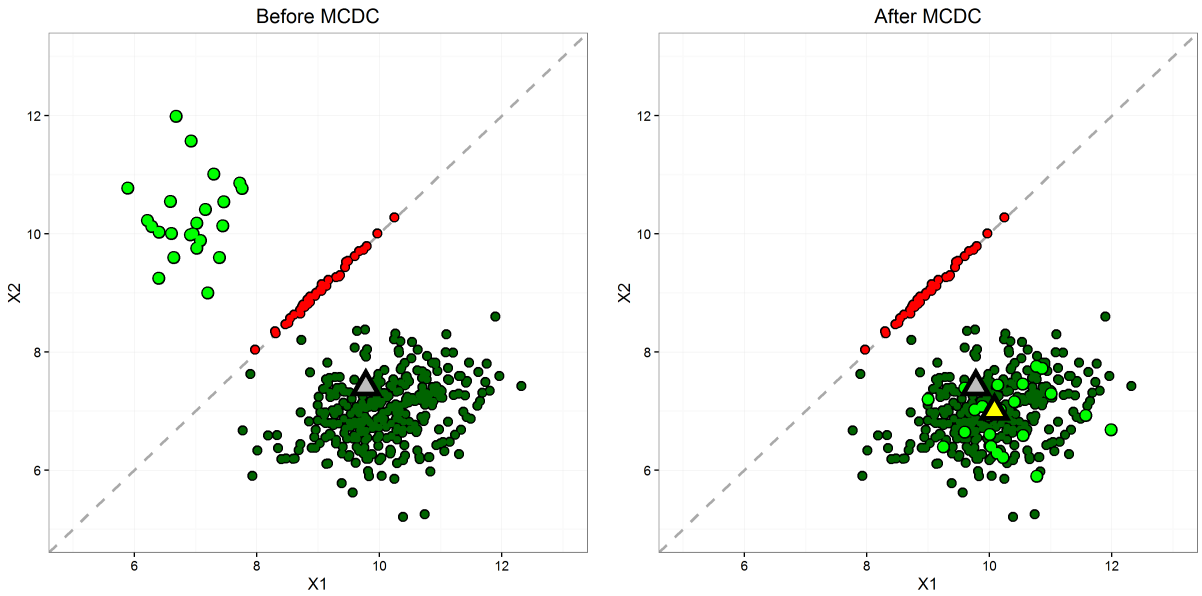

Figure 4.6: Simulated dataset 2: two clusters after flipping. For this example, 90% of the data falls in the main cluster, and $1 - \pi$, the fraction of points in the main cluster being flipped, is chosen to be 0.05. MCDC correctly identifies the two clusters and the flipped points, as seen in the plot on the left. The grey triangle is the mean of all the data, which is moved from its true position due to the second cluster. The yellow triangle is the mean of the largest cluster found by MCDC and is much closer to the true value.

improvement of the data in aggregate as well as improvement in a specific inferential setting. For each pair of genes, I ran MCDC with 1 to 9 clusters on 2,044 untreated experiments and chose the optimal number of clusters by BIC. Running this on a laptop with a 2.6GHz Intel i7-6700HQ processor took approximately 47 minutes for all gene pairs, or under 6 seconds for running MCDC with each of 1 to 9 clusters on a single gene pair.

Three gene pairs were selected to illustrate the results of applying MCDC to the L1000 data. Figure 4.11 shows the results of applying MCDC to one gene pair in untreated experiments on cell line A375. Each point corresponds to a single experiment, and most of the points fall in the same region. However, there is a single point in the top-left corner that appears to be mirrored across the $x = y$ line, and I suspect that this point has had the expression levels of the two genes switched. MCDC corrects this point and it does indeed

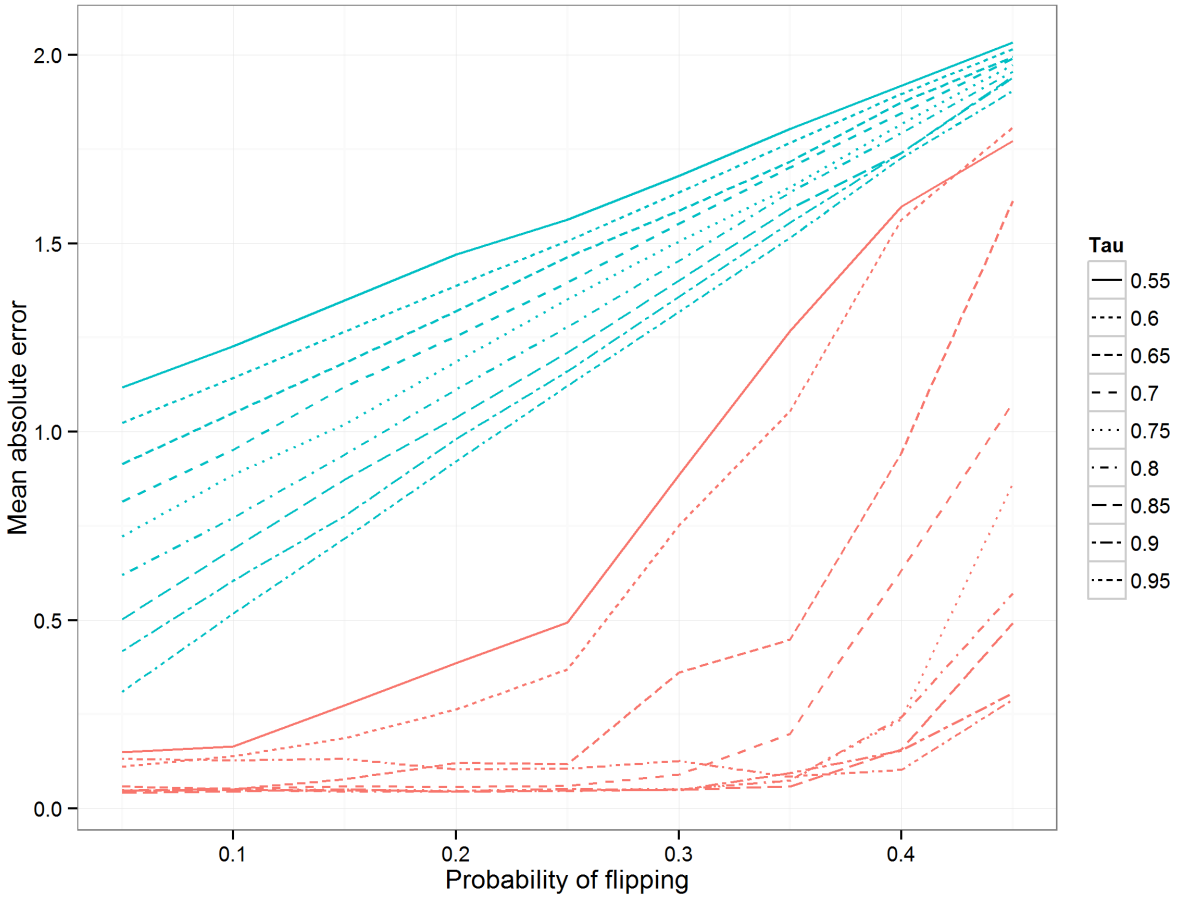

Figure 4.7: Mean absolute error in inferred mean comparison for simulation 2 when varying τ, the probability that a point comes from the primary cluster. The blue line is using unaltered data, while the red line is using the mean of the largest cluster found by MCDC.

fall within the main body of points.

Note that here the best solution by BIC involves three clusters. This means that the distribution of the points may not be strictly normal, but here the components overlap such that they form one contiguous cluster.

Figures 4.12 and 4.13 show MCDC applied to additional gene pairs in the same dataset. In each case MCDC succeeded in removing the artifacts in the data. MCDC selects 3 clusters in Figure 4.12 and 5 clusters in Figure 4.13. Note in Figure 4.13 that the inferred mean after

MCDC is substantially different than the mean of the full dataset, moving it from a location not near any data to within the largest cluster. Figure 4.14 shows the distribution of the number of clusters chosen by BIC across all the gene pairs.

4.5.1 Agreement with External Baseline Data

I wanted to see if MCDC improves the data relative to an external baseline. There are 2,044 untreated experiments in the A375 cell line. These experiments should all yield similar expression levels since they are all done under the same experimental conditions. I get an estimate of the gene expression level of a particular gene by taking the mean across all the experiments. I refer to this as the unaltered data.

There are two expression level baseline datasets included in the LINCS L1000 metadata for cell line A375, one using RNAseq technology and the other using Affymetrix microarray technology. Each of these data were generated using an independent technology and can be compared to the values in the L1000 data. Since the baseline datasets are produced using different technologies, the scales of the expression levels are different than that from the L1000 data. In order to take this difference into account, I looked at the mean squared error (MSE) from a simple linear regression of the baseline data on the inferred gene expression levels from the L1000 data.

I then applied MCDC to see if this improved the estimates of gene expression. To do this, I applied MCDC separately to each pair of genes that were measured using the same bead color. For a gene pair, I ran MCDC on the data from the 2,044 experiments. Doing this for all 500 gene pairs, I ended up with an estimated gene expression level for all of the 1,000 landmark genes. These estimates were also compared to the baseline estimates and I was able to compare the MSEs of the unaltered data with those from the corrected data.

Liu et al. (2015) introduced a new processing pipeline for the LINCS L1000 data in order to address some of the issues they found with the data. They started with the raw data and performed the deconvolution step with a Gaussian mixture model approach rather than the k-means approach used in the original pipeline. This yielded what they refer to as Level

2 data, which they then further normalized and performed quality control on to produce Level 3 data. The Liu Level 2 data can be compared with the Level 3 data from the L1000 pipeline, while the Liu Level 3 data is similar to the Level 4 L1000 data.

As a comparison with MCDC, we looked at the same regression against the Affymetrix and RNAseq baselines using the mean values from the Liu Level 2 data for each gene, again for the A375 untreated experiments. We used the Liu Level 2 data because part of the process of creating the Level 3 data removes the means from the gene data, which is not useful for our purpose. The Liu data included observations from 532 experiments.

Table 4.2 shows the results of this analysis. Using the corrected data improved the MSE by 8% when using the Affymetrix data and by 7% when using the RNAseq data. The Liu data also improved the MSEs, though not as well as MCDC in the case of the Affymetrix baseline.

It is also of interest to note that the performance of MCDC in improving gene expression estimates is not dependent on the number of clusters inferred. This is shown in Figure 4.15, where the improvement in the residual for each gene from the regression using the unaltered means versus the regression using the MCDC-corrected estimates is shown. Figure 4.15 shows the results when using the Affymetrix baseline, and results are similar when using the RNAseq baseline. There is not a substantial change in the improvement based on whether few or many clusters is chosen.

Table 4.2: MSE of regressing external baseline data on imputed gene means. Comparison of unaltered means, means from the Liu data, and MCDC data. Affymetrix and RNAseq baselines both are from external sources independent of the LINCS L1000 data.

	Affymetrix	**RNAseq**
Method	**Baseline**	**Baseline**
Unaltered	1.91	1.66
Liu	1.87	1.58
MCDC	1.76	1.55

4.5.2 Gene Regulatory Network Inference

As well as improving the overall estimates of gene expression levels, MCDC identifies particular experiments where the gene pairs are flipped. This improvement of the data leads to improvements in methods that use the data in a more granular way. To explore this, I applied MCDC to the untreated and knockdown experiments for cell line A375, the same data used in chapter 3. I then applied the posterior probability method to the corrected data for comparison with the method on the uncorrected data. This resulted in two ranked lists of gene pairs with associated posterior probabilities. I compared these with the T&J assessment dataset by taking all edges with a posterior probability over a specified cutoff and creating two-by-two tables showing how well the truncated edgelists overlap with the T&J edgelist.

Table 4.3 shows the two-by-two tables generated at posterior probability cutoffs of 0.5 and 0.95. I also report approximate p-values by using the probability of getting at least the number of true positives found using a binomial(n, p) distribution, where n is the number of pairs in the inferred list and p is the probability of selecting a true edge from the total number of possible edges. From the table, the p-value is better for the corrected data at both probability cutoffs. The corrected data includes more edges at both cutoffs but maintains a similar precision, the proportion of edges which are true edges.

Another way of looking at the results is via the precision-recall curve (Raghavan et al., 1989). Precision and recall are both calculated by truncating the ranked list of edges and looking only at the edges in the truncated list. Precision is the proportion of the edges in the truncated list which are true edges. Recall is the proportion of all true edges which are in the truncated list. The precision-recall curve takes a ranked list of edges from a procedure and shows how the precision varies as more and more edges are included from that list. High precision at low recall indicates that the procedure is good at identifying true edges at the highest probability. This is important in many cases, particularly genetic studies, because it gives researchers good information on where to focus their efforts in subsequent studies.

Table 4.3: Two by two tables for cell line A375 using knockdown experiments for finding edges. Compared to TRANSFAC and JASPAR from Enrichr. When using the unaltered data and looking at edges with posterior probability of 0.5 or greater, 41 of the 302 candidate edges are found in TRANSFAC and JASPAR, and 14 of the 81 candidate edges at a cutoff of 0.95 are true edges. Similarly, when using the MCDC-corrected data, 63 of the 463 candidate edges at a cutoff of 0.5 are true edges and 20 of the 119 at a cutoff of 0.95 are true edges. Approximate binomial p-values included.

		T&J			
		cutoff: 0.5		**cutoff: 0.95**	
		Yes	No	Yes	No
Unaltered	Yes	41	261	14	67
	No	4152	38836	4179	39030
		p-value: 0.02		p-value: 0.02	
MCDC	Yes	63	400	20	99
	No	4130	38697	4173	38998
		p-value: 0.004		p-value: 0.01	

Figure 4.16 compares the precision-recall curves for the unaltered and corrected data at the very top of the edgelist. The dashed line shows what would be expected by randomly ordering the edges, and above that line is an improvement. Both methods give improved results, but the corrected data yields much better results for the very top edges returned. This is of particular importance for further research because having high confidence in the top edges allows the biologist to develop further experiments to focus on these edges in additional, more targeted experiments. In this respect, data correction with MCDC provides substantial improvement.

This is seen by looking at the inferred edges, ranked so that the first edge has the highest posterior probability, the second has the second highest, and so one. Table 4.4 is constructed by ranking the edgelist from the posterior probability method on a particular dataset. Thus

the first edge in the list is that with the highest posterior probability, the second edge has the next highest posterior probability, and so on. Each edge is looked for in the T&J assessment edgelist. The rank in the table indicates the position at with the n-th edge in T&J was found in the ranked edgelist. So the edge with highest posterior probability using the MCDC-corrected data is in T&J, as is the edge with the 5th highest posterior probability, etc. Only one of the top 10 edges from the unaltered data is a true edge, while 5 of the top 10 edges from the MCDC-corrected data are true edges.

Table 4.4: Comparison of the rank of the first 5 edges found that match the TRANSFAC and JASPAR edgelist. Edges ranked by posterior probability. MCDC-corrected data produces found edges at higher ranks than the uncorrected data. See text for explanation of how the table was constructed.

Found Edge	**Unaltered Rank**	**MCDC Rank**
1	7	1
2	11	5
3	14	6
4	19	7
5	26	10

I also applied the posterior probability method to the Liu Level 3 data. In this case, the Level 3 data is appropriate since it is more comparable to the z-value transformation and quality control has been performed to improve the data. I used the same prior and choice of g as for the other datasets. Due to the quality control step employed by Liu, the number of experiments available was decreased and only 24,377 testable edges had posterior probabilities. Of these, 2,746 were in the T&J assessment dataset. None of the edges had a posterior probability over 0.95, and only two had a posterior probability of at least 0.5. Neither of these edges were in the T&J dataset. When I looked at the edges as ranked by posterior probability, I found that the edge with the 12th-highest posterior probability, at 0.04, was the first which was also found in T&J. The lack of high posterior probability edges

and lack of positive results is due in part to the quality control step used, which resulted in fewer observations for each knockdown data set.

4.6 Discussion

When working with any data, understanding the unique aspects of how it was generated and processed can be helpful in developing models and methods, leading to improved inference. This is particularly true with genomic data. There are often many steps of data transformation and normalization between the raw measured data and what is used by the researcher in drawing conclusions (Binder and Preibisch, 2008). When these steps are not known or understood, assumptions about sources of error can be misinformed and lead to degraded performance in inference. Price et al. (2006) identified population stratification of allele frequency in disease studies, while Gomez-Alvarez et al. (2009) found that a particular sequencing technique resulted in many artificial replicates. Lehmann et al. (2013) showed that quantile normalization of microarray data introduced a phase shift into time-series in strains of cyanobacteria, changing night-expressed genes into day-expressed genes and vice versa. Stokes et al. (2007) developed a tool to identify and remove artifacts in genomic data. Batch effects have been identified as a significant source of systematic error that can be corrected (Leek et al., 2010; Chen et al., 2011; Sun et al., 2011). Identifying these sources of error is crucial, and in some cases can lead to vastly improved results.

I showed how understanding the data-processing pipeline of the LINCS L1000 data allowed identification of the introduction of a particular error, namely the flipping of expression values for gene pairs. This led to the development of MCDC, which is able to identify and correct these flipping errors. I was able to apply MCDC to improve the L1000 data in aggregate, as measured against external standards. This improvement of the data also led to improved inference of regulatory relationships between the genes, in particular for the edges ranked highest. Moreover, the use of the EM algorithm for optimization makes MCDC fast and useful for large datasets.

I showed that MCDC compared favorably to the data from Liu et al. (2015) in my

assessments, but it is important to note that my approach is orthogonal to theirs. As an example of this, Figure 4.17 shows the Liu Level 2 data for a gene pair in the untreated A375 data. There is evidence that this data would benefit from MCDC as well, prior to the processing that creates the Level 3 data.

MCDC is an extension of model-based clustering, which has been extensively used in other analyses of genetic data, including sequence analysis (Verbist et al., 2015) and image analysis of microarrays (Li et al., 2005). One of the most common uses of model-based clustering in genetics is in finding meaningful groups among gene expression profiles across multiple experiments under different experimental conditions (cell sources, phases, applied drugs, etc.) (Siegmund et al., 2004; Jiang et al., 2004). This includes methods using Gaussian mixture models (Yeung et al., 2001), infinite mixture models (Medvedovic and Sivaganesan, 2002) and Bayesian hierarchical clustering (Cooke et al., 2011). The use of MCDC as a step in improving data quality is complementary to these analysis methods. An implementation of this method will be made available as an `R` package, **mcdc**, on CRAN.

I showed in the simulation experiments that MCDC is able to accurately identify the data points which have been altered and thus improve the quality of the data. It is not limited to flipping as seen in the LINCS data, but is able to handle any dataset where a subset of the data points have been altered in a known way. For the L1000 data, the transformation is informed by understanding the way the data is generated and pre-processed. In cases where there are multiple possibilities for $\mathbf{T}$, it may be possible to run MCDC with each candidate transformation and compare the results to identify the one most compatible with the data.

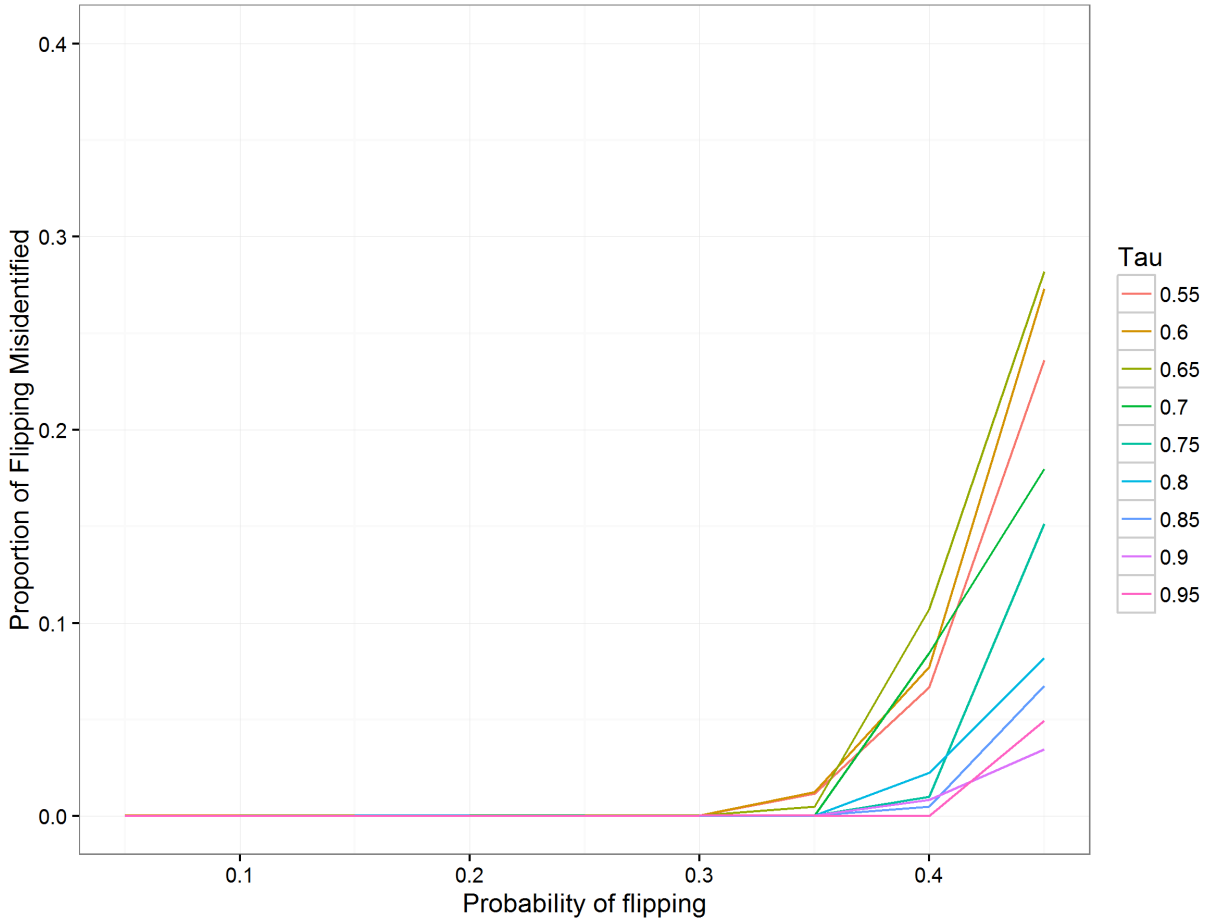

Figure 4.8: Proportion of points correctly identified as flipped or not flipped for simulation 2 when varying τ, the probability that a point comes from the primary cluster. When there is a high probability of flipping (near 0.5), there may be more points in the flipped cluster, leading to MCDC identifying it as the main cluster and thus misidentifying all points for a particular dataset.

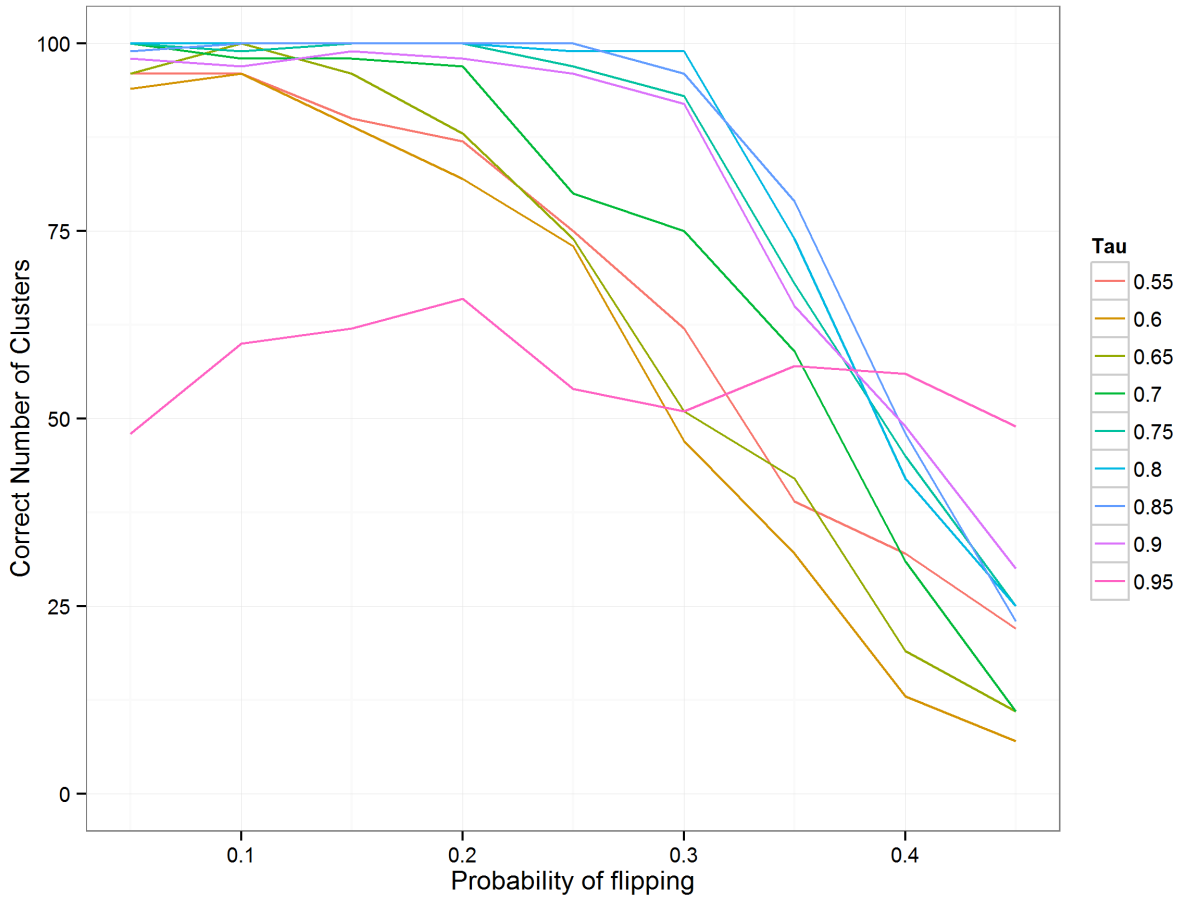

Figure 4.9: Number of datasets (out of 100) which identify 2 clusters as the best result for simulation 2 when varying τ, the probability that a point comes from the primary cluster. When τ was high, MCDC did not always correctly identify the cluster on the diagonal properly due to the low number of points in that cluster.

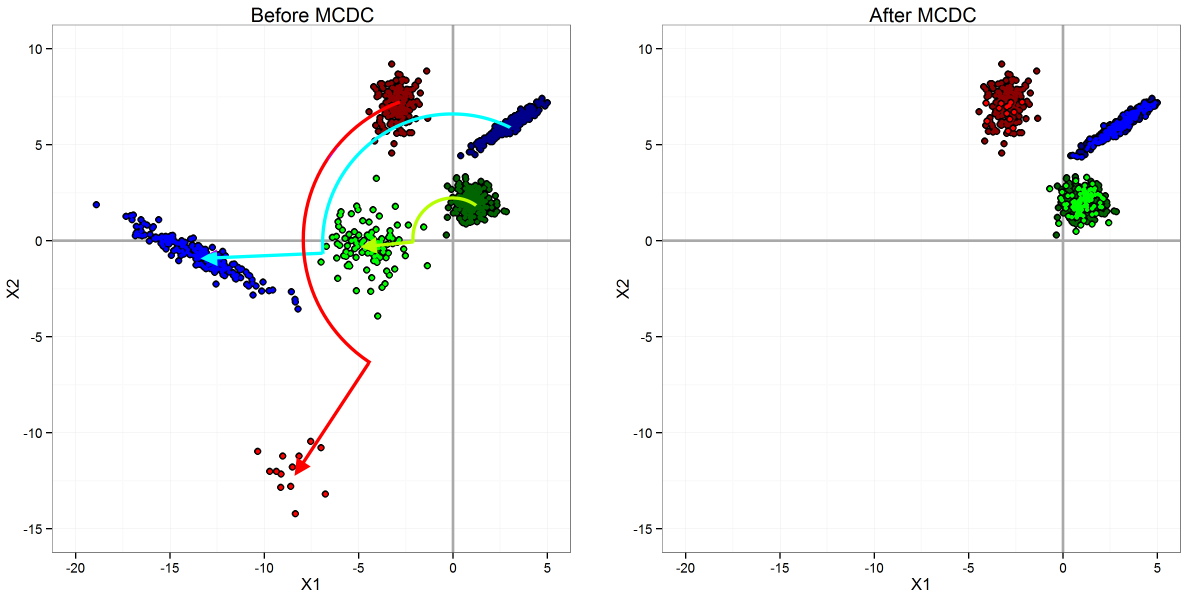

Figure 4.10: Simulation dataset 3: 3 clusters with rotation and scaling. A data point is transformed by rotating it 120° counter-clockwise around the origin and then scaling out from the origin by a factor of 2, as seen in the plot on the left. MCDC is able to identify the correct clusters and assign the transformed points back into the appropriate clusters, as on the right.

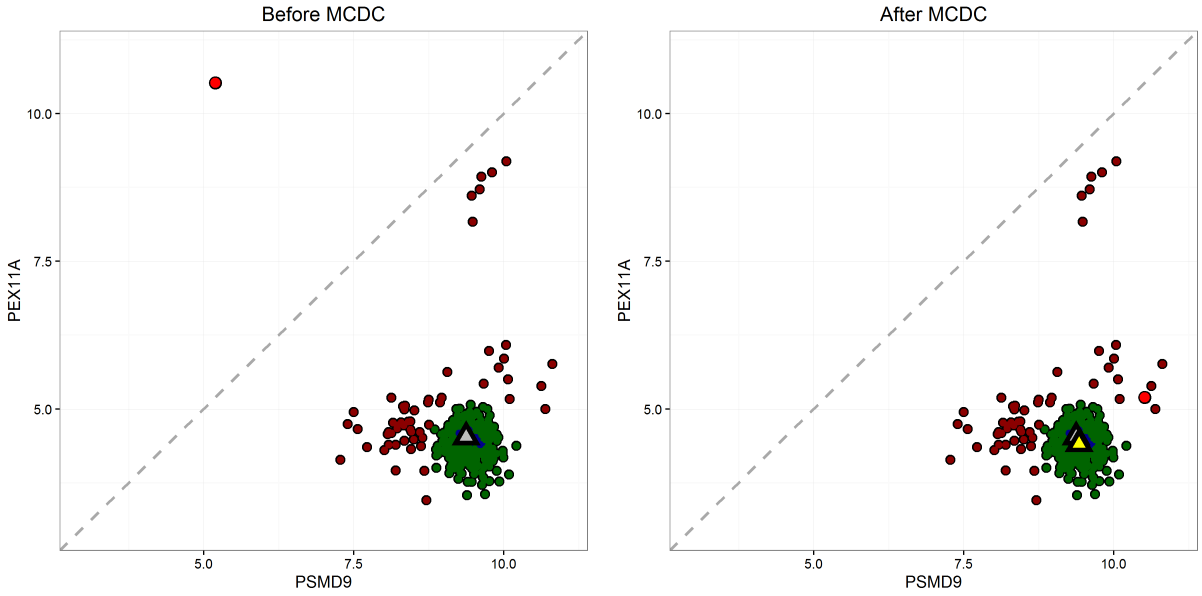

Figure 4.11: Example 1 showing results of applying MCDC to L1000 control data. MCDC chooses 3 clusters by BIC. On the left is the data before correction, and on the right is the data after correction. Triangles indicate inferred mean - gray is the mean of all the data while yellow is the mean of the largest cluster found by MCDC.

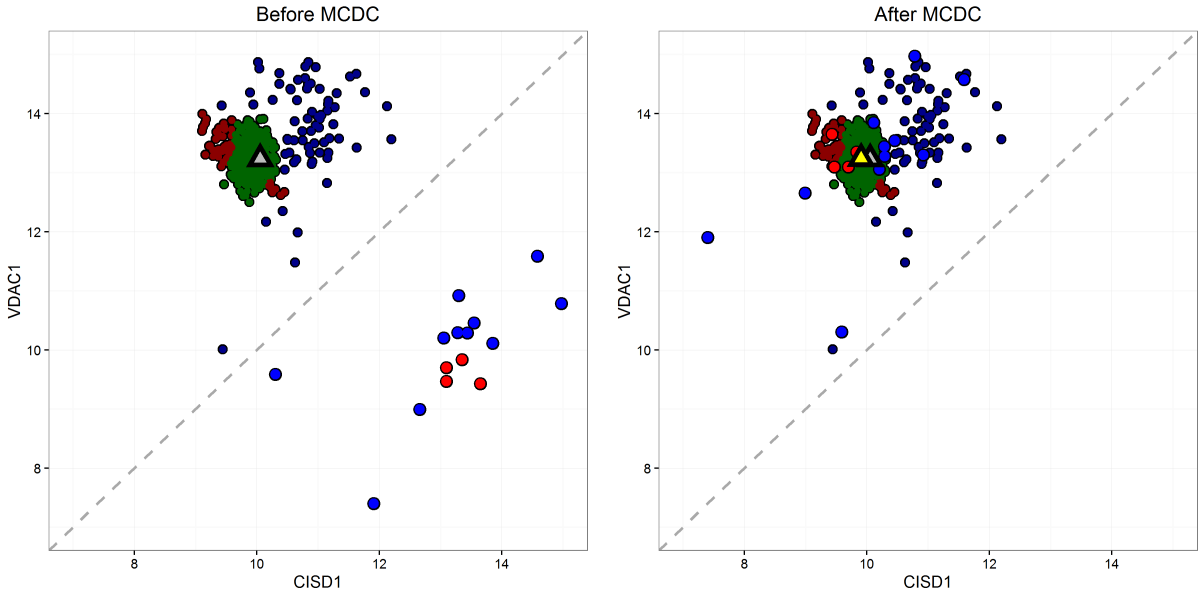

Figure 4.12: Example 2 showing results of applying MCDC to L1000 control data. MCDC chooses 3 clusters by BIC. On the left is the data before correction, and on the right is the data after correction. Triangles indicate inferred mean - gray is the mean of all the data while yellow is the mean of the largest cluster found by MCDC.

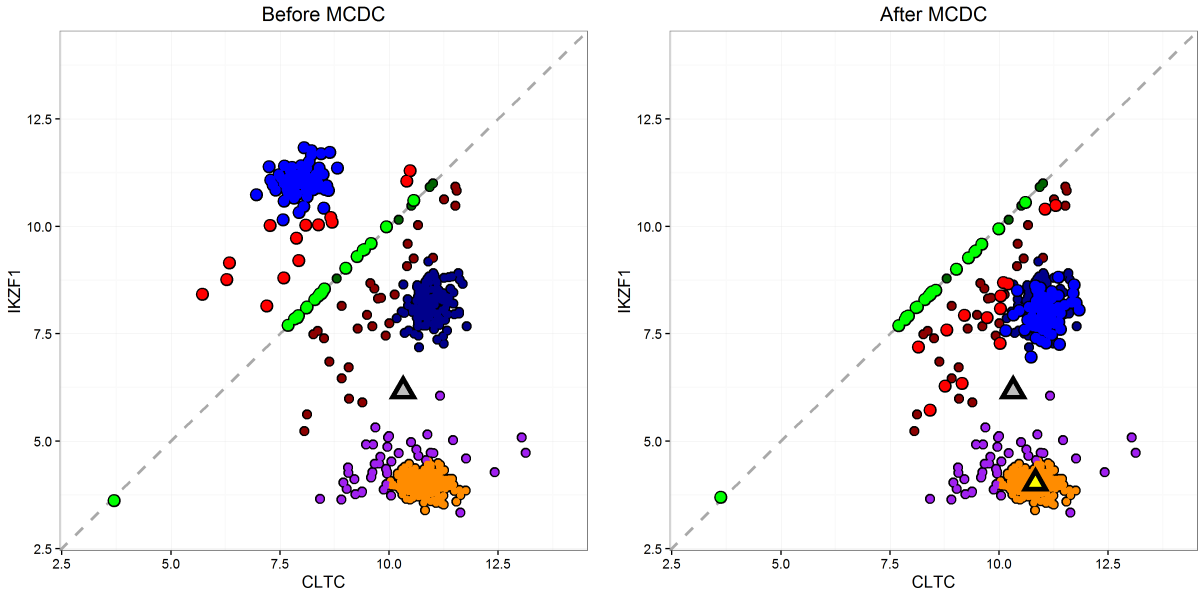

Figure 4.13: Example 3 showing results of applying MCDC to L1000 control data. MCDC chooses 5 clusters by BIC. On the left is the data before correction, and on the right is that data after correction. Triangles indicate inferred mean - gray is the mean of all the data while yellow is the mean of the largest cluster found by MCDC.

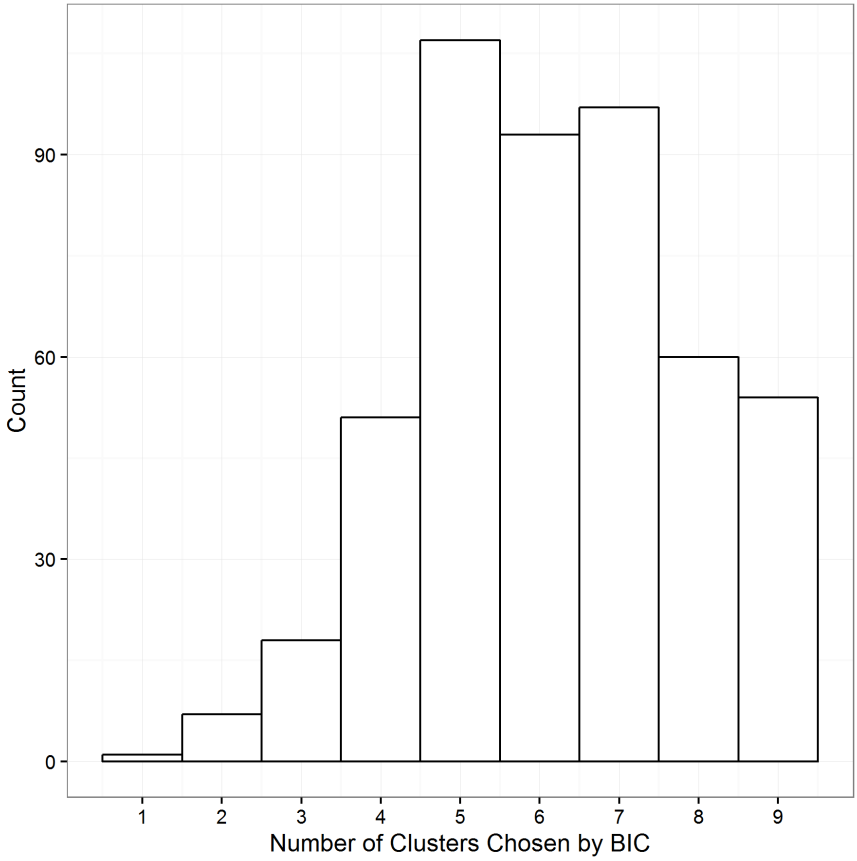

Figure 4.14: Histogram of the number of clusters chosen by BIC for each gene pair.

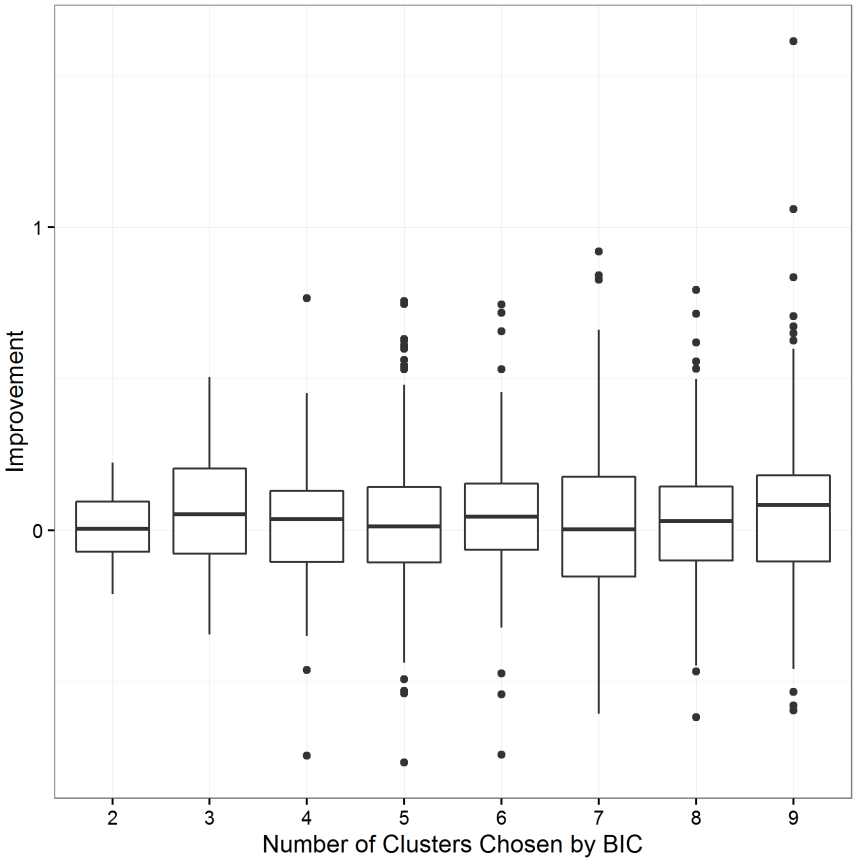

Figure 4.15: Boxplots showing the improvement in gene expression estimation for each gene versus the Affymetrix baseline, by the number of clusters chosen. Improvement is calculated as the absolute residual from regression using the original data minus the absolute residual from regression using the MCDC estimates. Positive values indicate improvement from using MCDC.

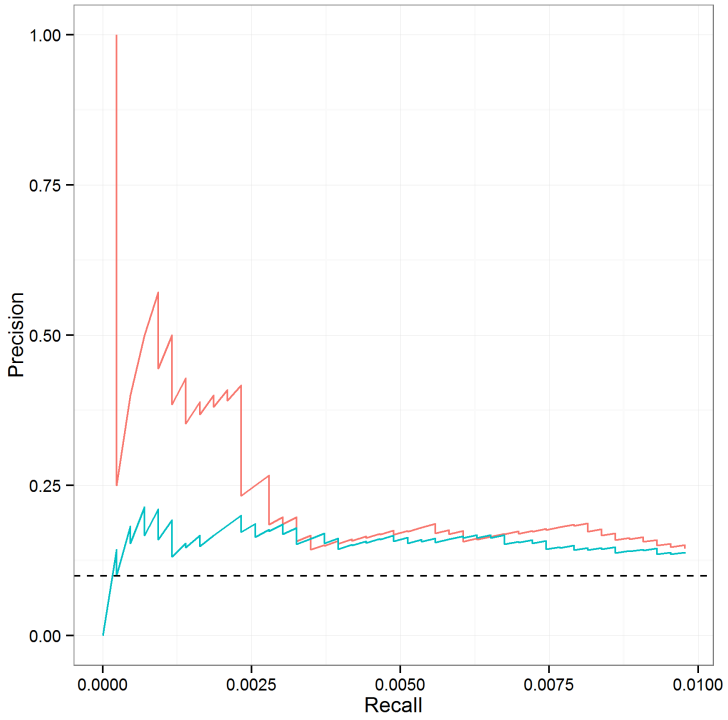

Figure 4.16: Precision-recall curve comparing edgelists from unaltered (blue line) and MCDC-corrected (red line) data on knock down data.

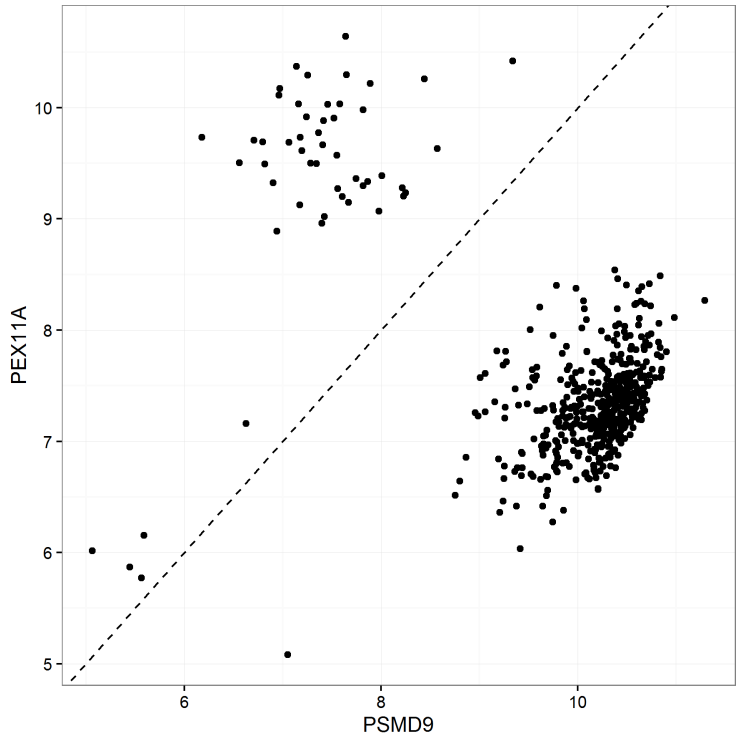

Figure 4.17: Expression levels for two paired genes in the untreated experiments for cell line A375 from the Liu Level 2 data (532 experiments), demonstrating that MCDC could potentially be useful in this processing pipeline as well as the original L1000 pipeline.

Chapter 5

IDENTIFYING VAR PARAMETERS FROM EQUILIBRIUM SAMPLES

5.1 Motivation - Steady State Data

I have discussed methods for inferring regulatory relationships from time-series data and knockdown data. However, these are not the only types of gene expression data available for analysis. In the LINCS L1000 data, for example, most of the experiments involve chemical perturbations. In these experiments, a drug is added to the cell culture and, after a specified period of time, the expression levels are measured. Additionally, the L1000 protocol results in a large number of control experiments. These are experiments on every plate that serve as a baseline reference for the other experiments on the plate. That these experiments are done on every plate means that they make up the largest set of experiments done with a single experimental condition.

Many of the methods previously discussed are applied to perturbation data as well. Mutual information methods and correlation-based methods are commonly used, although the resulting networks tend to lack directionality (Tusher et al., 2001; Basso et al., 2005; Margolin et al., 2006; Faith et al., 2007; Meyer et al., 2007). Regression-based methods are also applied to steady-state data (Omranian et al., 2016; Singh and Vidyasagar, 2016) to infer network structure. Bayesian network methods (Friedman et al., 2010) can result in directed graphs, but the networks generated are, by necessity, acyclic. This means that any cycles, as are found in real biological systems, will not be captured.

In this chapter I propose a method based on an implicit time-series model that uses steady-state data and has the capability to test the existence of edges in directed networks. Additionally, this method is not restricted to acyclic networks, but can be used to infer

information about cycles in the network. I present proofs of consistency and efficiency in a constrained case as well as a likelihood ratio test for the existence of an edge. I also look at simulation results and apply the method to a synthetic dataset.

5.2 VAR(1) Model

First, consider again the model that was used for time-series data for a system with p genes. This is a VAR(1) model with no correlation in the error term between genes (Lütkepohl, 2005). We can write the model as

$$\begin{aligned} \mathbf{x}_t &= \mathbf{A}\mathbf{x}_{t-1} + \epsilon_t, \\ \epsilon_t &\sim N(0, \mathbf{D}), \end{aligned}$$

where $\mathbf{D}$ is diagonal. $\mathbf{x}_t$ and ϵ_t are vectors of length p while $\mathbf{A}$ and $\mathbf{D}$ are $p \times p$ matrices. Here, $\mathbf{A}$ identifies the relationships between the genes. A non-zero (i, j)-th element of $\mathbf{A}$ indicates that there is an edge from gene j to gene i. With independent errors, this is a restatement of the time-series model used for the yeast data. At first glance, it does not appear to be applicable to steady-state data. However, if the eigenvalues of $\mathbf{A}$ are less than 1 in absolute value, then as $t \to \infty$, x_t converges to a stable equilibrium distribution (Anderson, 2000). We can use this equilibrium distribution as the model for steady-state data. This can be applied to either a set of experiments with the same perturbation applied or to the control experiments.

5.2.1 Equilibrium Distribution

Since this is a Gaussian VAR model, we can write the equilibrium distribution (if it exists) as

$$\mathbf{x}_\infty \sim N(\mathbf{0}, \mathbf{\Sigma}),$$

where $\mathbf{\Sigma}$ is also a $p \times p$ matrix. To find $\mathbf{\Sigma}$, use an iterative relationship between the variances at consecutive time points:

$$\begin{aligned} \text{Var}[\mathbf{x}_1] &= \mathbf{D}, \\ \text{Var}[\mathbf{x}_{t+1}] &= \mathbf{A}\text{Var}[\mathbf{x}_t]\mathbf{A}^T + \mathbf{D}, \\ \mathbf{\Sigma} &= \sum_{i=0}^{\infty} \mathbf{A}^i \mathbf{D} (\mathbf{A}^T)^i. \end{aligned}$$

I will call this the *summation identity* (Sheppard, 2013). Another expression for the asymptotic variance uses the identity $\text{vec}(\mathbf{AXB}) = (\mathbf{B}^T \otimes \mathbf{A})\text{vec}(\mathbf{X})$:

$$\begin{aligned} \mathbf{\Sigma} &= \sum_{i=0}^{\infty} \mathbf{A}^i \mathbf{D} (\mathbf{A}^T)^i, \\ \text{vec}(\mathbf{\Sigma}) &= \text{vec}\left(\sum_{i=0}^{\infty} \mathbf{A}^i \mathbf{D} (\mathbf{A}^T)^i \right), \\ &= \sum_{i=0}^{\infty} \text{vec}(\mathbf{A}^i \mathbf{D} (\mathbf{A}^T)^i), \\ &= \sum_{i=0}^{\infty} \left(\mathbf{A}^i \otimes \mathbf{A}^i \right) \text{vec}(\mathbf{D}), \\ &= \sum_{i=0}^{\infty} (\mathbf{A} \otimes \mathbf{A})^i \, \text{vec}(\mathbf{D}), \\ &= (\mathbf{I} - \mathbf{A} \otimes \mathbf{A})^{-1} \, \text{vec}(\mathbf{D}). \end{aligned}$$

This *Kronecker identity* is convenient for calculating the asymptotic variance from a given $\mathbf{A}$ and $\mathbf{D}$ since it eliminates the infinite sum (Sheppard, 2013). We can also write out a *recursive identity* for $\mathbf{\Sigma}$ since, at the equilibrium, the variance does not change from one time step to the next:

$$\mathbf{\Sigma} = \mathbf{A}\mathbf{\Sigma}\mathbf{A}^T + \mathbf{D}. \tag{5.1}$$

5.3 Identifiability

When looking at steady-state data as opposed to time-series data, we must rely on the relationship between $(\mathbf{A}, \mathbf{D})$ and $\mathbf{\Sigma}$ in order to infer anything about $\mathbf{A}$ and $\mathbf{D}$. This raises

an issue of identifiability. As a simple example, substituting $-\mathbf{A}$ for $\mathbf{A}$ in any of the variance identities above does not change anything. More generally, let $\mathbf{Q}$ be any $p \times p$ orthogonal matrix. An orthogonal matrix is defined as any matrix such that $\mathbf{Q}\mathbf{Q}^T = \mathbf{I}$. For any such $\mathbf{Q}$, $\tilde{\mathbf{A}}$ defined as

$$\tilde{\mathbf{A}} = \mathbf{A}\boldsymbol{\Sigma}^{1/2}\mathbf{Q}\boldsymbol{\Sigma}^{-1/2}$$

will satisfy the recursive identity (Tong et al., 1992).

We can state the identifiability problem as follows. Under what conditions is the map

$$(\mathbf{A}, \mathbf{D}) \leftrightarrow \boldsymbol{\Sigma}$$

bijective? As a necessary condition we can count parameters.

Proposition 5.3.1. *A necessary condition for identifiability in the VAR equilibrium model of dimension p is that the number of non-zero off-diagonal elements of $\mathbf{A}$ must be no more than $p(p-3)/2$. Additionally, it is required that $p \geq 5$.*

Proof. The number of parameters in $\boldsymbol{\Sigma}$ for a p-dimensional model is $p(p+1)/2$. The total number of parameters in $\mathbf{A}$ and $\mathbf{D}$ must not exceed that number. $\mathbf{D}$ is diagonal and thus accounts for p parameters. Similarly, the diagonal of $\mathbf{A}$ is assumed to be non-zero - the expression levels of each gene at timepoint $t+1$ is dependent on the expression level of that gene at timepoint t for all genes. This accounts for an additional p parameters. Thus a maximum $p(p-3)/2$ off-diagonal elements of $\mathbf{A}$ may be non-zero. Now, the graph defined by $\mathbf{A}$ must be connected. If it is not, then it can be decomposed into two smaller, independent components, and each of those must be identifiable. We need at least $p-1$ off-diagonal elements of $\mathbf{A}$ for it to be connected, so p must be at least 5. If $p = 4$, only a maximum of 2 off-diagonal elements is allowed, which is not enough to fully connect the 4 nodes. □

Parameter counting leads to a necessary condition, but it is not a sufficient condition. I have not found sufficient conditions in this general case. I have, however, looked at identifiability in two specific examples, one acyclic and one cyclic. To assess the identifiability of a

specific model, I used the recursive identity,

$$\mathbf{\Sigma} = \mathbf{A}\mathbf{\Sigma}\mathbf{A}^T + \mathbf{D}.$$

If a given model is identifiable, then there will be only one $(\mathbf{A}, \mathbf{D})$ pair which satisfies this equation.

The first example is a 5-dimensional model with a simple network structure, given in figure 5.1. This model has 4 off-diagonal non-zero elements in $\mathbf{A}$. The maximum allowable by parameter counting is 5, so we are close but not at that limit. Writing out the recursive identity as a system of equations yields equations that look like

$$\begin{aligned}
\sigma_{11} &= a_{11}^2\sigma_{11} + 2a_{11}a_{12}\sigma_{12} + a_{22}^2\sigma_{22} + d_1, \\
\sigma_{12} &= a_{11}a_{22}\sigma_{12} + a_{12}a_{22}\sigma_{22} + a_{11}a_{23}\sigma_{13} + a_{12}a_{23}\sigma_{23}, \\
&\dots
\end{aligned}$$

where a_{ij} is the entry of $\mathbf{A}$ in the i'th row and j'th column, and d_i is the i-th diagonal entry in $\mathbf{D}$. Next, I solve the system of equations to eliminate as many variables as possible. Then I perform a grid search on the free variables to find all potential solutions of the recursive identity. In this example, I was able to solve the system down to a single free parameter, a_{55}.

To test the parameter solutions, I used the following model parameters:

$$\begin{aligned}
\mathbf{A} &= \begin{bmatrix} 0.6 & 0.3 & 0 & 0 & 0 \\ 0 & 0.4 & -0.4 & 0 & 0 \\ 0 & 0 & 0.8 & 0.5 & 0 \\ 0 & 0 & 0 & 0.5 & -0.3 \\ 0 & 0 & 0 & 0 & 0.7 \end{bmatrix}, \\
\mathbf{D} &= \mathbf{I}_5.
\end{aligned}$$

The grid search was performed by allowing a_{55} to vary between 0 and 1 in increments of 0.0001. This is the allowable range for a_{55} since $\mathbf{A}$ and $-\mathbf{A}$ both result in the same $\mathbf{\Sigma}$. For

each prospective value of a_{55}, the other elements of $\mathbf{A}$ and $\mathbf{D}$ were calculated. These values were then used to evaluate the objective function $\log ||\mathbf{\Sigma} - \mathbf{A\Sigma A}^T - \mathbf{D}||_2^2$. Figure 5.2 shows the results. From this it is clear that the only value of a_{55} that satisfies the recursive identity is the true value, 0.7, where the objective is minimized. The flat region of the objective curve to the left of $a_{55} \approx 0.65$ reflects the fact that there is no valid solution for $\mathbf{A}$ and $\mathbf{D}$ with that starting value for a_{55}, either because some element of $\mathbf{A}$ is not a real value or because a diagonal element of $\mathbf{D}$ is negative. Also note that there is a degenerate solution at $a_{55} = 1$ such that $\mathbf{A} = \mathbf{I}$ and $\mathbf{D} = \mathbf{0}$. Since $\mathbf{D}$ represents the variance of the noise added to the VAR system, its diagonal elements must be positive. So in this example the model is identifiable.

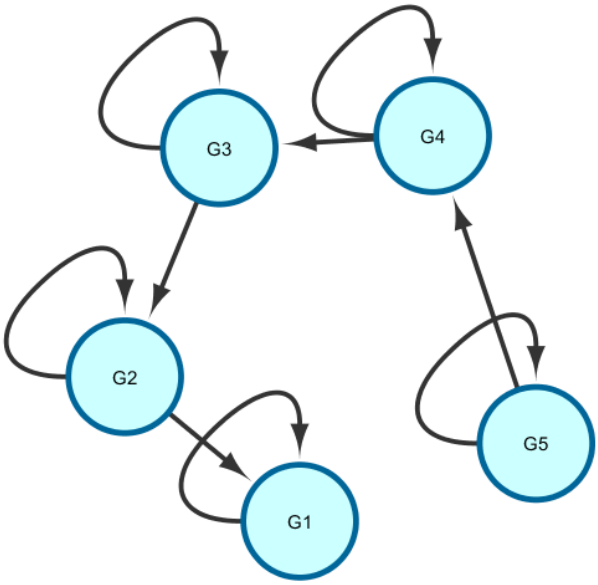

Figure 5.1: The network structure for the cyclic identifiability example.

I also did an example using a cyclic network. For this example, I added a single edge to the previous network making a 3-node cycle, as shown in Figure 5.3. I used the following

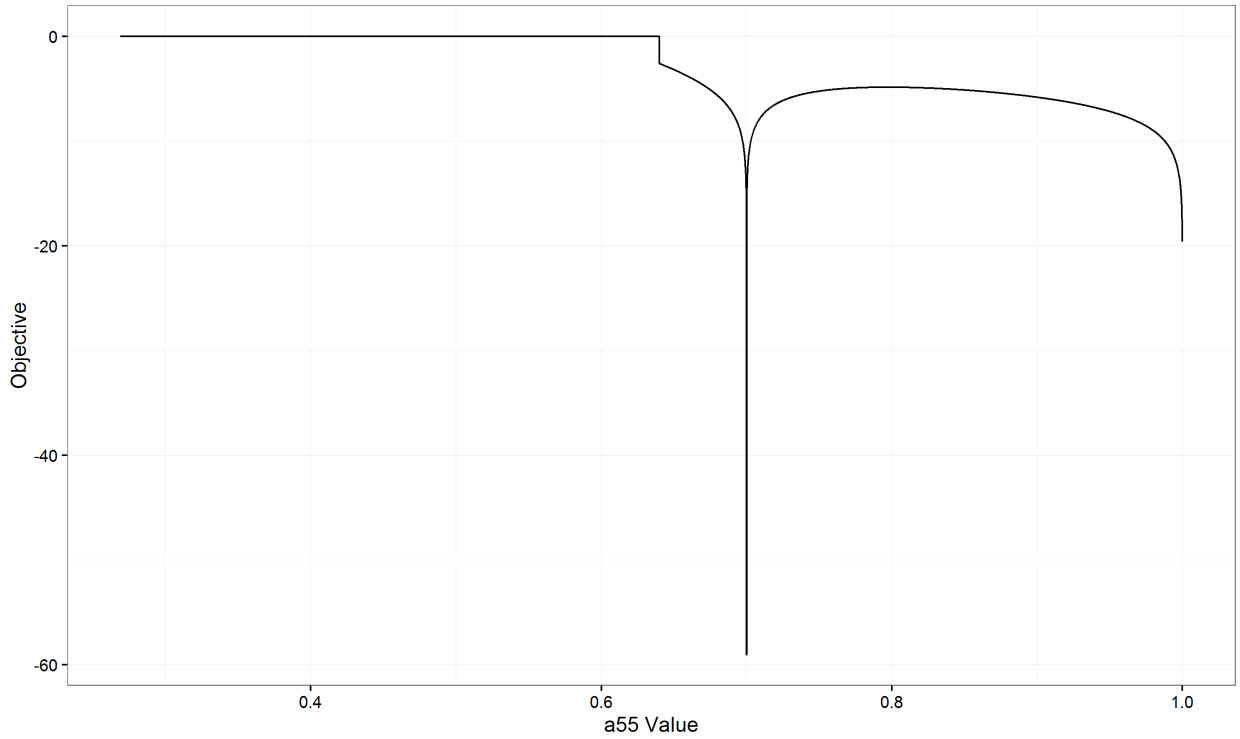

Figure 5.2: Grid search for solutions of the acyclic identifiability example. The objective value is minimized at the true value of $a_{55} = 0.7$, showing that the model is identifiable in this example.

model parameters:

$$\mathbf{A} = \begin{bmatrix} 0.3 & 0.6 & 0 & 0 & 0 \\ 0 & 0.1 & -0.3 & 0 & 0 \\ 0 & 0 & 0.7 & 0.2 & 0 \\ 0 & 0.3 & 0 & 0.4 & -0.6 \\ 0 & 0 & 0 & 0 & 0.5 \end{bmatrix},$$
$$\mathbf{D} = \mathbf{I}_5.$$

For this model, I was able to solve down to two free parameters, a_{45} and a_{55}. I again performed a grid search over the appropriate parameter space ($a_{45} \in (-1, 1)$, $a_{55} \in (0, 1)$) with an initial grid increment of 0.001 and evaluated the objective at each point. Figure 5.4

shows the results of the grid search, both in the larger space as well as zoomed in around the truth, where a much smaller grid increment of 10^{-11} was used. There are large areas where a valid solution is not achieved, indicated by the gray areas. Again, however, the original solution is the only valid one, indicating that this model is also identifiable.

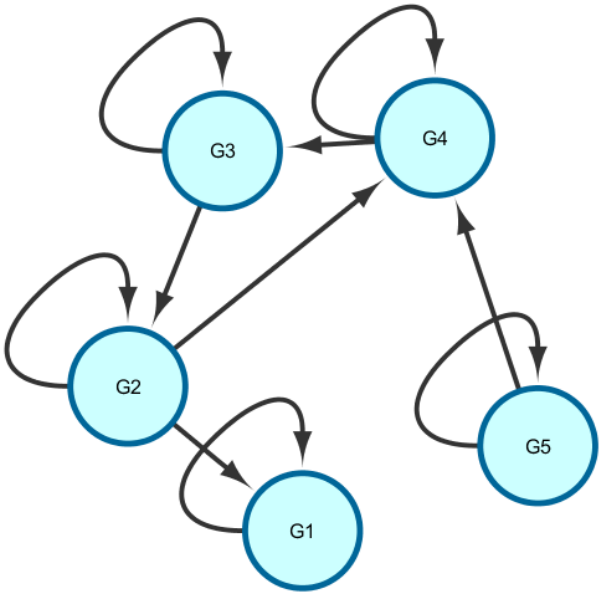

Figure 5.3: The network structure for the cyclic identifiability example.

5.3.1 Simple Case

Although the parameter counting condition is helpful and results in identifiability in at least a couple examples, it is important to find a more general result for identifiability. This can be done by restricting the model considerably to make it tractable to analysis. First, I constrain **D** to be known. Secondly, I require that the model is acyclic and thus the nodes can be arranged such that **A** is lower-diagonal. Finally, the sign of the diagonal entries of **A** must be known and they must be less than 1. With these constraints, the model is identifiable and has good asymptotic properties.

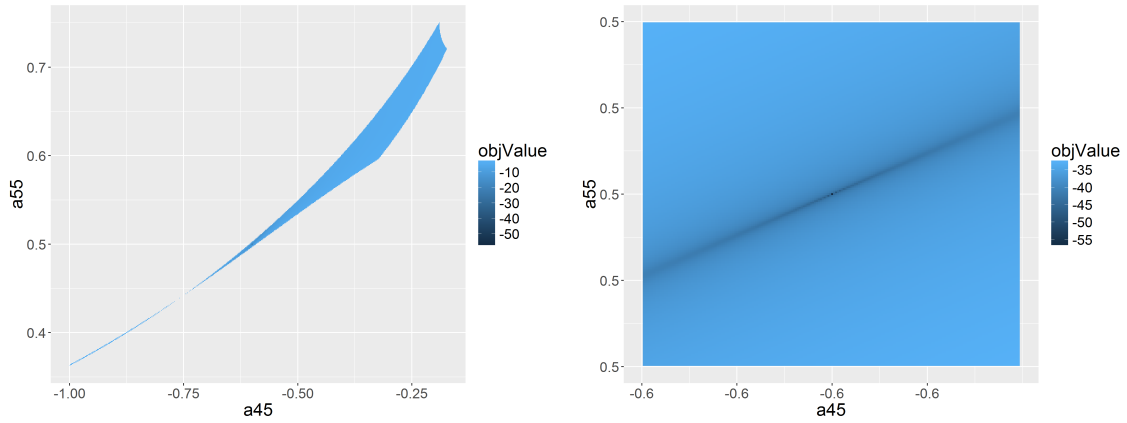

Figure 5.4: Grid search for solutions of the cyclic identifiability example showing the objective value at each point searched. The left plot shows the grid search over the entire model space, while the right plot is zoomed in on the true parameter value. The gray areas indicate parameter values that result in invalid solutions. The true solution at (-0.6, 0.5) is the only valid solution.

Consider the model as constrained above. Order the nodes in the network topologically, meaning that there are no edges from node i to node j if $i > j$. This creates an $\mathbf{A}$ that is lower-diagonal. We will use the recursive identity for the equilibrium variance:

$$\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T + \mathbf{D} = \boldsymbol{\Sigma}, \tag{5.2}$$

$$\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T = \boldsymbol{\Sigma} - \mathbf{D}. \tag{5.3}$$

We can solve this identity recursively in a manner inspired by Drton et al. (2011) by looking at the leading principal submatrices. First, define $\mathbf{S}_i$ to be the leading $i \times i$ principal submatrix of $\boldsymbol{\Sigma}$. That is the matrix made of elements in the first i rows and columns of $\boldsymbol{\Sigma}$. Similarly, define $\mathbf{B}_i$ to the be the leading principal submatrix of $\boldsymbol{\Sigma} - \mathbf{D}$ and $\boldsymbol{\Gamma}_i$ to be the leading principal submatrix of $\mathbf{A}$. Also, define $\mathbf{s}_{i+1}$ to be the vector containing the first i elements of the $(i+1)^{\text{th}}$ column of $\boldsymbol{\Sigma}$ or $\mathbf{B}$. That is, $\mathbf{s}_{i+1} = [\sigma_{1,i+1}, \ldots, \sigma_{i,i+1}]'$. Similarly,

$\boldsymbol{\gamma}_{i+1}^T$ is the vector containing the first i elements of the $(i+1)^{\text{th}}$ row of $\boldsymbol{\Gamma}$. So,

$$\begin{aligned}
\mathbf{S}_{i+1} &= \begin{bmatrix} \mathbf{S}_i & \mathbf{s}_{i+1} \\ \mathbf{s}_{i+1}^T & s_{i+1,i+1} \end{bmatrix}, \quad s_{ii} = \sigma_{ii} \\
\mathbf{B}_{i+1} &= \begin{bmatrix} \mathbf{B}_i & \mathbf{s}_{i+1} \\ \mathbf{s}_{i+1}^T & b_{i+1,i+1} \end{bmatrix}, \quad b_{ii} = \sigma_{ii} - d_{ii} \\
\boldsymbol{\Gamma}_{i+1} &= \begin{bmatrix} \boldsymbol{\Gamma}_i & 0 \\ \boldsymbol{\gamma}_{i+1}^T & \gamma_{i+1,i+1} \end{bmatrix}, \quad \gamma_{ii} = a_{ii}.
\end{aligned}$$

Now, for $i = 1$, we get

$$\begin{aligned}
\gamma_{ii}^2 s_{ii} &= b_{ii}, \\
a_{ii} &= \sqrt{\frac{\sigma_{ii} - d_{ii}}{\sigma_{ii}}}.
\end{aligned}$$

This gives us $\mathbf{A}_1$. Now, suppose we know $\mathbf{A}_i$. Then, for $i+1$, we have

$$\begin{aligned}
\mathbf{B}_{i+1} &= \boldsymbol{\Gamma}_{i+1}\mathbf{S}_{i+1}\boldsymbol{\Gamma}_{i+1}^T, \\
\begin{bmatrix} \mathbf{B}_i & \mathbf{s}_{i+1} \\ \mathbf{s}_{i+1}^T & b_{i+1,i+1} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{\Gamma}_i\mathbf{S}_i\boldsymbol{\Gamma}_i^T & \boldsymbol{\Gamma}_i\mathbf{S}_i\boldsymbol{\gamma}_{i+1} + \gamma_{i+1,i+1}\boldsymbol{\Gamma}_i\mathbf{s}_{i+1} \\ - & \boldsymbol{\gamma}_{i+1}^T\mathbf{S}_i\boldsymbol{\gamma}_{i+1} + 2\gamma_{i+1,i+1}\mathbf{s}_{i+1}^T\boldsymbol{\gamma}_{i+1} + \gamma_{i+1,i+1}^2 s_{i+1,i+1} \end{bmatrix}.
\end{aligned}$$

We solve for $\gamma_{i+1,i+1}$ and $\boldsymbol{\gamma}_{i+1}$ to get

$$\begin{aligned}
\gamma_{i+1,i+1} &= \sqrt{\frac{b_{i+1,i+1} - \mathbf{s}_i^T(\boldsymbol{\Gamma}_i\mathbf{S}_i\boldsymbol{\Gamma}_i^T)^{-1}\mathbf{s}_i}{s_{i+1,i+1} - \mathbf{s}_i^T\boldsymbol{\Gamma}^{-1}\mathbf{s}_i}}, \\
\boldsymbol{\gamma}_{i+1} &= (\boldsymbol{\Gamma}_i\mathbf{S}_i)^{-1}(\mathbf{I} - \gamma_{i+1,i+1}\boldsymbol{\Gamma}_i)\mathbf{s}_i.
\end{aligned}$$

Notice here that the expression for $\gamma_{i+1,i+1}$ has two solutions, the positive and negative square root. These elements are the diagonal entries in $\mathbf{A}$, thus requiring that the sign of the diagonal be specified. If it is not specified, there are 2^p possible solutions. Once the sign of these elements is specified, however, we have a unique solution for $\mathbf{A}$ given $\boldsymbol{\Sigma}$ and $\mathbf{D}$. However, there still exist positive definite $\boldsymbol{\Sigma}$ that do not map back to a valid $\mathbf{A}$ given $\mathbf{D}$. As a simple example, if $\sigma_{ii} < d_{ii}$, then the model will not work since that fact means that the equilibrium variance is less than the error variance added at each step.

Now I want to show that the MLE for $\mathbf{A}$ in this model is both consistent and efficient. Although this may seem straightforward given the parametric nature of the model, I want to ensure that nothing comes up due to the infinite sum in the variance of the distribution.

Theorem 5.3.2. *Suppose p-dimensional* $\mathbf{x}_1, \ldots, \mathbf{x}_n$ *are observed from the distribution*

$$X \sim N\left(0, \sum_{i=0}^{\infty} \mathbf{A}^i \mathbf{D} (\mathbf{A}^T)^i\right),$$

where $\mathbf{D}$ *is known and* $\mathbf{A}$ *is lower-diagonal with the sign of the diagonal elements known and value of the diagonal elements less than 1 in absolute value. Then* $\hat{\mathbf{A}}$*, the MLE for* $\mathbf{A}$ *is asymptotically consistent and efficient with*

$$\sqrt{n}(\hat{\mathbf{a}} - \mathbf{a}) \to N(0, \mathcal{F}(\mathbf{a})^{-1}),$$

where $\mathbf{a}$ *is the vectorization of the non-zero elements of* $\mathbf{A}$ *and* $\mathcal{F}(\mathbf{a})$ *is the Fisher information, whose elements are*

$$\mathcal{F}(\mathbf{a})_{ij} = E\left[\left(\frac{\partial}{\partial a_i} \log f(\mathbf{x}|\mathbf{A})\right)\left(\frac{\partial}{\partial a_j} \log f(\mathbf{x}|\mathbf{A})\right)\right].$$

Proof. Appendix A derives the first and second partials and shows that they exist and are continuous, satisfying condition 2. For example, the first partial derivatives take the form

$$\frac{\partial \sigma_{ij}}{\partial a_{mn}} = \frac{1}{1 - a_{ii}a_{jj}} \left[\begin{array}{c} (\mathbf{A\Sigma})_{in}\mathbf{1}_{[j=m, j\geq n]} + (\mathbf{A\Sigma})_{jn}\mathbf{1}_{[i=m, i\geq n]} \\ + \sum_{k=1}^{i}\sum_{l=1}^{j} a_{ik}a_{jl}\frac{\partial \sigma_{kl}}{\partial a_{mn}}\mathbf{1}_{[(k,l)\neq(i,j)]} \end{array} \right].$$

Both the first and second partial derivatives can be calculated iteratively.

Additionally, let

$$\begin{aligned} \mathbf{\Psi}(x, \mathbf{a}) &= \frac{\partial}{\partial \mathbf{a}} \log f(x|\mathbf{a}), \qquad \text{a } k \text{ vector,} \\ \dot{\mathbf{\Psi}}(x, \mathbf{a}) &= \frac{\partial^2}{\partial \mathbf{a}^2} \log f(x|\mathbf{a}), \qquad \text{a } k \times k \text{ matrix.} \end{aligned}$$

Note that the components of $\dot{\mathbf{\Psi}}(x, \mathbf{a})$, as shown in Appendix A, are all of the form

$$\sum_{i,j} c_{ij} (\mathbf{x}\mathbf{x}^T)_{ij},$$

where the c_{ij}'s can be complex to calculate, but constant. Further, taking expectation of the absolute value of the components is finite since $E[|\mathbf{x}\mathbf{x}^T|_{ij}] < \infty$ (Li and Wei, 2012). Thus, take

$$K(\mathbf{x}) = \sum_{i,j} m_{ij} |\mathbf{x}\mathbf{x}^T|_{ij},$$

where m_{ij} is the maximum of all the $|c_{ij}|$'s across all components. Then each component of $\dot{\mathbf{\Psi}}(x, \mathbf{a})$ is bounded by $K(\mathbf{x})$ in absolute value and $K(\mathbf{x})$ has finite expectation.

These results, along with the earlier proof of identifiability and the form of the constraints on $\mathbf{A}$, satisfy Cramér's Theorem as found in Ferguson (1996). Thus the MLE for $\mathbf{A}$ is asymptotically consistent and efficient. □

In addition to asymptotic consistency and efficiency, we also can form a likelihood-ratio test for the existence of an edge.

Corollary 5.3.3. *Let $\mathcal{A}_1$ be the space of all valid $\mathbf{A}$ in the VAR equilibrium model, possibly with some entries constrained to be 0. Let $\mathcal{A}_0 \subset \mathcal{A}_1$ with a difference in dimension between the two spaces of c. This corresponds to c additional entries of $\mathbf{A}$ constrained to be 0. Then the log-likelihood ratio test statistic*

$$\lambda = -2 \log \left(\frac{\sup_{\mathbf{A} \in \mathcal{A}_0} L(\mathbf{A}|\mathbf{x})}{\sup_{\mathbf{A} \in \mathcal{A}_1} L(\mathbf{A}|\mathbf{x})} \right)$$

has a χ^2_c distribution.

Proof. The conditions which satisfy Cramér's theorem also satisfy Wilk's theorem concerning the likelihood-ratio test statistic. □

5.3.2 Simulation Results - Asymptotic Variance

I have an asymptotic distribution for the MLE, and now I would like to check it via simulation. Because of the complexity of the calculations involved in finding the Fisher information, I will restrict myself to a 2-dimensional model. This model has three parameters for $\mathbf{A}$,

$$\mathbf{A} = \begin{bmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{bmatrix},$$

so the Fisher information matrix will be 3 by 3, with elements given by

$$F(\mathbf{A})_{ij} = E\left[\left(\frac{\partial}{\partial a_i}\log f(\mathbf{x}|\mathbf{A})\right)\left(\frac{\partial}{\partial a_j}\log f(\mathbf{x}|\mathbf{A})\right)\right].$$

For my simulations, I used

$$\begin{aligned}\mathbf{A} &= \begin{bmatrix} 0.9 & 0 \\ 0.7 & 0.8 \end{bmatrix}, \\ \mathbf{D} &= \mathbf{I}_2.\end{aligned}$$

Using these values, the asymptotic variance of the MLE for $\mathbf{A}$ is calculated to be

$$F(\mathbf{A})^{-1} = \begin{bmatrix} 0.022 & -0.126 & 0.029 \\ -0.126 & 3.094 & -0.850 \\ 0.029 & -0.850 & 0.251 \end{bmatrix}.$$

To test this, I performed a simulation where I generated n samples from the equilibrium distribution. I then found the value of $\hat{\mathbf{A}}$ that maximized the likelihood. To do this, I initialized $\mathbf{A}$ using the recursive algorithm described above and used `optim` in `R` to optimize the log-likelihood of the model. The `optim` function allows the user to specify the method used. By default, it uses a method developed by Nelder and Mead (1965). It also allows optimization using the quasi-Newton method developed in (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970), along with the ability to define constraints for the method (Byrd et al., 1995). I experimented with the different optimization methods and found that the results were not sensitive to the method used. I repeated the initialization and optimization procedure 1000 times and used the resulting $\hat{\mathbf{A}}$ values to calculate an empirical variance. I used three different sample sizes n, resulting in the following empirical variances:

$n = 1000$,

$$Var(\hat{\mathbf{A}}) = \begin{bmatrix} 0.021 & -0.077 & 0.009 \\ -0.077 & 2.633 & -0.717 \\ 0.009 & -0.717 & 0.219 \end{bmatrix},$$

$n = 10000$,

$$Var(\hat{\mathbf{A}}) = \begin{bmatrix} 0.023 & -0.074 & 0.007 \\ -0.074 & 2.705 & -0.746 \\ 0.007 & -0.746 & 0.230 \end{bmatrix},$$

$n = 100000$,

$$Var(\hat{\mathbf{A}}) = \begin{bmatrix} 0.023 & -0.091 & 0.012 \\ -0.091 & 2.757 & -0.737 \\ 0.012 & -0.737 & 0.220 \end{bmatrix},$$

The results validate the theory in that, although not perfect, they do capture the magnitude and sign of the components.

5.3.3 Simulation Results - Coverage

Another way to verify that the derivation works in practice is to look at coverage of confidence intervals around the MLE. To do this, I used the same 2-dimensional model as above. In a single simulation, a_{11} and a_{22} are chosen at random from a Uniform(0.1, 0.9) distribution and a_{21} is chosen at random from a Uniform(-0.9, 0.9) distribution. I then generate n samples from the equilibrium distribution and use those samples to obtain an MLE for $\mathbf{A}$ by maximiing the log-likelihood as before. I can then use the empirical asymptotic variance of the MLE to obtain confidence intervals and see if the true values are covered. I repeat this 1000 times with the same $\mathbf{A}$ to get coverage results.

I ran this procedure 10 times each for n equal to 100, 1000 and 10000, generating a different $\mathbf{A}$ each time. The coverage results are shown in Figures 5.5-5.8. The x-axis is the true value of a_{11}. This value appears to matter more than the other $\mathbf{A}$ values because it propagates to all the values of $\mathbf{\Sigma}$. As n increases, the coverage results definitely seem to get better. With small n and a_{11}, the observed variance of x_1 can be smaller than that of the known $\mathbf{D}$. This can cause problems with optimizing $\mathbf{A}$. But it does seem that the coverage is correct for large n, both marginally and jointly.

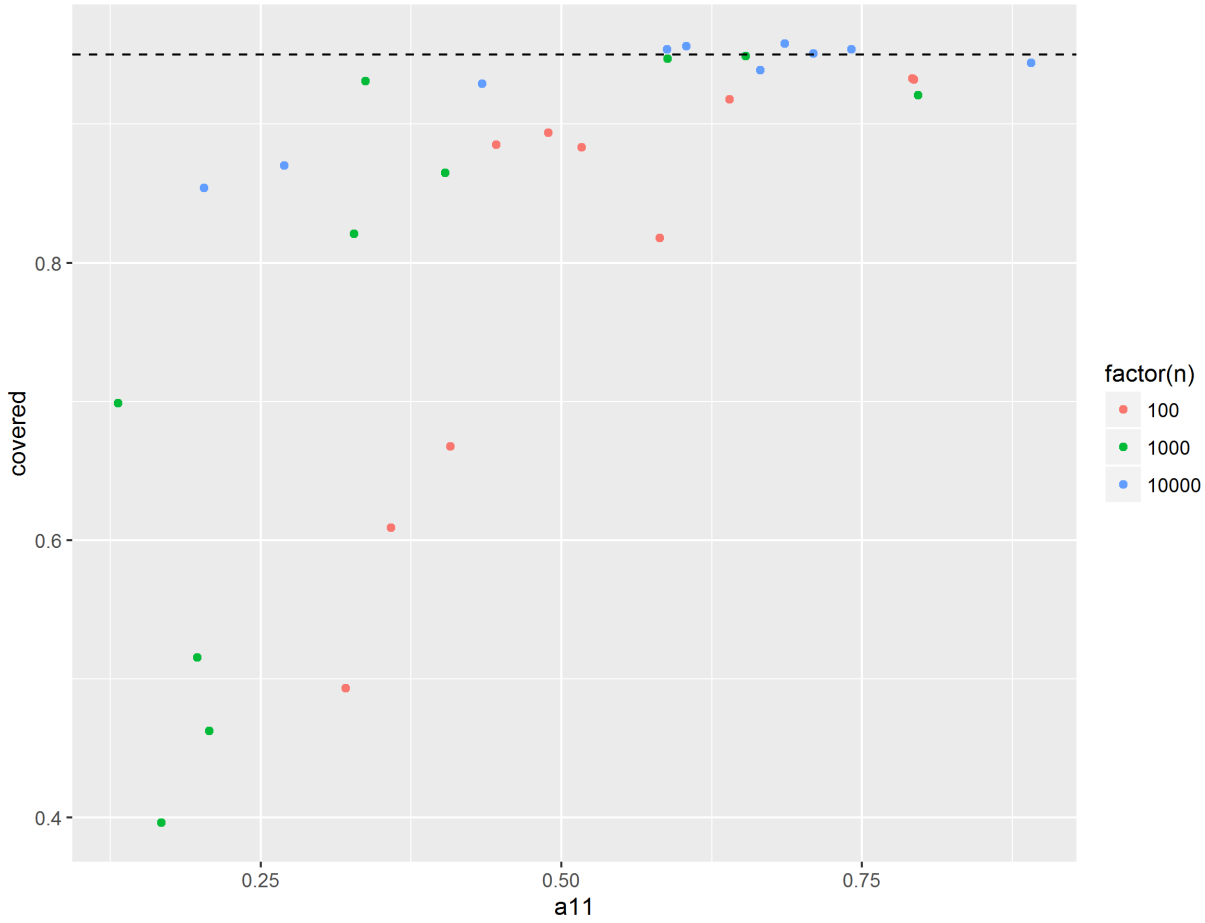

Figure 5.5: Joint coverage for all parameters of $\mathbf{A}$ using the 95% confidence interval from the multivariate normal asymptotic distribution of the MLE.

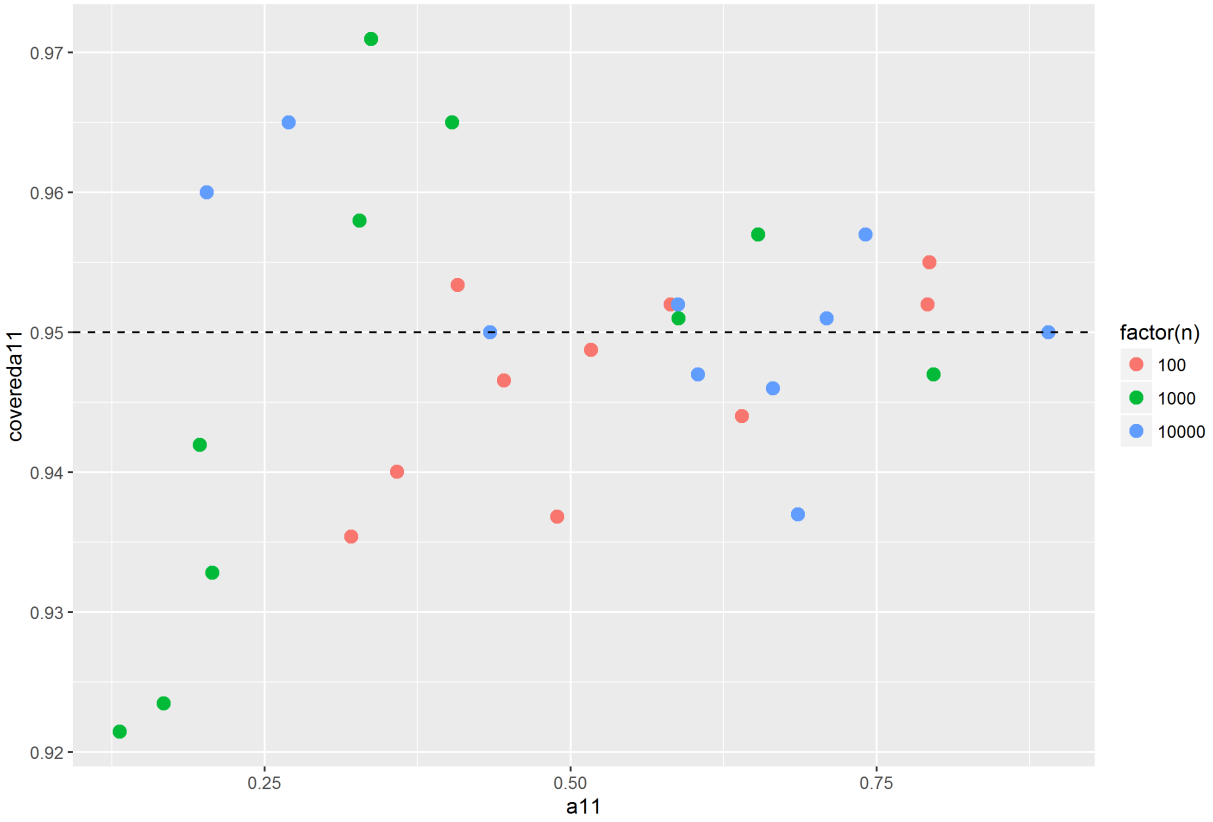

Figure 5.6: Marginal coverage for a_{11} using the 95% confidence interval from the asymptotic distribution of the MLE.

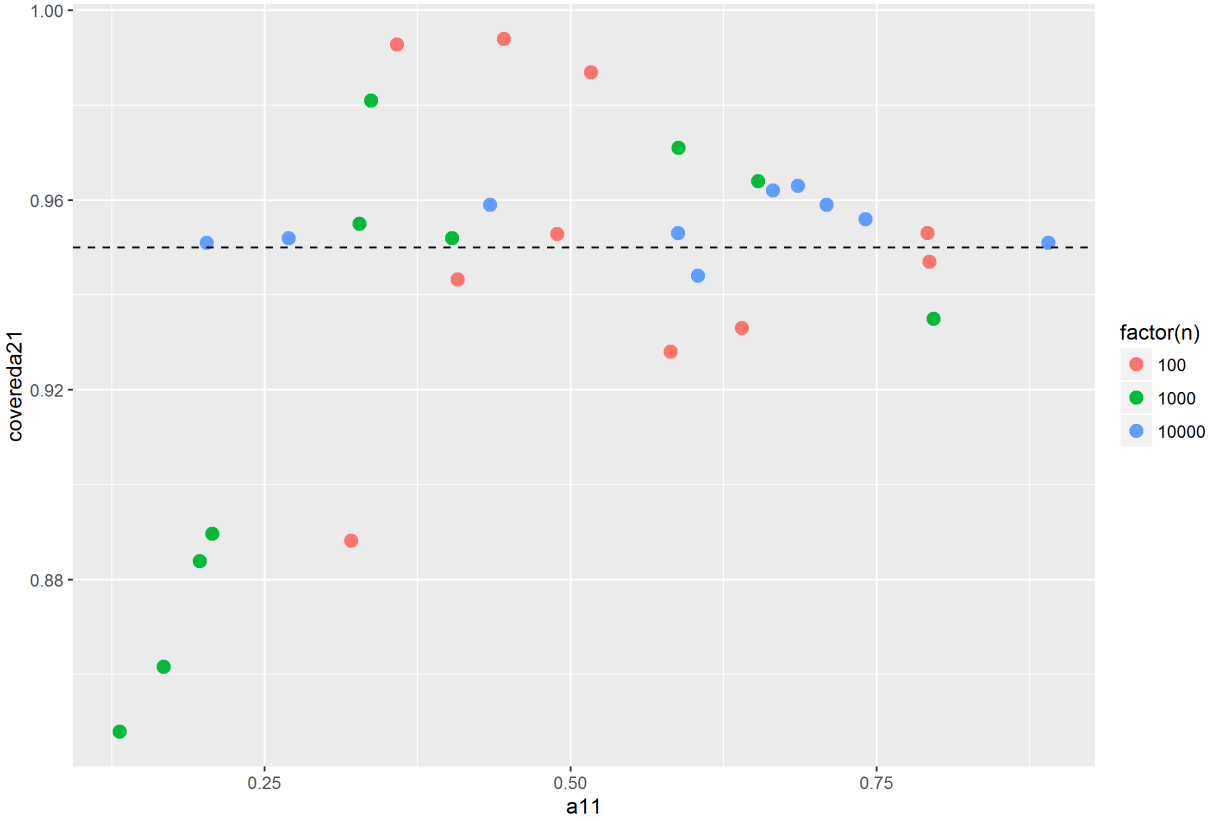

Figure 5.7: Marginal coverage for a_{21} using the 95% confidence interval from the asymptotic distribution of the MLE.

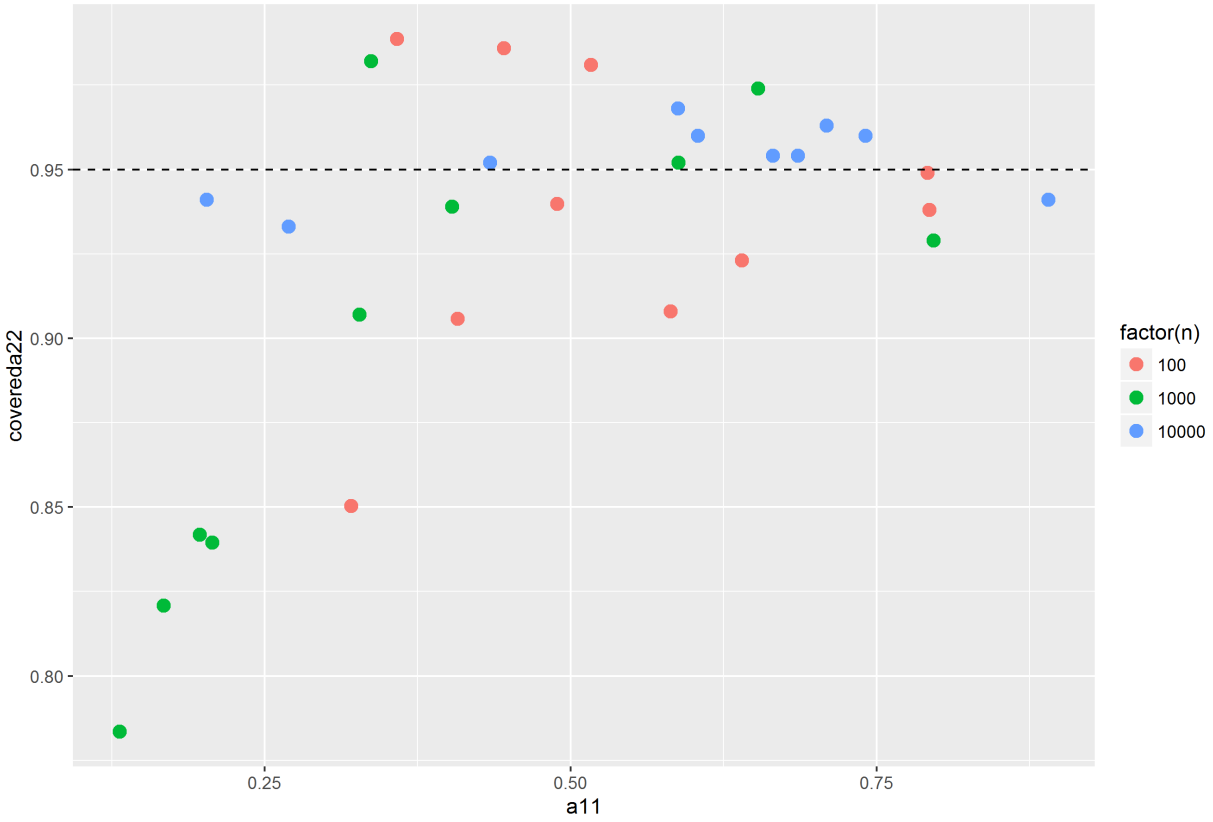

Figure 5.8: Marginal coverage for a_{22} using the 95% confidence interval from the asymptotic distribution of the MLE.

5.4 *Likelihood Ratio Test*

I would also like to verify the validity of the likelihood-ratio test for the simple VAR equilibrium model. To test this, I used a 3-dimensional system with

$$\mathbf{A} = \begin{bmatrix} 0.6 & 0 & 0 \\ -0.3 & 0.4 & 0 \\ 0 & -0.5 & 0.8 \end{bmatrix}.$$

For this system, $\mathcal{A}_0$ has the correct constraint that the lower-left element is zero while $\mathcal{A}_1$ does not have that constraint. For a dataset generated from the model, I find the MLE under both hypotheses and construct the likelihood ratio test statistic.

To get the empirical distribution of the test statistic, I generated 10,000 samples from the equilibrium distribution and calculated the likelihood ratio for that data. I repeated this 1,000 times to get a distribution of the statistic values. Figure 5.9 shows the distribution compared with a χ^2 distribution with one degree of freedom. The distributions are close, indicating that the theory does in fact hold up in this simulated example.

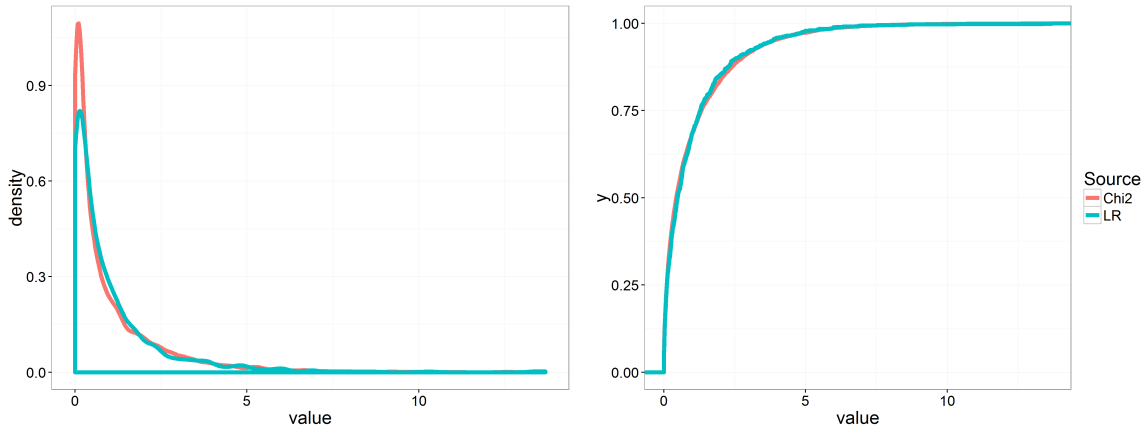

Figure 5.9: Comparison of the empirical distribution of the likelihood ratio statistic to test for an additional edge in the VAR equilibrium problem with a χ^2 distribution with one degree of freedom. The left-hand plot shows the density of the test statistic compared with the χ^2 distribution while the right shows the empirical CDF comparison.

5.5 Application to Real Data

I have shown that the theoretical properties of the simple VAR equilibrium model are upheld in a simulation setting. What happens when we try to apply the method outside of its theoretically guaranteed setting? Knowing $\mathbf{D}$ a priori is not likely in an real data setting, and in gene networks the graph is not necessarily restricted to be acyclic. That the general VAR equilibrium model does not require an acyclic graph is a potential advantage over Bayesian network methods, which require a directed acyclic graph.

5.5.1 GeneNetWeaver Data

To look at the method in a more realistic setting, I need a reasonable size network where the truth is known. A good candidate is the networks from the DREAM4 competition. The 10-gene networks from the competition are a good size for analysis, include cycles, and are known. The original data used in the competition includes time series data simulating a perturbation to the network. From the documentation, the first and last time points correspond to a stable equilibrium for the network, so perhaps these data could be used. However, the original data does not include enough separate time series from which to get samples.

However, the software used to generate the data, GeneNetWeaver, is available online at `http://gnw.sourceforge.net/` (Marbach et al., 2009; Schaffter et al., 2011). The software can be configured to generate any number of independent time series from a given network model. The data is generated according to a dynamical model using ordinary differential equations with added stochastic noise. For the time series data, the network begins at equilibrium and then the network is artificially perturbed for the first half of the time course. At the midpoint of the time course, the perturbation is removed and the network is allowed to return to its equilibrium state by the end of the time course. The GeneNetWeaver software is designed to provide a rich simulation of the dynamics of biological regulatory networks. The ODEs used include terms simulating transcription and translation rates as well as protein

degradation rates and simulates mRNA levels and protein levels simultaneously. Noise is added to the system to simulate both random fluctuations in the actual levels of mRNA and protein as well as measurement error.

I used GeneNetWeaver to generate time series data using the network model used in the first 10-gene network from the DREAM4 competition. The GeneNetWeaver software has this network available as a pre-configured model from which data can be generated. The network has 10 nodes and 15 non-self edges, including a cycle involving three genes. All settings for data generation were the same as those used in the DREAM4 competition, including the amount of added noise to the system and the model of noise in microarrays. I used GNW to generate 100 random time series of 21 points each from the model and took the first and last timepoints as samples from the equilibrium distribution.

5.5.2 Method Application

To see how the VAR equilibrium method works on this data, I used the known network structure as my initial model for $\mathbf{A}$. I did not assume a known $\mathbf{D}$, but included it as additional parameters for the model. This resulted in a total of 35 parameters for the model - 10 from $\mathbf{D}$, 10 from the diagonal of $\mathbf{A}$, and 15 off-diagonal elements of $\mathbf{A}$ reflecting the 15 non-self edges in the network. $\mathbf{\Sigma}$ has 55 parameters, so this is well within the parameter counting requirement. This does not guarantee identifiability, but it is a reasonable assumption.

To find the MLE for the model, I randomly initialize $\mathbf{A}$ and $\mathbf{D}$. The diagonal elements of $\mathbf{A}$ are initialized randomly from a Uniform(0,0.9) distribution to reflect the belief that the autoregressive parameter for any gene on itself should be positive. The off-diagonal elements are initialized from a Uniform(-0.9, 0.9) distribution. A requirement on $\mathbf{D}$ is that each element must be less than the variance of the corresponding gene in the equilibrium data, so each element is initialized to be the variance of the corresponding gene times a draw from a Uniform(0.1, 0.9) distribution.

After $\mathbf{A}$ and $\mathbf{D}$ are initialized, the log-likelihood of the data is optimized using the `optim` method in `R`. The optimization scheme may get stuck in a local mode since the likelihood is

not convex in **A** and **D**, so the random initialization and optimization was repeated 1,000 times to provide a better search of the model space. The best **A** and **D** are kept as the approximate MLE.

This gives an estimate of the parameters for the true model, but in order to test individual edges, I need to find the MLE for models adjacent to the true model. These models are defined by taking the original network structure and adding or removing a single edge. When removing edges, I need to ensure that the connectivity of the graph is not broken. 3 of the edges in the original network, if removed, would isolate a single node. This leaves 12 of the original 15 edges available to be tested. Adding an edge is not a problem, and thus I can test all 75 extra edges. With the results from all 88 models, I can look at the likelihood ratio test to see which edges it identifies as significant.

Running the above optimization scheme for each of the 88 models results in an approximate MLE for each model. Because these optimizations are run independently for each model and the model space is not necessarily fully searched, this results in some mis-ordering of the models. That is, the MLE found for one model results in a likelihood that is worse than that of a model in which some of the parameters of the first model are constrained to be zero. This is never allowed since the parameters of the smaller model can be used for the larger model, resulting in the same likelihood.

To fix the ordering issue, I perform a second optimization step for each target model. I initialize `optim` with the parameter values from each of the models. This may involve throwing out parameter values for extra edges or having zeroes from smaller models. The model is then optimized from that starting point. Further, if there are zeroes from a smaller model, I re-randomize just the zeros a small number of times to give better search coverage. This iteration scheme may not fix the ordering issues in a single pass, but after a few passes the models will be appropriately ordered.

5.5.3 Results

I applied the optimization scheme described above to both the initial and final timepoint data from the generated data from GeneNetWeaver. For each model explored, I obtained the best log-likelihood value for comparison. I then compared the true model with each other model, either with one less or one more edge, and computed the likelihood ratio statistic for testing the edge. From the likelihood ratio statistic I computed a p-value by comparing the value to a χ^2 distribution with one degree of freedom.

Ideally, the p-values corresponding to the 12 edges which are in the true network would be low and the p-values for the 75 edges added to the true model would be high. Figure 5.10 shows the results for both the first and the last timepoint. Looking at the red dots, we find that 7 of the 150 edges added to the true network were identified as true edges at $p = 0.05$. This is a Type I error rate of 5%, which is in line with what we expect. 5 of the 24 true edges which were tested were identified as true edges, yielding a power of 21%. This is not an overwhelming result, but statistical tests often do not have great power.

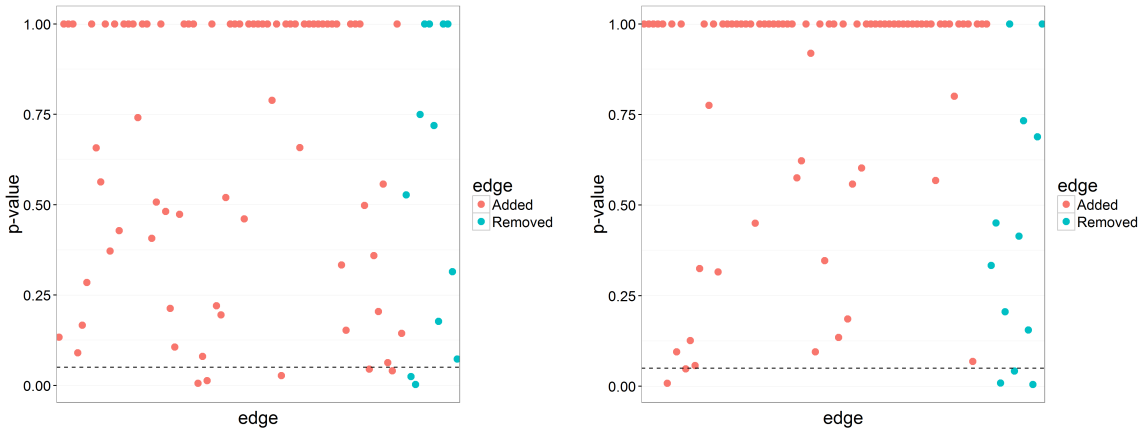

Figure 5.10: p-values for each tested edge in the GeneNetWeaver data. Red dots correspond to edges added to the true network (false edges) and blue dots correspond to edges removed from the true network (true edges). The left figure is using data from the first timepoint and the right figure is using data from the last timepoint. The dashed line corresponds with a p-value of 0.05.

There are multiple reasons that the results from analyzing the GeneNetWeaver data might not be optimal. First of all, the VAR model I use is not the true generating model. The GeneNetWeaver data is generated from a system of differential equations rather than a simple vector autoregressive model. This mismatch may cause problems, although the ScanBMA analysis of the DREAM4 data showed that the time series model is still able to identify true relationships. The equilibrium variance of the data may still be affected significantly, however, by the differences.

A second problem is that the optimization scheme for finding the MLE may not be very good. It may be the case that the best value for $\mathbf{A}$ and $\mathbf{D}$ are not found and the LR test statistic is based on local modes for the likelihood rather than the global maximum.

To test this idea, I simulated data starting with the network structure used in the GeneNetWeaver data. First, I generated a random $\mathbf{A}$ as I did in the search for the MLE and a random $\mathbf{D}$ with values on the diagonal sampled from a Uniform(0.1,0.9) distribution. With this $\mathbf{A}$ and $\mathbf{D}$ I calculated $\mathbf{\Sigma}$, the asymptotic variance of the VAR model. I generated 100 samples from the asymptotic distribution to parallel the data from GeneNetWeaver. I then attempted to find the MLE for $\mathbf{A}$ and $\mathbf{D}$ as with the GeneNetWeaver data by randomly initializing $\hat{\mathbf{A}}$ and $\hat{\mathbf{D}}$ 1,000 times and using the `optim` function to find the best likelihood starting from those initial positions.

I simulated data and optimized the likelihood in this manner with 10 randomly generated $\mathbf{A}$ and $\mathbf{D}$ to see how often good values for $\hat{\mathbf{A}}$ and $\hat{\mathbf{D}}$ are found. To compare, I started the optimization from the true $\mathbf{A}$ and $\mathbf{D}$. Let these optimal values be denoted $\mathbf{A}^*$ and $\mathbf{D}^*$. I also computed the optimal log-likelihood in the unconstrained model. In that case, I directly calculated $\mathbf{\Sigma}$ as the variance of the data. If the optimization method is good, then the $\hat{\mathbf{A}}$ and $\hat{\mathbf{D}}$ should yield a likelihood that is at least as good as $\mathbf{A}^*$ and $\mathbf{D}^*$.

Table 5.1 compares the log-likelihood values found by the different methods for the 10 random data sets. In some cases, the random initialization scheme does as good as or better than the smart initialization of $\mathbf{A}$ and $\mathbf{D}$, but in other cases it does very poorly. Using the smart initialization does not yield a likelihood as good as using the best possible $\mathbf{\Sigma}$, but

that is to be expected since we are working with a constrained model. I also calculated the correlation between the values of the parameters found via optimization with the true parameters. The correlation is positive in all cases, which is good, but varies greatly from one test to the next.

Finding the optimal **A** and **D** is not an simple problem. This simulation result shows that even when the data is generated from the VAR equilibrium model, it is not guaranteed that the optimal solution will be found.

Table 5.1: Log-likelihoods of the 10 generated datasets of 100 observations from the VAR equilibrium distribution, using randomly chosen **A** and **D**. 'Random' refers to the optimal model found by 1,000 replications of starting at a random **A** and **D**. 'Smart' refers to starting the optimization from the true model. The 'Best' column shows the best possible log-likelihood obtained by using the empirical variance of the data. The 'Cor' column shows the correlation between the inferred values from the random optimization and the true values of **A** and **D**.

Random	**Smart**	**Best**	**Cor**
1323	1332	1309	0.74
1901	1710	1693	0.41
1584	1545	1527	0.51
1176	1180	1162	0.42
1607	1553	1528	0.69
1465	1456	1442	0.66
1783	1608	1589	0.25
1539	1520	1499	0.20
1414	1420	1400	0.53
1261	1263	1246	0.55

5.6 Discussion

Steady-state gene expression data presents a challenge in inferring directional edges by its very nature. I have presented a model which can be used to test for the existence of edges in such data. Although the model has good theoretical properties in the constrained case, it is difficult to extend them to more general cases, in large part due to issues of identifiability. It is in general difficult to find conditions under which $\mathbf{A}$ and $\mathbf{D}$ are identifiable.

The identifiability problem is also found in structural equation modeling (SEM) (Bentler and Weeks, 1980). SEM is similar to a VAR model in that it specifies relationships among a set of variables. SEM, however, relates the variables without considering the time element. If $\mathbf{X}$ is the collection of random variables observed, then a linear structural equation model with Gaussian noise can be written as

$$\begin{aligned} \mathbf{X} &= \mathbf{A}\mathbf{X} + \boldsymbol{\epsilon}, \\ \boldsymbol{\epsilon} &\sim N(0, \mathbf{D}). \end{aligned}$$

Seeking identifiability conditions has been the subject of a number of papers (Drton et al., 2011; Brito and Pearl, 2012; Drton and Weihs, 2015), and general conditions for this class of models are not known.

I have shown that even in cases where the theory has not been validated, the VAR equilibrium method can still identify true relationships among genes. This is evident in the analysis of the GeneNetWeaver data, where true edges were identified more consistently than extraneous edges.

The VAR equilibrium method is not applicable in all situations. If the network is partially known, then the likelihood ratio statistic can be used to test for extra edges and thus build up a more comprehensive view of the network. As an example, we may want to learn about a network perturbed with a certain drug. If the unperturbed network is known, we can test edges in and around that network using the perturbation data to learn about the changes induced by the drug. Since we may not expect the entire network to change, the VAR equilibrium method takes advantage of prior knowledge about the network structure.

To apply the VAR equilibrium method to steady-state data such as that in the LINCS L1000 data will take further development. One way to go about this would be to take a set of genes and look at all possible network structures for those genes. For each model, the steady-state data would be used to maximize the likelihood. One could then combine the models using a Bayesian model averaging type of approach, similar to the that used in Chapter 2. To do this, the integrated likelihood needs to be approximated for each model. As a starting point, BIC could be used. The BIC is equal to $-2 \log \hat{L} + k \log n$, where $\hat{L}$ is the maximized likelihood for the model, k would be equal to $2p$ plus the number of edges in the network, and n is the number of observations. Averaging over all models would result in a posterior probability for each edge in the network.

Chapter 6

DISCUSSION AND FUTURE WORK

In this thesis, I looked at different approaches to inferring gene regulatory networks from gene expression data. The various methods were motivated by the different types of expression data that are commonly available: time series, knockdown, and perturbation and other steady-state data. In chapter 2 I described a new algorithm, ScanBMA, that searches the model space for Bayesian model averaging faster and more thoroughly than previous methods. I showed how it can be used to improve on previous results when applied to yeast time series data as well as how it gives comparable results to other popular methods on the DREAM4 results while producing networks that are similar in size to the truth.

Chapter 3 explains a simple Bayesian method applied to knockdown experiments in the LINCS L1000 data to detect causal relationships between genes. The method performs better than competing methods on the L1000 knockdown data. A key feature of both this method and ScanBMA from chapter 2 is the ability to incorporate informative prior information. This was particularly helpful in analyzing the yeast data. Unfortunately, the information about the human genome is not as complete, and no informative prior was available. When such a prior is available, I expect that results would improve significantly.

In chapter 4 I took a closer look at the L1000 data and artifacts introduced from the experimental setup and processing pipeline. I developed an extension of model-based clustering, MCDC, that can identify and correct the artifacts in the data. I showed how MCDC works as expected in a simulation setting. When applied to the L1000 data, MCDC improved the data in comparison with external reference data. Applying the posterior probability method from chapter 3 to the corrected data also yielded improvements in regulatory relationship inference with respect to the TRANSFAC & JASPAR reference standard.

MCDC is potentially applicable to more than just the L1000 data and artifacts. It is, however, specific in that the transformation of the data which is applied to the data must be known. It is possible that multiple discrete transformations can be compared with each other. To do this, the BIC of the best model for each transformation could be compared, and the best transformation would be identified.

Chapter 5 presented an approach to analyzing steady-state data by considering it as coming from the equilibrium distribution of a VAR process. Proofs were given of consistency and efficiency of the MLE in a restricted case, and a test for a single edge using the likelihood ratio statistic was proposed. The theoretical results were validated using simulation. The method was applied to simulated data from GeneNetWeaver, a tool for creating benchmark data for testing gene network inference.

The theoretical results for the VAR equilibrium method are quite limited in scope. It would be helpful to extend the results to more general cases. One strong constraint that should be relaxed is that $\mathbf{D}$, the error variance from the time-series model, is known. In general this will not be the case and $\mathbf{D}$ will need to be estimated. It would also be helpful to have theoretical results that are applicable to the cyclic case. One of the strengths of the VAR equilibrium model is that the network to be estimated is not necessarily acyclic. Both of these generalizations seem to be reasonable according to the parameter counting requirement, but I do not currently have any proofs of identifiability in the more general cases.

Another current limitation of the VAR equilibrium method is that it is slow in finding the MLE in even small networks of 10 genes. The software for optimization is currently written in `R`. The current brute force optimization method could be refined to try to find the MLE faster, and the code itself could be rewritten in a compiled language such as `C++` to improve the speed.

Going forward, there is potential for using a BMA type of approach for applying the VAR equilibrium method to steady-state data such as that found in the LINCS L1000 data. This will require a method for proposing good models and rely on improvement in the speed

of optimizing the likelihood. The VAR equilibrium method could also be used to improve inference in conjunction with other methods which use other data types. It could be used to score network models proposed from other data, similar to the work done by Shojaie et al. (2014).

BIBLIOGRAPHY

Anderson, T. W. (2000). A note on a vector-variate normal distribution and a stationary autoregressive process. *Journal of Multivariate Analysis*, 72:149–150.

Ball, C. A., Sherlock, G., Parkinson, H., Rocca-Sera, P., Brooksbank, C., Causton, H. C., Cavalieri, D., Gaasterland, T., Hingamp, P., Holstege, F., et al. (2002). Standards for microarray data. *Science*, 298:539–539.

Banfield, J. D. and Raftery, A. E. (1993). Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49:803–821.

Bansal, M., Della Gatta, G., and Di Bernardo, D. (2006). Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22:815–822.

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

Basu, S., Shojaie, A., and Michailidis, G. (2015). Network granger causality with inherent grouping structure. *Journal of Machine Learning Research*, 16:417–453.

Bélisle, C. J. (1992). Convergence theorems for a class of simulated annealing algorithms on Rd. *Journal of Applied Probability*, 29:885–895.

Bentler, P. M. and Weeks, D. G. (1980). Linear structural equations with latent variables. *Psychometrika*, 45:289–308.

Biernacki, C., Celeux, G., and Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22:719–725.

Biernacki, C., Celeux, G., and Govaert, G. (2003). Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. *Computational Statistics and Data Analysis*, 41:561–575.

Binder, H. and Preibisch, S. (2008). "Hook"-calibration of Genechip-microarrays: Theory and algorithm. *Algorithms for Molecular Biology*, 3:1–25.

Blocker, A. W., Meng, X.-L., et al. (2013). The potential and perils of preprocessing: Building new foundations. *Bernoulli*, 19:1176–1211.

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

Bottolo, L., Richardson, S., and others (2010). Evolutionary stochastic search for Bayesian model exploration. *Bayesian Analysis*, 5:583–618.

Brito, C. and Pearl, J. (2012). Graphical condition for identification in recursive SEM. *arXiv preprint arXiv:1206.6821*.

Broyden, C. G. (1970). The convergence of a class of double-rank minimization algorithms 1. general considerations. *IMA Journal of Applied Mathematics*, 6:76–90.

Bühlmann, P., Kalisch, M., and Maathuis, M. H. (2010). Variable selection in high-dimensional linear models: partially faithful distributions and the PC-simple algorithm. *Biometrika*, page asq008.

Bühlmann, P., Kalisch, M., and Meier, L. (2014). High-dimensional statistics with a view toward applications in biology. *Annual Review of Statistics and Its Application*, 1:255–278.

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16:1190–1208.

Campbell, J. G., Fraley, C., Stanford, D., Murtagh, F., and Raftery, A. E. (1999). Model-based methods for textile fault detection. *International Journal of Imaging Systems and Technology*, 10:339–346.

Celeux, G. and Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition*, 28:781–793.

Chen, B., Greenside, P., Paik, H., Sirota, M., Hadley, D., and Butte, A. (2015). Relating chemical structure to cellular response: An integrative analysis of gene expression, bioactivity, and structural data across 11,000 compounds. *CPT: Pharmacometrics & Systems Pharmacology*, 4:576–584.

Chen, C., Grennan, K., Badner, J., Zhang, D., Gershon, E., Jin, L., and Liu, C. (2011). Removing batch effects in analysis of expression microarray data: an evaluation of six batch adjustment methods. *PloS one*, 6:e17238.

Chen, E. Y., Tan, C. M., Kou, Y., Duan, Q., Wang, Z., Meirelles, G. V., Clark, N. R., and Ma'ayan, A. (2013a). Enrichr: interactive and collaborative HTML5 gene list enrichment analysis tool. *BMC Bioinformatics*, 14:128.

Chen, J., Hu, Z., Phatak, M., Reichard, J., Freudenberg, J. M., Sivaganesan, S., and Medvedovic, M. (2013b). Genome-wide signatures of transcription factor activity: connecting transcription factors, disease, and small molecules. *PLoS Computational Biology*, 9:article e1003198.

Cho, H., Berger, B., and Peng, J. (2016). Reconstructing causal biological networks through active learning. *PloS One*, 11:e0150611.

Christley, S., Nie, Q., and Xie, X. (2009). Incorporating existing network information into gene network inference. *PLoS One*, 4:article e6799.

Clyde, M. and George, E. I. (2004). Model uncertainty. *Statistical Science*, pages 81–94.

Cooke, E. J., Savage, R. S., Kirk, P. D., Darkins, R., and Wild, D. L. (2011). Bayesian hierarchical clustering for microarray time series data with replicates and outlier measurements. *BMC Bioinformatics*, 12:article 399.

Crick, F. and others (1970). Central dogma of molecular biology. *Nature*, 227:561–563.

Crick, F. H. (1958). On protein synthesis. In *Symposia of the Society for Experimental Biology*, volume 12, page 138.

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

D'haeseleer, P., Wen, X., Fuhrman, S., Somogyi, R., and others (1999). Linear modeling of mRNA expression levels during CNS development and injury. In *Pacific Symposium on Biocomputing*, volume 4, pages 41–52. World Scientific.

Ding, C. and Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3:185–205.

Drton, M., Foygel, R., and Sullivant, S. (2011). Global identifiability of linear structural equation models. *The Annals of Statistics*, 39:865–886.

Drton, M. and Weihs, L. (2015). Generic identifiability of linear structural equation models by ancestor decomposition. *arXiv preprint arXiv:1504.02992*.

Duan, Q., Flynn, C., Niepel, M., Hafner, M., Muhlich, J. L., Fernandez, N. F., Rouillard, A. D., Tan, C. M., Chen, E. Y., Golub, T. R., and others (2014). LINCS Canvas Browser: interactive web app to query, browse and interrogate LINCS L1000 gene expression signatures. *Nucleic Acids Research*, page gku476.

Dunbar, S. A. (2006). Applications of Luminex® xMAP™ technology for rapid, high-throughput multiplexed nucleic acid detection. *Clinica Chimica Acta*, 363:71–82.

Ellefsen, K. J., Smith, D. B., and Horton, J. D. (2014). A modified procedure for mixture-model clustering of regional geochemical data. *Applied Geochemistry*, 51:315–326.

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

Faith, J. J., Hayete, B., Thaden, J. T., Mogno, I., Wierzbowski, J., Cottarel, G., Kasif, S., Collins, J. J., and Gardner, T. S. (2007). Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biology*, 5:e8.

Ferguson, T. S. (1996). *A course in large sample theory*, volume 49. Chapman & Hall London.

Fletcher, R. (1970). A new approach to variable metric algorithms. *The Computer Journal*, 13:317–322.

Flynt, A. and Daepp, M. I. (2015). Diet-related chronic disease in the northeastern United States: a model-based clustering approach. *International Journal of Health Geographics*, 14:1–14.

Fraley, C. and Raftery, A. E. (1998). How many clusters? Which clustering method? Answers via model-based cluster analysis. *Computer Journal*, 41:578–588.

Fraley, C. and Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97:611–631.

Fraley, C. and Raftery, A. E. (2007). Model-based methods of classification: Using the mclust software in chemometrics. *Journal of Statistical Software*, 18:1–13.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33:1–22.

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

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601–620.

Fröhlich, H., Fellmann, M., Sültmann, H., Poustka, A., and Beißbarth, T. (2007). Large scale statistical inference of signaling pathways from RNAi and microarray data. *BMC Bioinformatics*, 8:386.

Gentleman, R. C., Carey, V. J., Bates, D. M., Bolstad, B., Dettling, M., Dudoit, S., Ellis, B., Gautier, L., Ge, Y., Gentry, J., et al. (2004). Bioconductor: open software development for computational biology and bioinformatics. *Genome Biology*, 5:1.

Ghanbari, M., Lasserre, J., and Vingron, M. (2015). Reconstruction of gene networks using prior knowledge. *BMC Systems Biology*, 9(1):1.

Goldfarb, D. (1970). A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24:23–26.

Gomez-Alvarez, V., Teal, T. K., and Schmidt, T. M. (2009). Systematic artifacts in metagenomes from complex microbial communities. *The ISME Journal*, 3:1314–1317.

Grechkin, M., Logsdon, B. A., Gentles, A. J., and Lee, S.-I. (2016). Identifying network perturbation in cancer. *PLoS Computational Biology*, 12:e1004888.

Guelzim, N., Bottani, S., Bourgine, P., and Képès, F. (2002). Topological and causal structure of the yeast transcriptional regulatory network. *Nature Genetics*, 31:60–63.

Gustafsson, M., Hörnquist, M., Lundström, J., Björkegren, J., and Tegnér, J. (2009). Reverse engineering of gene networks with LASSO and nonlinear basis functions. *Annals of the New York Academy of Sciences*, 1158:265–275.

Hecker, M., Lambeck, S., Toepfer, S., Van Someren, E., and Guthke, R. (2009). Gene regulatory network inference: data integration in dynamic models-a review. *Biosystems*, 96:86–103.

Hoeting, J. A., Madigan, D., Raftery, A. E., and Volinsky, C. T. (1999). Bayesian model averaging: a tutorial. *Statistical science*, pages 382–401.

Jeong, H., Mason, S. P., Barabási, A.-L., and Oltvai, Z. N. (2001). Lethality and centrality in protein networks. *Nature*, 411:41–42.

Jiang, D., Tang, C., and Zhang, A. (2004). Cluster analysis for gene expression data: A survey. *IEE Transactions on Knowledge and Data Engineering*, 16:1370–1386.

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

Kazor, K. and Hering, A. S. (2015). Assessing the performance of model-based clustering methods in multivariate time series with application to identifying regional wind regimes. *Journal of Agricultural, Biological, and Environmental Statistics*, 20:192–217.

Kim, K.-H., Yun, S.-T., Park, S.-S., Joo, Y., and Kim, T.-S. (2014). Model-based clustering of hydrochemical data to demarcate natural versus human impacts on bedrock groundwater quality in rural areas, South Korea. *Journal of Hydrology*, 519:626–636.

Kim, S., Imoto, S., and Miyano, S. (2004). Dynamic Bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data. *Biosystems*, 75:57–65.

Kim, S. Y., Imoto, S., Miyano, S., and others (2003). Inferring gene networks from time series microarray data using dynamic Bayesian networks. *Briefings in Bioinformatics*, 4:228.

Klamt, S., Flassig, R. J., and Sundmacher, K. (2010). TRANSWESD: inferring cellular networks with transitive reduction. *Bioinformatics*, 26:2160–2168.

Lèbre, S., Becq, J., Devaux, F., Stumpf, M. P., and Lelandais, G. (2010). Statistical inference of the time-varying structure of gene-regulation networks. *BMC Systems Biology*, 4:article 130.

Lee, W.-P. and Tzou, W.-S. (2009). Computational methods for discovering gene networks from expression data. *Briefings in Bioinformatics*, 10:408–423.

Leek, J. T., Johnson, W. E., Parker, H. S., Jaffe, A. E., and Storey, J. D. (2012). The sva package for removing batch effects and other unwanted variation in high-throughput experiments. *Bioinformatics*, 28:882–883.

Leek, J. T., Scharpf, R. B., Bravo, H. C., Simcha, D., Langmead, B., Johnson, W. E., Geman, D., Baggerly, K., and Irizarry, R. A. (2010). Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews Genetics*, 11:733–739.

Lehmann, R., Machné, R., Georg, J., Benary, M., Axmann, I. M., and Steuer, R. (2013). How cyanobacteria pose new problems to old methods: challenges in microarray time series analysis. *BMC Bioinformatics*, 14:article 133.

Li, J. and Tibshirani, R. (2013). Finding consistent patterns: a nonparametric approach for identifying differential expression in RNA-seq data. *Statistical Methods in Medical Research*, 22:519–536.

Li, Q., Fraley, C., Bumgarner, R. E., Yeung, K. Y., and Raftery, A. E. (2005). Donuts, scratches and blanks: robust model-based segmentation of microarray images. *Bioinformatics*, 21:2875–2882.

Li, W. V. and Wei, A. (2012). A Gaussian inequality for expected absolute products. *Journal of Theoretical Probability*, 25:92–99.

Li, Z., Li, P., Krishnan, A., and Liu, J. (2011). Large-scale dynamic gene regulatory network inference combining differential equation models with local dynamic Bayesian network analysis. *Bioinformatics*, 27:2686–2691.

Liu, C., Su, J., Yang, F., Wei, K., Ma, J., and Zhou, X. (2015). Compound signature detection on LINCS L1000 big data. *Molecular BioSystems*, 11:714–722.

Liu, X. and Rattray, M. (2010). Including probe-level measurement error in robust mixture clustering of replicated microarray gene expression. *Statistical Applications in Genetics and Molecular Biology*, 9:article 42.

Lo, K., Raftery, A. E., Dombek, K. M., Zhu, J., Schadt, E. E., Bumgarner, R. E., and Yeung, K. Y. (2012). Integrating external biological knowledge in the construction of regulatory networks from time-series expression data. *BMC Systems Biology*, 6:101.

Lockhart, D. J., Dong, H., Byrne, M. C., Follettie, M. T., Gallo, M. V., Chee, M. S., Mittmann, M., Wang, C., Kobayashi, M., Horton, H., et al. (1996). Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature biotechnology*, 14:1675–1680.

Lopes, F. M., de Oliveira, E. A., and Cesar, R. M. (2011). Inference of gene regulatory networks from time series by Tsallis entropy. *BMC Systems Biology*, 5:61.

Lütkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer Science & Business Media.

Madigan, D. and Raftery, A. E. (1994). Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of the American Statistical Association*, 89:1535–1546.

Marbach, D., Prill, R. J., Schaffter, T., Mattiussi, C., Floreano, D., and Stolovitzky, G. (2010). Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107:6286–6291.

Marbach, D., Schaffter, T., Mattiussi, C., and Floreano, D. (2009). Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of Computational Biology*, 16:229–239.

Margolin, A. A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Favera, R. D., and Califano, A. (2006). ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7:1–10.

Markowetz, F. and Spang, R. (2007). Inferring cellular networks-a review. *BMC Bioinformatics*, 8:article S5.

McGeachie, M. J., Chang, H.-H., and Weiss, S. T. (2014). CGBayesNets: conditional Gaussian Bayesian Network learning and inference with mixed discrete and continuous data. *PLoS Computational Biology*, 10:article e1003676.

McLachlan, G. and Krishnan, T. (2007). *The EM algorithm and extensions*, volume 382. John Wiley & Sons.

McLachlan, G. J. and Peel, D. (2000). *Finite Mixture Models*. Wiley.

Medvedovic, M. and Sivaganesan, S. (2002). Bayesian infinite mixture model based clustering of gene expression profiles. *Bioinformatics*, 18:1194–1206.

Men'endez, P., Kourmpetis, Y. A., ter Braak, C. J., and van Eeuwijk, F. A. (2010). Gene regulatory networks from multifactorial perturbations using graphical lasso: application to the DREAM4 challenge. *PloS One*, 5:e14147.

Meyer, P. E., Kontos, K., Lafitte, F., and Bontempi, G. (2007). Information-theoretic inference of large transcriptional regulatory networks. *EURASIP Journal on Bioinformatics and Systems Biology*, 2007:1–9.

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

Michailidis, G. and d'Alché Buc, F. (2013). Autoregressive models for gene regulatory network inference: Sparsity, stability and causality issues. *Mathematical Biosciences*, 246:326–334.

Morrissey, E. R., Juárez, M. A., Denby, K. J., and Burroughs, N. J. (2010). On reverse engineering of gene interaction networks using time course data with repeated measurements. *Bioinformatics*, 26:2305–2312.

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

Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. *The Computer Journal*, 7:308–313.

Omranian, N., Eloundou-Mbebi, J. M., Mueller-Roeber, B., and Nikoloski, Z. (2016). Gene regulatory network inference using fused LASSO on multiple data sets. *Scientific Reports*, 6.

Pearl, J. (2009). *Causality*. Cambridge university press.

Peck, D., Crawford, E. D., Ross, K. N., Stegmaier, K., Golub, T. R., and Lamb, J. (2006). A method for high-throughput gene expression signature analysis. *Genome Biology*, 7:article R61.

Pe'er, D., Regev, A., Elidan, G., and Friedman, N. (2001). Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 17:S215–S224.

Pinna, A., Soranzo, N., and De La Fuente, A. (2010). From knockouts to networks: establishing direct cause-effect relationships through graph analysis. *PloS One*, 5:e12912.

Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. A., and Reich, D. (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nature Genetics*, 38:904–909.

Prill, R. J., Marbach, D., Saez-Rodriguez, J., Sorger, P. K., Alexopoulos, L. G., Xue, X., Clarke, N. D., Altan-Bonnet, G., and Stolovitzky, G. (2010). Towards a rigorous assessment of systems biology models: the DREAM3 challenges. *PloS One*, 5:article e9202.

Raftery, A. E. (1995). Bayesian model selection in social research. *Sociological Methodology*, 25:111–164.

Raftery, A. E. (1999). Bayes factors and BIC. *Sociological Methods & Research*, 27:411–417.

Raftery, A. E., Madigan, D., and Hoeting, J. A. (1997). Bayesian model averaging for linear regression models. *Journal of the American Statistical Association*, 92:179–191.

Raghavan, V., Bollman, P., and Jung, G. S. (1989). A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions on Information Systems*, 7:205–220.

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

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

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

Salleh, F. H. M., Arif, S. M., Zainudin, S., and Raih, M. F. (2015). Reconstructing gene regulatory networks from knock-out data using Gaussian noise model and Pearson correlation coefficient. *Computational Biology and Chemistry*.

Sanchez-Castillo, M., Tienda-Luna, I., Blanco, D., Carrion-Perez, M. C., and Huang, Y.-P. (2013). Bayesian sparse factor model for transcriptional regulatory networks inference. In *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European*, pages 1–4. IEEE.

Sandelin, A., Alkema, W., Engstrm, P., Wasserman, W. W., and Lenhard, B. (2004). JASPAR: an open-access database for eukaryotic transcription factor binding profiles. *Nucleic Acids Research*, 32:D91–D94.

Sch afer, J., Opgen-Rhein, R., and Strimmer, K. (2001). Reverse engineering genetic networks using the GeneNet package. *J Am Stat Assoc*, 96:1151–1160.

Schaffter, T., Marbach, D., and Floreano, D. (2011). GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, 27:2263–2270.

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

Scutari, M. (2010). Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35:1–22.

Sebastiani, P., Gussoni, E., Kohane, I. S., and Ramoni, M. F. (2003). Statistical challenges in functional genomics. *Statistical Science*, 18:33–60.

Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D., and Friedman, N. (2003). Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34:166–176.

Shao, H., Peng, T., Ji, Z., Su, J., and Zhou, X. (2013). Systematically studying kinase inhibitor induced signaling network signatures by integrating both therapeutic and side effects. *PLoS One*, 8:article e80832.

Sheppard, K. (2013). Financial econometrics notes.

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

Shojaie, A., Jauhiainen, A., Kallitsis, M., and Michailidis, G. (2014). Inferring regulatory networks by combining perturbation screens and steady state gene expression profiles. *PloS One*, 9:article e82393.

Shojaie, A. and Michailidis, G. (2009). Analysis of gene sets based on the underlying regulatory network. *Journal of Computational Biology*, 16:407–426.

Shojaie, A. and Michailidis, G. (2010). Discovering graphical Granger causality using the truncating lasso penalty. *Bioinformatics*, 26:i517–i523.

Siegmund, K. D., Laird, P. W., and Laird-Offringa, I. A. (2004). A comparison of cluster analysis methods using DNA methylation data. *Bioinformatics*, 20:1896–1904.

Singh, N. and Vidyasagar, M. (2016). bLARS: An algorithm to infer gene regulatory networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 13:301–314.

Smith, S. M., Fulton, D. C., Chia, T., Thorneycroft, D., Chapple, A., Dunstan, H., Hylton, C., Zeeman, S. C., and Smith, A. M. (2004). Diurnal changes in the transcriptome

encoding enzymes of starch metabolism provide evidence for both transcriptional and post-transcriptional regulation of starch metabolism in Arabidopsis leaves. *Plant Physiology*, 136:2687–2699.

Stokes, T. H., Moffitt, R. A., Phan, J. H., and Wang, M. D. (2007). Chip artifact CORRECTion (caCORRECT): a bioinformatics system for quality assurance of genomics and proteomics array data. *Annals of Biomedical Engineering*, 35:1068–1080.

Sun, Z., Wu, Y., White, W. M., Donkena, K. V., Klein, C. J., Garovic, V. D., Therneau, T. M., and Kocher, J.-P. A. (2011). Batch effect correction for genome-wide methylation data with Illumina Infinium platform. *BMC Medical Genomics*, 4:article 84.

Teixeira, M. C., Monteiro, P., Jain, P., Tenreiro, S., Fernandes, A. R., Mira, N. P., Alenquer, M., Freitas, A. T., Oliveira, A. L., and S-Correia, I. (2006). The YEASTRACT database: a tool for the analysis of transcription regulatory associations in Saccharomyces cerevisiae. *Nucleic Acids Research*, 34:D446–D451.

Templ, M., Filzmoser, P., and Reimann, C. (2008). Cluster analysis applied to regional geochemical data: problems and possibilities. *Applied Geochemistry*, 23:2198–2213.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67:91–108.

Tong, L., Yu, S., and Liu, R. (1992). Identifiability of a set of quadratic equations with unknown coefficients. In *Circuits and Systems, 1992. ISCAS'92. Proceedings., 1992 IEEE International Symposium on*, volume 1, pages 292–295.

Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences*, 98:5116–5121.

Vempati, U. D., Chung, C., Mader, C., Koleti, A., Datar, N., Vidović, D., Wrobel, D., Erickson, S., Muhlich, J. L., Berriz, G., et al. (2014). Metadata standard and data exchange specifications to describe, model, and integrate complex and diverse high-throughput screening data from the Library of Integrated Network-based Cellular Signatures (LINCS). *Journal of Biomolecular Screening*, 19:803–816.

Verbist, B., Clement, L., Reumers, J., Thys, K., Vapirev, A., Talloen, W., Wetzels, Y., Meys, J., Aerssens, J., Bijnens, L., et al. (2015). ViVaMBC: estimating viral sequence variation in complex populations from illumina deep-sequencing data using model-based clustering. *BMC Bioinformatics*, 16:article 59.

Verzelen, N. (2012). Minimax risks for sparse regressions: Ultra-high dimensional phenomenons. *Electronic Journal of Statistics*, 6:38–90.

Vidović, D., Koleti, A., and Schürer, S. C. (2013). Large-scale integration of small molecule-induced genome-wide transcriptional responses, kinome-wide binding affinities and cell-growth inhibition profiles reveal global trends characterizing systems-level drug action. *Frontiers in Genetics*, 5:342–342.

Wainwright, M. J. (2009). Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (Lasso). *IEEE Transactions on Information Theory*, 55:2183–2202.

Wang, Z., Gerstein, M., and Snyder, M. (2009). RNA-seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10:57–63.

Wingender, E., Chen, X., Hehl, R., Karas, H., Liebich, I., Matys, V., Meinhardt, T., Prüß,

M., Reuter, I., and Schacherer, F. (2000). TRANSFAC: an integrated system for gene expression regulation. *Nucleic Acids Research*, 28:316–319.

Wolfe, J. H. (1970). Pattern clustering by multivariate mixture analysis. *Multivariate Behavioral Research*, 5:329–350.

Wu, C. F. J. (1983). On convergence properties of the EM algorithm. *Annals of Statistics*, 11:95–103.

Yeung, K. Y., Bumgarner, R. E., and Raftery, A. E. (2005). Bayesian model averaging: development of an improved multi-class, gene selection and classification tool for microarray data. *Bioinformatics*, 21:2394–2402.

Yeung, K. Y., Dombek, K. M., Lo, K., Mittler, J. E., Zhu, J., Schadt, E. E., Bumgarner, R. E., and Raftery, A. E. (2011). Construction of regulatory networks using expression time-series data of a genotyped population. *Proceedings of the National Academy of Sciences*, 108:19436–19441.

Yeung, K. Y., Fraley, C., Murua, A., Raftery, A. E., and Ruzzo, W. L. (2001). Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17:977–987.

Yip, K. Y., Alexander, R. P., Yan, K.-K., and Gerstein, M. (2010). Improved reconstruction of in silico gene regulatory networks by integrating knockout and perturbation data. *PloS One*, 5:article e8121.

Yoo, C., Thorsson, V., and Cooper, G. F. (2002). Discovery of causal relationships in a gene-regulation pathway from a mixture of experimental and observational DNA microarray data. In *Pacific Symposium on Biocomputing*, volume 7, pages 498–509.

Young, W. C., Raftery, A. E., and Yeung, K. Y. (2014). Fast Bayesian inference for gene regulatory networks using ScanBMA. *BMC Systems Biology*, 8:47.

Young, W. C., Yeung, K. Y., and Raftery, A. E. (2016a). Model-based clustering with data correction for removing artifacts in gene expression data. *arXiv preprint arXiv:1602.06316*.

Young, W. C., Yeung, K. Y., and Raftery, A. E. (2016b). A posterior probability approach for gene regulatory network inference in genetic perturbation data. *arXiv preprint arXiv:1603.04835*.

Zellner, A. (1986). On assessing prior distributions and Bayesian regression analysis with g-prior distributions. *Bayesian Inference and Decision Techniques: Essays in Honor of Bruno De Finetti*, 6:233–243.

Zhu, J., Chen, Y., Leonardson, A. S., Wang, K., Lamb, J. R., Emilsson, V., and Schadt, E. E. (2010). Characterizing dynamic changes in the human blood transcriptional network. *PloS Computational Biology*, 6:article e1000671.

Zhu, J., Zhang, B., Smith, E. N., Drees, B., Brem, R. B., Kruglyak, L., Bumgarner, R. E., and Schadt, E. E. (2008). Integrating large-scale functional genomic data to dissect the complexity of yeast regulatory networks. *Nature Genetics*, 40:854–861.

Zoppoli, P., Morganella, S., and Ceccarelli, M. (2010). TimeDelay-ARACNE: Reverse engineering of gene networks from time-course data by an information theoretic approach. *BMC Bioinformatics*, 11:154.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67:301–320.

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

Appendix A

VAR DERIVATIVES

A.1 Model

The VAR equilibrium model in section 5.3.1 can be written as

$$\begin{aligned} \mathbf{x} &\sim N(0, \mathbf{\Sigma}), \\ \mathbf{\Sigma} &= \sum_{i=0}^{\infty} \mathbf{A}^i \mathbf{D} (\mathbf{A}^T)^i, \end{aligned}$$

where $\mathbf{D}$ is known and $\mathbf{A}$, the parameter of interest, is lower-triangular. Additionally, the sign of the diagonal entries of $\mathbf{A}$ is known and both the eigenvalues and diagonal entries are less than 1 in absolute value. We can write out the likelihood as follows:

$$\begin{aligned} f(\mathbf{x}|\mathbf{\Sigma}) &= (2\pi)^{-p/2} |\mathbf{\Sigma}|^{-1/2} \exp\left[-\frac{1}{2} \mathbf{x}^T \mathbf{\Sigma}^{-1} \mathbf{x} \right], \\ f(\mathbf{x}|\mathbf{A}) &= (2\pi)^{-p/2} |\mathbf{\Sigma}(\mathbf{A})|^{-1/2} \exp\left[-\frac{1}{2} \mathbf{x}^T \mathbf{\Sigma}(\mathbf{A})^{-1} \mathbf{x} \right], \\ \mathbf{\Sigma}(\mathbf{A}) &= \sum_{i=0}^{\infty} \mathbf{A}^i \mathbf{D} (\mathbf{A}^T)^i. \end{aligned}$$

Here p is the dimension of $\mathbf{x}$.

A.2 First Derivatives

To find the first derivatives, I actually want to work with the log-density and take partial derivatives of that:

$$\begin{aligned} \log f(\mathbf{x}|\mathbf{A}) &= -\frac{p}{2} \log(2\pi) - \frac{1}{2} \log |\mathbf{\Sigma}(\mathbf{A})| - \frac{1}{2} \mathbf{x}^T \mathbf{\Sigma}(\mathbf{A})^{-1} \mathbf{x}, \\ \frac{\partial}{\partial a_{ij}} \log f(\mathbf{x}|\mathbf{A}) &= \sum_{k,l} \frac{\partial}{\partial \sigma_{kl}} \log f(\mathbf{x}|\mathbf{\Sigma}) \frac{\partial \sigma_{kl}}{\partial a_{ij}}, \\ &= -\frac{1}{2} \sum_{k,l} \left[\mathbf{\Sigma}^{-1} - \mathbf{\Sigma}^{-1} \mathbf{x} \mathbf{x}^T \mathbf{\Sigma}^{-1} \right]_{kl} \frac{\partial \sigma_{kl}}{\partial a_{ij}}. \end{aligned}$$

Now we need to find the derivatives of $\mathbf{\Sigma}$ with respect to the elements of $\mathbf{A}$. Break it into the summands and consider each separately. Let $\mathbf{A} = \{a_{ij}\}$ be as previously defined (lower-diagonal) and $\mathbf{B} = \{b_{ij}\}$ be a $p \times p$ symmetric matrix. Note that $j > i \Rightarrow a_{ij} = 0$.

$$\begin{aligned}
(\mathbf{AB})_{ij} &= \sum_{k=1}^{p} a_{ik} b_{kj}, \\
&= \sum_{k=1}^{i} a_{ik} b_{kj}, \\
(\mathbf{ABA}^T)_{ij} &= \sum_{k=1}^{p} (\mathbf{AB})_{ik} a_{jk}, \\
&= \sum_{k=1}^{j} (\sum_{l=1}^{i} a_{il} b_{lk}) a_{jk}.
\end{aligned}$$

Now, take the derivative

$$\begin{aligned}
\frac{\partial}{\partial a_{mn}} (\mathbf{AB})_{ik} &= \sum_{l=1}^{i} \left[b_{lk} \mathbf{1}_{[i=m, l=n]} + a_{il} \frac{\partial b_{lk}}{\partial a_{mn}} \right], \\
&= b_{nk} \mathbf{1}_{[i=m, i \geq n]} + \sum_{l=1}^{i} a_{il} \frac{\partial b_{lk}}{\partial a_{mn}}, \\
\frac{\partial}{\partial a_{mn}} (\mathbf{ABA}^T)_{ij} &= \sum_{k=1}^{j} \left[(\mathbf{AB})_{ik} \mathbf{1}_{[j=m, k=n]} + a_{jk} \frac{\partial}{\partial a_{mn}} (\mathbf{AB})_{ik} \right], \\
&= (\mathbf{AB})_{in} \mathbf{1}_{[j=m, j \geq n]} + \sum_{k=1}^{j} a_{jk} \frac{\partial}{\partial a_{mn}} (\mathbf{AB})_{ik}, \\
&= (\mathbf{AB})_{in} \mathbf{1}_{[j=m, j \geq n]} + \sum_{k=1}^{j} \left[a_{jk} b_{kn} \mathbf{1}_{[i=m, i \geq n]} + \sum_{l=1}^{i} a_{jk} a_{il} \frac{\partial b_{lk}}{\partial a_{mn}} \right], \\
&= (\mathbf{AB})_{in} \mathbf{1}_{[j=m, j \geq n]} + (\mathbf{AB})_{jn} \mathbf{1}_{[i=m, i \geq n]} + \sum_{k=1}^{j} \sum_{l=1}^{i} a_{jk} a_{il} \frac{\partial b_{lk}}{\partial a_{mn}}.
\end{aligned}$$

Now we want to apply this to one of our summands

$$\begin{aligned}\frac{\partial}{\partial a_{mn}}(\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^s)_{ij} &= \frac{\partial}{\partial a_{mn}}(\mathbf{A}[\mathbf{A}^{s-1}\mathbf{D}(\mathbf{A}^T)^{s-1}]\mathbf{A}^T)_{ij},\\ &= (\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^{s-1})_{in}\mathbf{1}_{[j=m,j\geq n]} + (\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^{s-1})_{jn}\mathbf{1}_{[i=m,i\geq n]}\\ &\quad + \sum_{k=1}^{j}\sum_{l=1}^{i} a_{jk}a_{il}\frac{\partial}{\partial a_{mn}}(\mathbf{A}^{s-1}\mathbf{D}(\mathbf{A}^T)^{s-1})_{lk}.\end{aligned}$$

Now we're ready to put it all together.

$$\begin{aligned}\frac{\partial\sigma_{ij}}{\partial a_{mn}} &= \frac{\partial}{\partial a_{mn}}\left(\sum_{s=0}^{\infty}(\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^s)_{ij}\right),\\ &= \sum_{s=1}^{\infty}\frac{\partial}{\partial a_{mn}}(\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^s)_{ij},\\ &= \sum_{s=1}^{\infty}\left[(\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^{s-1})_{in}\mathbf{1}_{[j=m,j\geq n]} + (\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^{s-1})_{jn}\mathbf{1}_{[i=m,i\geq n]}\right]\\ &\quad + \sum_{s=1}^{\infty}\sum_{k=1}^{j}\sum_{l=1}^{i} a_{jk}a_{il}\frac{\partial}{\partial a_{mn}}(\mathbf{A}^{s-1}\mathbf{D}(\mathbf{A}^T)^{s-1})_{lk},\\ &= (\mathbf{A}\sum_{s=0}^{\infty}\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^s)_{in}\mathbf{1}_{[j=m,j\geq n]} + (\mathbf{A}\sum_{s=0}^{\infty}\mathbf{A}^s\mathbf{D}(\mathbf{A}^T)^s)_{jn}\mathbf{1}_{[i=m,i\geq n]} + B_{ijmn},\\ &= (\mathbf{A}\boldsymbol{\Sigma})_{in}\mathbf{1}_{[j=m,j\geq n]} + (\mathbf{A}\boldsymbol{\Sigma})_{jn}\mathbf{1}_{[i=m,i\geq n]} + B_{ijmn}.\end{aligned}$$

Now looking at just the third term

$$
\begin{aligned}
B_{ijmn} &= \sum_{s=1}^{\infty}\sum_{k=1}^{j}\sum_{l=1}^{i} a_{jk}a_{il}\frac{\partial}{\partial a_{mn}}(\mathbf{A}^{s-1}\mathbf{D}(\mathbf{A}^T)^{s-1})_{lk}, \\
&= \sum_{k=1}^{j}\sum_{l=1}^{i} a_{jk}a_{il}\frac{\partial}{\partial a_{mn}}\left(\sum_{s=1}^{\infty}\mathbf{A}^{s-1}\mathbf{D}(\mathbf{A}^T)^{s-1}\right)_{lk}, \\
&= \sum_{k=1}^{j}\sum_{l=1}^{i} a_{jk}a_{il}\frac{\partial}{\partial a_{mn}}\left(\sum_{s=0}^{\infty}\mathbf{A}^{s}\mathbf{D}(\mathbf{A}^T)^{s}\right)_{lk}, \\
&= \sum_{k=1}^{j}\sum_{l=1}^{i} a_{jk}a_{il}\frac{\partial\sigma_{lk}}{\partial a_{mn}}, \\
&= a_{ii}a_{jj}\frac{\partial\sigma_{ij}}{\partial a_{mn}} + \sum_{k=1}^{j}\sum_{l=1}^{i} a_{jk}a_{il}\frac{\partial\sigma_{lk}}{\partial a_{mn}}\mathbf{1}_{[(k,l)\neq(j,i)]}.
\end{aligned}
$$

The last equation just takes out the (i,j) term. Putting it all together, we get

$$
\frac{\partial\sigma_{ij}}{\partial a_{mn}} = \frac{1}{1-a_{ii}a_{jj}}\left[\begin{array}{c}(\mathbf{A\Sigma})_{in}\mathbf{1}_{[j=m,j\geq n]} + (\mathbf{A\Sigma})_{jn}\mathbf{1}_{[i=m,i\geq n]} \\ +\sum_{k=1}^{i}\sum_{l=1}^{j} a_{ik}a_{jl}\frac{\partial\sigma_{kl}}{\partial a_{mn}}\mathbf{1}_{[(k,l)\neq(i,j)]}\end{array}\right].
$$

This is an iterative solution. That is, the derivative of σ_{ij} is dependent on the derivatives of only the other elements in the submatrix of $\mathbf{\Sigma}$ from the $(1,1)$ element to the (i,j) element. Thus calculation of all the first partials is straightforward. Also, note that our constraint on the diagonal elements of $\mathbf{A}$ means that $a_{ii}a_{jj} < 1$, and thus the first derivatives are finite.

A.3 Second Derivatives

Now for a derivation of the second partials. Start with some notation for simpler representation:

$$\begin{aligned}
\frac{\partial \sigma_{ij}}{\partial a_{mn}} &= \frac{1}{1-a_{ii}a_{jj}} Q_{ijmn}, \\
Q_{ijmn} &= (\mathbf{A\Sigma})_{in}\mathbf{1}_{[j=m,j\geq n]} + (\mathbf{A\Sigma})_{jn}\mathbf{1}_{[i=m,i\geq n]} + \sum_{k=1}^{i}\sum_{l=1}^{j} a_{ik}a_{jl}\frac{\partial \sigma_{kl}}{\partial a_{mn}}\mathbf{1}_{[(k,l)\neq(i,j)]}, \\
\frac{\partial}{\partial \sigma_{uv}}\left(\frac{\partial \sigma_{ij}}{\partial a_{mn}}\right) &= \frac{Q_{ijmn}}{(1-a_{ii}a_{jj})^2}\left(\mathbf{1}_{[i=u,i=v]}a_{jj} + \mathbf{1}_{j=u,j=v]}a_{ii}\right) \\
&+ \frac{1}{1-a_{ii}a_{jj}}\frac{\partial Q_{ijmn}}{\partial a_{uv}}, \\
\frac{\partial Q_{ijmn}}{\partial a_{uv}} &= \mathbf{1}_{[j=m,j\geq n]}\frac{\partial}{\partial a_{uv}}(\mathbf{A\Sigma})_{in} + \mathbf{1}_{[i=m,i\geq n]}\frac{\partial}{\partial a_{uv}}(\mathbf{A\Sigma})_{jn} \\
&+ \mathbf{1}_{[i=u,i\geq v]}\sum_{l=1}^{j} a_{jl}\frac{\partial \sigma_{vl}}{\partial a_{mn}} + \mathbf{1}_{[j=u,j\geq v]}\sum_{k=1}^{i} a_{ik}\frac{\partial \sigma_{kv}}{\partial a_{mn}} \\
&- \mathbf{1}_{[i=u,i=v]}a_{jj}\frac{\partial \sigma_{vj}}{\partial a_{mn}} - \mathbf{1}_{[j=u,j=v]}a_{ii}\frac{\partial \sigma_{iv}}{\partial a_{mn}} \\
&+ \sum_{k=1}^{i}\sum_{l=1}^{j}\mathbf{1}_{[(k,l)\neq(i,j)]}a_{ik}a_{jl}\frac{\partial}{\partial a_{uv}}\left(\frac{\partial \sigma_{kl}}{\partial a_{mn}}\right), \\
\frac{\partial}{\partial a_{uv}}(\mathbf{A\Sigma})_{in} &= \mathbf{1}_{[i=u,i\geq v]}\sigma_{vn} + \sum_{l=1}^{i} a_{il}\frac{\partial \sigma_{ln}}{\partial a_{uv}}.
\end{aligned}$$

Although complex, this has the same iterative structure as the first partials. This means that we can calculate all the second partials. Now, to get the second partial derivatives of the log-likelihood:

$$\begin{aligned}
\frac{\partial}{\partial a_{uv}}\left(\frac{\partial}{\partial a_{mn}}\log f(\mathbf{x}|\mathbf{A})\right) &= \frac{\partial}{\partial a_{uv}}\left(\sum_{i,j}\frac{\partial}{\partial \sigma_{ij}}\log f(\mathbf{x}|\mathbf{\Sigma})\frac{\partial \sigma_{ij}}{\partial a_{mn}}\right), \\
&= -\frac{1}{2}\sum_{i,j}\frac{\partial}{\partial a_{uv}}\left(\left[\mathbf{\Sigma}^{-1} - \mathbf{\Sigma}^{-1}\mathbf{x}\mathbf{x}^T\mathbf{\Sigma}^{-1}\right]_{ij}\frac{\partial \sigma_{ij}}{\partial a_{mn}}\right), \\
\frac{\partial}{\partial a_{uv}}\left(\left[\mathbf{\Sigma}^{-1} - \mathbf{\Sigma}^{-1}\mathbf{x}\mathbf{x}^T\mathbf{\Sigma}^{-1}\right]_{ij}\frac{\partial \sigma_{ij}}{\partial a_{mn}}\right) &= \frac{\partial}{\partial a_{uv}}\left(\left[\mathbf{\Sigma}^{-1} - \mathbf{\Sigma}^{-1}\mathbf{x}\mathbf{x}^T\mathbf{\Sigma}^{-1}\right]_{ij}\right)\frac{\partial \sigma_{ij}}{\partial a_{mn}} \\
&+ \left[\mathbf{\Sigma}^{-1} - \mathbf{\Sigma}^{-1}\mathbf{x}\mathbf{x}^T\mathbf{\Sigma}^{-1}\right]_{ij}\frac{\partial}{\partial a_{uv}}\left(\frac{\partial \sigma_{ij}}{\partial a_{mn}}\right).
\end{aligned}$$

The second term has been shown above. The first term can be shown to be

$$\begin{aligned}
\frac{\partial}{\partial a_{uv}}\left(\left[\boldsymbol{\Sigma}^{-1}-\boldsymbol{\Sigma}^{-1}\mathbf{x}\mathbf{x}^{T}\boldsymbol{\Sigma}^{-1}\right]_{ij}\right) &= \sum_{k,l}\frac{\partial}{\partial \sigma_{kl}}\left(\left[\boldsymbol{\Sigma}^{-1}-\boldsymbol{\Sigma}^{-1}\mathbf{x}\mathbf{x}^{T}\boldsymbol{\Sigma}^{-1}\right]_{ij}\right)\frac{\partial \sigma_{kl}}{\partial a_{uv}}, \\
\frac{\partial}{\partial \sigma_{kl}}\left(\left[\boldsymbol{\Sigma}^{-1}-\boldsymbol{\Sigma}^{-1}\mathbf{x}\mathbf{x}^{T}\boldsymbol{\Sigma}^{-1}\right]_{ij}\right) &= \sum_{m,n}(\boldsymbol{\Sigma}^{-1})_{mk}(\boldsymbol{\Sigma}^{-1})_{ln}\times \\
&\qquad (\boldsymbol{\Sigma}^{-1}\mathbf{x}\mathbf{x}^{T}\mathbf{J}^{mn}+\mathbf{J}^{nm}\mathbf{x}\mathbf{x}^{T}\boldsymbol{\Sigma}^{-1}-\mathbf{J}^{mn})_{ij},
\end{aligned}$$

where $\mathbf{J}^{mn}$ is a matrix of all zeros with a 1 in the (m,n) position. Again, all the second partials are both finite and continuous.

VITA

William Chad Young was born in Portland, Oregon. He earned his Bachelor of Science in Physics from the California Institute of Technology in 2002 and then worked in software development until 2009. He obtained his Masters of Science in Statistics from Portland State University in 2009 and began his Ph.D. in the Department of Statistics, University of Washington, in the same year.