Causal Gene Network Inference from Genetical Genomics Experiments via Structural Equation Modeling

Bing Liu

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Statistics

Ina Hoeschele, Chair
Jeffrey B. Birch
M. A. Saghai Maroof
Pedro Mendes
Keying Ye

September 11th, 2006
Blacksburg, Virginia

Keywords: Gene Network, Genetical Genomics, Structural Equation Modeling, Gene Expression, Microarray

Causal Gene Network Inference from Genetical Genomics Experiments via Structural Equation Modeling

Bing Liu

(ABSTRACT)

The goal of this research is to construct causal gene networks for genetical genomics experiments using expression Quantitative Trait Loci (eQTL) mapping and Structural Equation Modeling (SEM). Unlike Bayesian Networks, this approach is able to construct cyclic networks, while cyclic relationships are expected to be common in gene networks. Reconstruction of gene networks provides important knowledge about the molecular basis of complex human diseases and generally about living systems.

In genetical genomics, a segregating population is expression profiled and DNA marker genotyped. An Encompassing Directed Network (EDN) of causal regulatory relationships among genes can be constructed with eQTL mapping and selection of candidate causal regulators. Several eQTL mapping approaches and local structural models were evaluated in their ability to construct an EDN. The edges in an EDN correspond to either direct or indirect causal relationships, and the EDN is likely to contain cycles or feedback loops. We implemented SEM with genetics algorithms to produce sub-models of the EDN containing fewer edges and being well supported by the data. The EDN construction and sparsification methods were tested on a yeast genetical genomics data set, as well as the simulated data. For the simulated networks, the SEM approach has an average detection power of around ninety percent, and an average false discovery rate of around ten percent.

Acknowledgements

I would like to thank my advisor, Dr. Ina Hoeschele, for her time, patience, guidance, and encouragement during my doctoral study. Without her effort and support I would not have been able to finish this. I am very fortunate to have had the opportunity to work with her. I also would like to thank Dr. Jeffrey Birch, Dr. Saghai Maroof, Dr. Pedro Mendes, and Dr. Keying Ye for serving on my committee, sharing their knowledge, and providing guidance and support. Thank them for taking the time to read my dissertation and for their critical assessment of my research. I appreciate the opportunity of having studied in their classrooms.

A special thank goes to my colleague Dr. Alberto de la Fuente. We have worked closely on this project, and we have some nice discussions almost everyday. I also extend my gratitude to my other colleagues in Dr. Hoeschele's group: Drs. Guiming Gao, Yongcai Mao, Hua Li and Nan Bing. I am very grateful to them for their friendship, valuable technique discussions, and support.

I would also like to thank my collaborators on the microarray expression analysis, Dr. Allen Taylor, Dr. Fu Shang and especially Dr. Karen Duca. It is a great honor to work with them in the past years.

Finally to my parents, Zhenhua Liu and Qinying Chen. Their faithful love, support and concern have helped me through these years far from home. To my husband Xiaobo Zhou, for always being there for me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the advanced technologies including gene microarrays, more and more data are available containing a vast amount of valuable information. The exciting new technologies call for advanced statistical methods for exploring those data. In genetical genomics (JANSEN 2003; JANSEN and NAP 2001), combining gene expression data and marker genotype information can give us some insights into the construction of the gene network, which is a projection of the complex functional network of DNA, RNA, proteins and metabolites onto the gene space (BRAZHNIK *et al.* 2002). Reconstruction of gene networks provides important knowledge about the molecular basis of complex human diseases and generally about living systems. Gene networks can be described by graphical models, either undirected structures from observational data, or causal structures from experimental data, which is our focus. In this work, various methods for the analysis of microarray data are compared, and then a method for the reconstruction of causal gene networks for genetical genomics experiments is presented.

Bayesian Networks are currently a popular tool for gene network inference (e.g. FRIEDMAN *et al.* 2000; HARTEMINK *et al.* 2002; IMOTO *et al.* 2002; PE'ER *et al.* 2001; YOO *et al.* 2002). Bayesian networks use partially directed graphical models to represent conditional independence relationships among variables of interest and can describe complex stochastic processes. They are suitable for learning from noisy data, for example, expression data (FRIEDMAN *et al.* 2000). However, Bayesian Networks are Directed Acyclic Graphical (DAG) models, which cannot represent structures with cyclic relationships. Cyclic relationships are expected to be common in gene networks, which are hence better modeled as Directed Cyclic Graphs (DCGs). Based on the assumption that a cyclic graph represents a dynamic system at equilibrium (FISHER 1970), this problem can be theoretically resolved by including a time dimension, which produces causal graphs without cycles (DAG). Then DAGs could be studied using Bayesian Networks. Such approach is called Dynamic Bayesian Networks (HARTEMINK *et al.* 2002; MURPHY and MIAN 1999). However, such approach requires the collection of time series data, which is difficult to accomplish, as it requires synchronization of cells and close time intervals not allowing for feedback (SPIRTES *et al.* 2000). Samples at wider time intervals represent near steady state data and hence require cyclic network reconstruction.

Here, we construct causal gene networks in the context of genetical genomics experiments. The concept of inferring gene networks from combining genomic marker information and gene expression data was proposed by JANSEN (2003) and JANSEN and NAP (2001). In genetical genomics, a segregating population of hundreds of individuals is expression profiled and genotyped. An Encompassing Directed Network (EDN) of causal regulatory relationships among genes can be constructed with expression Quantitative Trait Locus (eQTL) mapping and selection of regulator-target pairs. The variation in the expression levels of genes is determined by the variation in many polymorphisms (genotypes) across the genome. The genotypes can thus be regarded as natural multifactorial perturbations (JANSEN 2003; JANSEN and NAP 2001) resulting in different gene expression "phenotypes", and a relationship can be established between the measured genotypes and the measured gene expression phenotypes.

In contrast to the approaches using specific experimental perturbations, in genetical genomics we do not know where the perturbations occur and we must identify their origin. This can be achieved by eQTL mapping, which treats the gene expression profiles in a segregating population as quantitative traits and performs Quantitative Trait Locus (QTL) mapping on those traits. QTL mapping identifies the polymorphic genome regions having significant effects on a quantitative trait. The result of eQTL analysis is the knowledge that certain genomic regions have causal effects on the expression levels of particular genes. Then, using DNA sequence information, genes located in an eQTL region can be identified as candidate causal regulators of the genes whose expression levels are affected by that eQTL. After the identification of the candidate regulatory genes in each eQTL, an EDN of causal regulatory relationships among genes can be constructed. The constructed EDN consists of gene nodes and eQTL nodes. The directed edges in the EDN correspond to causal relationships or regulations among pairs of genes. A set of sparser networks well supported by the data can be found by searching within the space defined by the EDN.

Model search within the space defined by the EDN is based on likelihood estimation with Structural Equation Modeling (SEM). SEM is used because nonrecusive SEM can model cyclic relationships. XIONG *et al.* (2004) were the first to apply SEM for gene network reconstruction using gene expression data. However, their application was limited to gene networks without cyclic relationships by using a recursive SEM, which has an acyclic structure and uncorrelated errors. These authors reconstructed only small networks with less than 20 genes. Based on a factorization of the likelihood and a strongly constrained network topology search space, our implementation is capable of reconstructing network of several hundred genes.

Expression data analysis is very important for gene network inference. Microarray data is very noisy, even with the advanced technologies. How to extract valuable information from the detected array signal is an important research area. In the second chapter of the dissertation, methods for expression data analysis on the most popular microarray platform -- the Affymetrix whole genome short oligonucleotide array platform are discussed. A probe level three-step Linear Mixed Model Analysis approach that uses probe level data directly is compared to other popular methods that summarize probe level data to gene level. In the third chapter, EDN construction using genetic and causal analyses of expression profiles in genetical genomics experiments is discussed, with application to a genetical genomics dataset from yeast (BREM and KRUGLYAK 2005). In the fourth chapter of the dissertation, an EDN sparsification algorithm with SEM is discussed. The implemented algorithm was tested on simulated data set and a sub network of the EDN obtained from chapter 3. Finally in the fifth chapter, future research on gene network inference to genetical genomics experiments is sketched.

REFERENCES

BRAZHNIK, P., A. DE LA FUENTE and P. MENDES, 2002 Gene networks: how to put the function in genomics. Trends Biotechnol. **20:** 467-472.

BREM, R. B., and L. KRUGLYAK, 2005 The landscape of genetic complexity across 5,700 gene expression traits in yeast. Proc. Natl. Acad. Sci. USA **102:** 1572-1577.

FISHER, F. M., 1970 A correspondence principle for simultanious equation models. Econometrica **38:** 73-92.

FRIEDMAN, N., M. LINIAL, I. NACHMAN and D. PE'ER, 2000 Using Bayesian networks to analyze expression data. J. Comp. Biol. **7:** 601-620.

HARTEMINK, A., D. GIFFORD, T. JAAKKOLA and R. YOUNG, 2002 Combining location and expression data for principled discovery of genetic regulatory network models, pp. 437-449 in *Pac. Symp. Biocomput.*

IMOTO, S., K. SUNYONG, T. GOTO, S. ABURATANI, K. TASHIRO *et al.*, 2002 Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network, pp. 219-227 in *Proc. IEEE Comput. Soc. Bioinform. Conf.*

JANSEN, R. C., 2003 Studying complex biological systems using multifactorial perturbation. Nat. Revi. Gen. **4:** 145-151.

JANSEN, R. C., and J. P. NAP, 2001 Genetical genomics: the added value from segregation. Trends Genet. **17:** 388-391.

MURPHY, K., and S. MIAN, 1999 *Modelling gene expression data using dynamic Bayesian networks*. Technical report, Computer Science Division, University of California, Berkeley, CA.

PE'ER, D., A. REGEV, G. ELIDAN and N. FRIEDMAN, 2001 Inferring subnetworks from perturbed expression profiles. Bioinformatics **17:** 215-224.

SPIRTES, P., C. GLYMOUR, R. SCHEINES, S. KAUFFMAN, V. AIMALE *et al.*, 2000 Constructing Bayesian Network Models of Gene Expression Networks from Microarray Data, in *Proc. Atlantic Symp. Comp. Biol., and Genome Inf. Syst. and Technol.*

XIONG, M., J. LI and X. FANG, 2004 Identification of genetic networks. Genetics **166:** 1037-1052.

Yoo, C., V. Thorsson and G. Cooper, 2002 Discovery of causal relationships in a gene-regulation pathway from a mixture of experimental and observational DNA microarray data, pp. 498-509 in *Pac. Symp. Biocomput.*

Chapter 2

A comparison of microarray analysis methods applied to caloric restriction in the Emory mouse

Bing Liu[¶§], Fu Shang*, Allen Taylor*, Karen Duca[§], Ina Hoeschele[§¶]

[¶] Department of Statistics, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061

[§] Virginia Bioinformatics Institute (0477), Virginia Polytechnic Institute and State University, Blacksburg, VA 24061

* Laboratory for Nutrition and Vision Research, Jean Mayer USDA HNRC on Aging at Tufts University, 711 Washington Street, Boston, MA 02111, USA

ABSTRACT

There is no consensus regarding the best statistical methods to use for the evaluation of microarray expression data. Comparing the methods with data from realistic biological experiments provides new insights on this issue. In this study, a probe level three-step Linear Mixed Model Analysis approach, which uses the probe level data directly, was compared to other popular methods to identify significant changes in gene expression that might be associated with lifespan extension in Emory mice. Comparisons of the methods were based on a number of criteria, e.g. the number of genes detected and the overlap between methods. All methods identified the genes with large expression differences between the calorie restricted group and the control group as significant, but the methods yielded different results when the expression differences were small. Out of 12,488 genes on the array, a total of 97 genes were detected by all methods using the Bonferroni multiple testing adjustment criteria. By using a less stringent multiple testing criteria which controls false discovery rate, 855 differentially expressed genes were detected by all methods. Highly differentially expressed genes were relatively robust against the statistical method used. For other genes, no method performed consistently well for all genes since each method rests on different assumptions and extracts information via different mechanisms. The Bonferroni multiple testing adjustment method is clearly not suitable for large scale microarray experiments, due to its lack of power. Robust Multichip Average with GC-content background correction (GCRMA) and Affymetrix MicroArray Suite 5.0 (MAS) detected a smaller number of significant genes than the others. The results suggested that in this dataset GCRMA sacrificed power when guarding against outliers by using median polish gene summary method. Mismatch subtraction used by MAS may add variability to the data.

2.1 INTRODUCTION

Gene expression profiling technology has improved dramatically over the past decade. Various researchers have proposed many statistical methods for extracting useful knowledge from the dense stream of high-throughput data produced using these technologies. However, there is no consensus regarding the best statistical method(s) for analysis of the raw data. Comparing various methods with data from realistic biological experiments can provide new insights for this issue, and help identify the most solid results as well.

Different statistical methods are needed for different microarray platforms. Among many available microarray platforms, currently the most popular platform is the whole genome short oligonucleotide array platform from Affymetrix. The Affymetrix Genechips have around 10,000 to 60,000 gene specific probe sets represented on each chip. Each gene is represented by a probe set composed of several to about 20 pairs of 25-mer oligonucleotides. In this work, we compared a probe level three-step Linear Mixed Model Analysis (PLMMA) approach which uses probe level data directly, to four other popular methods which summarize probe level data to gene level: Robust Multichip Average with Guanosine (G) and Cytosine (C) content background correction (GCRMA) (IRIZARRY *et al.* 2003; WU and IRIZARRY 2005; WU *et al.* 2004), dChip (LI and WONG 2001), Affymetrix MicroArray Suite (MAS) 5.0 (AFFYMETRIX 2002), and MAS 5.0 with perfect match data only (MASPM). The data analysis was performed on microarray data obtained using livers from calorie restricted and control-fed Emory mice.

GCRMA estimates background noise based on a model using GC content (WU and IRIZARRY 2005; WU *et al.* 2004). Background correction in microarray analysis corrects for background noise and adjusts for cross hybridization. After background subtraction, the data is normalized using quantile normalization (BOLSTAD *et al.* 2003), to adjust for effects that are due to the technology rather than variations in the biology of interest, so that in the analysis the arrays are on the same scale and directly comparable. The noise comes from many different sources including sample RNA preparation, hybridization, and scanning. Quantile normalization assumes that data from each array comes from the same intensity distribution and preserves the rank order of the genes within each array. After normalization, GCRMA uses a robust estimation procedure in a linear additive model for gene summary to protect against outlier probes.

DChip uses a multiplicative model within a gene for probe level gene summary, and uses a specific outlier detection method to detect outliers. It normalizes all arrays to a common baseline array having a median overall brightness. The basic assumption is that a probe of a non-differentially expressed gene in two arrays should have similar intensity ranks. It uses an iterative procedure to identify a set of probes called invariant set, which consists of points from non-differentially expressed genes. This procedure start with points of all PM probes. If the proportion rank difference between arrays is small, then the point is kept for the new set. The procedure continues until the number of points in the new set does not decrease anymore. Then, a non-linear curve is fitted to the invariant set, and all probe pair intensities are transformed in a way such that the fitted curve becomes the line y=x. A baseline array without too many outliers should be selected (LI and WONG 2003).

MAS is the method used by Affymetrix. It requires very few assumptions and does not use combined information from multiple arrays within an experiment. It uses a robust average method to summarize gene expression value for each gene on each array. For normalization, it scales all arrays to the same level. While avoiding errors caused by violated assumptions, it does not utilize all information available and is not robust against noise. In addition, it performs mismatch subtraction, which we believe is not optimal as discussed later. For comparison reasons, other than the standard MAS method, we also used MAS with perfect match data only (MASPM).

The three-step PLMMA approach involves: background subtraction, normalization, and gene-specific LMMA using probe-level data. The methods used for the background subtraction and normalization steps were the same as in GCRMA (IRIZARRY *et al.* 2003; WU and IRIZARRY 2005; WU *et al.* 2004). In the gene-specific LMMA step, a linear model including the probe by treatment interaction effect is fitted to the probe level data directly.

2.2 DATA AND METHODS

2.2.1 Data set

The data analysis was performed on a data set from a mouse caloric restriction (CR) experiment performed at the Laboratory for Nutrition and Vision Research, under the direction of Jean Mayer USDA HNRC on Aging at Tufts University. To date, CR is the only well established treatment for extending the natural life span (MASORO 1988). The precise biological mechanism responsible for this effect, however, remains largely unknown. There is keen interest in developing methods to determine the mechanism which provides this benefit.

This experiment used Affymetrix Mgu74Av2 Genechips, which have 12,488 gene specific probe sets on each chip. Each gene is represented by 8 to 21 pairs of 25-mer oligonucleotides. Each probe pair has one perfect match (PM) and one mismatch (MM) oligonucleotide probe. PM probes are exact complements to the gene sequence. MM probes contain a point mutation at the center of sequence, so that they can be used to assess non-specific hybridization and scanner offset. The signals from the probe pairs for each gene can either be summarized to gene level data or be used directly to fit a model.

Probe preparation and hybridization were performed at the Virginia Bioinformatics Institute-Core Laboratory Facility. Ten chips were run each time for two microarray experiments, with RNA samples from five biological replicates of each treatment group (CR group and control-fed group). The two experiments were conducted at different times, four months apart, with different operators and array scanners. RNA concentrations, as determined by Bioanalyzer, were all adjusted to 2μg/μl.

To confirm differential expression, real-time PCR analysis were performed using RNA from five control mice and four CR mice on some genes detected as being present and differentially expressed with very low fold changes. In a prior PCR experiment, another four genes that were identified as being differentially expressed at the protein level were tested and this data was also included as part of the validation.

2.2.2 Data analysis

2.2.2.1 PLMMA

LMMA analyzes designed microarray experiments and can be applied to very complex multifactorial designs (CHU *et al.* 2002). The three-step PLMMA approach described here directly models probe level data and accommodates known sources of variability across all arrays via variance components.

First, background correction was performed by fitting a model which utilized GC content to estimate the background noise (WU and IRIZARRY 2005; WU *et al.* 2004). Then, quantile normalization (BOLSTAD *et al.* 2003) was performed on all arrays. In the third step, we analyzed the normalized probe level data using gene-specific LMMA. The treatment, probe and treatment by probe interaction factors were modeled as fixed, while array effects were random.

For the combined analysis of the two separate experiments, the following gene-specific model was fitted to each gene:

$$\log_2(S_{ijkl}) = \mu + E_i + T_j + (A{:}T)_k + P_l + P_l * T_j + {}_{ijkl} \qquad (2.1)$$

S_{ijkl} denotes the normalized signal of the i^{th} experiment, j^{th} treatment (CR vs. control), the k^{th} array at the l^{th} probe**.** The symbols *E, T, A* and *P* represent **E**xperiment (referring to the two separate experiments), **T**reatment (referring to the biologic effects of calorie restriction), **A**rray and **P**robe main effects, respectively. μ denotes the intersection and ε_{ijkl} denotes random error. The A_k's and ε_{ijkl}'s were assumed to be independent and identically distributed (iid.) normal random effects with a mean of zero and variance $\sigma_a^2$and σ^2, respectively. Separate analyses for each experiment were also performed as a check for consistency. In the separate analyses, the E_i's were dropped from the model.

2.2.2.2 MAS 5.0 (AFFYMETRIX 2002)

MAS 5.0 uses background subtraction, idealized mismatch subtraction, and probe level data aggregation via a Tukey Biweight Algorithm to generate gene level signals (AFFYMETRIX 2002). MAS estimates the background noise within grids using a weighted average method. It also performs absolute expression analysis to detect expression. That is achieved by first calculating a discrimination value (PM-MM)/(PM+MM) for each probe pair to determine a detection p-value using the one-sided Wilcoxon's signed rank test. This test assigns each probe pair a rank based on how far the probe pair discrimination value is from a certain threshold. Then, the p-value is compared to two cutoffs and a detection call of "Present", "Marginal", or "Absent" is given for each gene.

The signal for each gene on the array was calculated in the following manner:

$$S_{ij} = \text{TukeyBiweight}\,(\log(PM_{ijk}) - \log(IM_{ijk})) \qquad (2.2)$$

IM is Idealized Mismatch, which is equal to MM if MM is less than PM. If MM is larger than PM, an adjusted MM value is created based on the biweight mean of the PM and MM ratio. IM is used to avoid the problem of negative signal values. Then, the signals were scaled so that all arrays had a median target signal of 500.

2.2.2.3 DChip (LI and WONG 2003)

The software package dChip (LI and WONG 2003) detects outliers by fitting a model for each probe set and iteratively dropping out probe and array outliers with large standard errors (more than three times as large as the median standard deviation). Array outliers denote the arrays that have different patterns from other arrays within a gene. Probe outliers refer to the probes showing different patterns from other probes within a gene (LI and WONG 2003). dChip does not perform background correction.

The following model was fitted to the normalized data for each gene to obtain the Model-Based Expression Indexes (MBEI):

$$PM_{il} = \theta_i\phi_j + \gamma_{il}, \ \Sigma\phi_j^2 = J \qquad (2.3)$$

Where PM_{il} is the normalized PM signal for chip i and probe j, θ_i is the MBEI in chip i, ϕ_j is the multiplicative probe effect, and γ_{il} is the error term. A summation constraint is imposed on ϕ_j's. DChip assumes a multiplicative model within a gene (LI and WONG 2003). We treated MBEI from array outliers as missing values.

2.2.2.4 GCRMA (IRIZARRY et al. 2003; WU and IRIZARRY 2005; WU et al. 2004)

GCRMA estimates background noise based on a model using GC content (WU and IRIZARRY 2005; WU *et al.* 2004) and normalizes across all arrays using quantile normalization (BOLSTAD *et al.* 2003). Since non-specific binding tends to be related to the content and location of Guanosine (G) and Cytosine (C) in the probe sequences, this background correction method takes GC content into account and uses a stochastic model to adjust for cross hybridization. To summarize the probe level data to the gene level, GCRMA background adjusts the data and then fits a robust linear model to each gene:

$$\log_2(y_{ij}) = rma_i + k_j + \xi_{ij} \qquad (2.4)$$

Here y_{ij} is the background-adjusted signal for chip i and probe j, rma_i is the summarized gene signal for chip i, k_j is the probe effect, and ξ_{ij} is the random error. Rma_i is estimated using a robust estimation method called median polish. Median polish fits the model iteratively, and successively removes row and column medians. It accumulates the terms until the process converges (the rows in each dimension have median close to zero).

For the three methods described above, after obtaining the gene level signals, the following gene specific model was fitted to the summarized gene signals to test differential expression:

$$\log_2(S_{ijk}) = \mu + E_i + T_j + \eta_{ijk} \tag{2.5}$$

Here S_{ijk} is the summarized signal obtained using MAS, dChip or GCRMA. In the separated analysis for the two experiments, the E_i term was dropped from the model. After all analysis were performed, we filtered out the genes that were detected as absent on all arrays in either of the experiments. The absent calls were obtained using MAS5.0.

For multiple testing adjustments, we used both the Bonferroni family-wise type I error rate (FWER) control method and the positive False Discovery Rate (FDR) with Q-value (STOREY 2002). FDR controls the expected proportion of false positives among all genes detected as being differentially expressed (BENJAMINI and HOCHBERG 1995). The Q-value method provides an asymptotic form of simultaneous controlling of the FDR at all levels when the p-values are weakly dependent (STOREY and TIBSHIRANI 2003). For gene expression experiments FDR control has an obvious advantage over FWER control – the detection power does not go down as much when the number of tests increases. For microarray experiments with a large number of hypotheses, the Bonferroni adjustment is too stringent and has very little power to detect differentially expressed genes.

2.3 RESULTS AND DISCUSSION

The microarray data exhibited very high quality, with few outliers and high correlation between biological replicates. With relatively smaller technical errors, many genes were detected as being differentially expressed between the control and CR animals. Here, we reported the number of genes detected in the different statistical methods and compared the results based on the following criteria: number of genes detected, consistency between methods, effect of MM subtraction and outliers, and soundness of detection (based on biological knowledge and RT-PCR results). These statistical methods performed differently but they showed consistent results on many genes, especially those that were very significantly differentially expressed. We also reported the results on some of the genes that we found of interest.

2.3.1 Data quality assessments

With dChip's outlier detection method, we found only around 0.1% of the total data were outliers. We also used some diagnostic plots such as the RNA degeneration plots and correlation plots to check data quality. Figure 2.1 shows some examples of the correlation plots obtained using data summarized by dChip (LI and WONG 2003). Biological replicates from the same group in the same experiment showed very good correlation (at least 97.6%). The correlations became less significant for arrays from different treatment groups or experiments (at least 93.0%), as expected.

2.3.2 Differentially expressed genes

Many differentially expressed genes were detected using different statistical methods. Table 2.1 shows the number of genes detected by each method and how many of them were detected simultaneously by different methods. Separate and combined analyses were performed for the two microarray experiments. For each method, the number of genes that were reproducible from both experiments was also listed.

After Bonferroni adjustment, the number of genes detected as differentially expressed was small using any method, especially when only ten arrays were used and the degrees of freedoms of the tests were small. The power was very low after the Bonferroni adjustment. The number of genes detected after Q-value (STOREY 2002) FDR multiple testing adjustment was in the more reasonable range considering the fact that the mice between the control and CR group were very different. For microarray experiments with a large number of tests and many expected true alternative hypothesizes, the Bonferroni adjustment is not preferable because it is too stringent. In the combined analysis where the error can be estimated more precisely, all the methods detected many genes using the Q-value FDR control. In the combined analysis, GCRMA detected the smallest number of genes as significant. GCRMA uses median polish to summarize probe level signals, therefore is robust against outliers. However, in experiment 1, only 59% of the 726 genes detected by PLMMA were also detected by GCRMA, while 86% of them were detected as significant by GCRMA in the combined analysis with all 20 arrays. In experiment 2, only 48% of the 545 genes detected by PLMMA were also detected by GCRMA, while 90% of them were detected as significant by GCRMA in the combined analysis. The results suggested that GCRMA may have sacrificed power for robustness against outliers.

In the combined analysis of the two microarray experiments, 855 genes were detected as being differentially expressed by all the methods using the Q-value multiple adjusting method. 97 genes were detected as being significant by all statistical methods using the Bonferroni adjustment. Many of these genes are related to stress and metabolic abnormalities due to calorie restriction. For example, the immune response and electron transport genes may slow aging by improving antioxidant defense mechanisms (KOUBOVA and GUARENTE 2003); the biosynthesis, cell growth/maintenance, and protease inhibitor genes may help slow accumulation of abnormal proteins by speeding up protein turnover (SOHAL and WEINDRUCH 1996; TAYLOR *et al.* 1989). Complement component 9, which is related to cell death, is very significantly down regulated.

Among those genes detected with less stringent criteria, we identified a number of genes that may be related to aging, such as inflammatory response genes, cancer related genes and insulin like genes. Table 2.2 lists some of these genes and the statistical methods that detected them as significant.

Sirtuin 3 was detected as being up regulated by all methods. Sirtuin 3 is a Sir2 homolog gene. Sir2 is known to affect aging in some organisms. Lowered glucose due to CR may impose a state of partial energy limitation and thus increase Sir2 expression, which in turn extend life span (KAEBERLEIN *et al.* 1999; KOUBOVA and GUARENTE 2003; LIN *et al.* 2000).

Serum amyloid may affect aging by influencing the inflammatory response pathway. Glucose 6 phosphatase is on the glycolytic and gluconeogenic pathways of the liver and may affect aging by increasing protein turnover. Proteosome genes may also affect aging via similar ways.

2.3.3 Comparison with protein results

Prior proteomic analyses identified four proteins that were at significantly lower levels in CR vs. control mice. Results on the genes corresponding to these proteins were also conformed by RT-PCR. These genes are:

- major urinary protein (alpha2u-globulin)
- betaine homocysteine S-methyltransferase
- glutathione S-transferase-pi (GST-pi-1)
- carbonic anhydrase I (RT-PCR also tested Carbonic anhydrase 3)

The expression levels of genes for all major urinary proteins significantly decreased in expression values under CR, and the results were consistent for all statistical methods. There are two betaine-homocysteine methyltransferase genes, which were detected as significantly decreased by PLMMA and MAS methods. Glutathione S-transferase, pi 1 was detected as being significantly decreased in expression by all methods except GCRMA. Carbonic anhydrase 1 was not detected as present on any array (conformed by RT-PCR). Carbonic anhydrase 3 was found to be decreased in expression under CR. Expression analysis results of these genes are shown in Table 2.3.

The probe set AV279130 (betaine-homocysteine methyltransferase) was only detected as “present” in two arrays. As shown in Figure 2.2, most probes of AV279130 (on the left of Figure 2.2) had MM values that were not less than PM except for the last two probes.

therefore, it was probably a false positive. Those two probes were detected by dChip as outliers. GCRMA is robust against outliers, therefore GCRMA and dChip did not find AV279130 as being significantly expressed. Probe set X53451 (glutathione S-transferase, on the right of Figure 2.2) was detected as being differentially expressed by all methods except GCRMA. The expression differences between the CR and control groups were relatively small and the probes were variable, therefore the probe set was not detected as significant by GCRMA.

2.3.4 Results from PCR experiment

Aside from those genes analyzed in the protein study, real-time PCR analysis was only performed on genes with very low fold changes. However, due to low expression values, the PCR results had large standard errors. PCR on these genes were performed more than two years after the RNA samples were prepared, which could lead to the large standard error. For those genes, the microarray differential expression results did not match PCR results very well. It might be resulted from errors in the PCR analysis, or due to the fact that the tested genes had relatively low median signal, and hence might not be reliably detected by the array scanner. This suggests that we may filter the genes based on both the present/absent calls and the raw values below a certain threshold, or variances of the gene expressions. One gene identified in the protein study was detected as all absent by the microarray experiments. It was confirmed by the PCR study as absent.

2.3.5 Results on a “housekeeping" gene

Glyceraldehyde-3 phosphate dehydrogenase (GAPDH) is a housekeeping gene on the array. We examined the summarized expression values of the six probe sets of GAPDH on each aray using different statistical methods. Since PLMMA does not summarize the probe level data to the gene level, the average predicted values from the gene specific model with only array, probe and probe*treatment effects were used. All methods stabilized variance of GAPDH, except MAS (Table 2.4). The result showed that MAS added additional variability to the data, possibly through mismatch subtraction (Figure 2.3).

2.3.6 Pathway analysis

The detected significant genes can be checked against known pathway information to identify possible differentially expressed gene networks. This would give us some "reference genes" for further gene network inference. The detected significant genes were then submitted to a web tool called "pathway express" (DRAGHICI 2005). Several pathways were found to be associated with genes that were identified by each method. Table 2.5 lists some of those pathways identified by the PLMMA method.

2.4 CONCLUSIONS

With the development of more advanced technologies to reduce technical errors and better statistical methods for testing, genes with relatively smaller changes in expressions can be detected. We found that consistent results were obtained with various methods for genes that were very highly differentially expressed. For example, sirtuin 3, a homolog to the well-known aging related gene Sir2, was detected by all methods. However, different methods gave very different results with respect to genes with subtle changes in expression values.

No method performed consistently well for all the genes, since all methods tested have some underlying assumptions and use different mechanisms to extract valuable information from the data. The RT-PCR results did not give much insight on the performance of methods for genes with relatively subtle changes in expression. That suggests, other than filtering the genes using present/absent calls, we may also filter based on some threshold value of the raw signals, or variances of the genes. Though no method outperformed other methods in all cases, the importance of outlier detection and the drawback of MAS mismatch subtraction were shown. When probe outliers were presented in a probe set, dChip and GCRMA avoided possible false positives by removing outliers or using robust estimation for gene summary. GCRMA and MAS detected smaller numbers of significant genes than the others. Results suggested that in this dataset GCRMA may have sacrificed power when guarding against outliers by using the median polish gene summary method. MAS mismatch subtraction may add variability to the data, as shown in a housekeeping gene.

AUTHORS' CONTRIBUTIONS

Bing Liu carried out the analysis and drafted the manuscript. Dr. Fu Shang and Dr. Allen Taylor designed and performed the mouse experiment. Dr. Ina Hoeschele and Dr. Karen Duca directed the analysis.

REFERENCES

AFFYMETRIX, 2002 *Affymetrix ® Microarray Suite User's Guide*, Santa Clara, CA.

BENJAMINI, Y., and Y. HOCHBERG, 1995 Controlling the false discovery rate—a practical and powerful approach to multiple testing. J. Roy. Stat. Soc. B **57:** 289-300.

BOLSTAD, B. M., R. A. IRIZARRY, M. ASTRAND and T. P. SPEED, 2003 A comparison of normalization methods for high density oligonucleotide data based on variance and bias. Bioinformatics **19:** 185-193.

CHU, T. M., B. WEIR and R. WOLFINGER, 2002 A systematic statistical linear modeling approach to oligonucleotide array experiments. Math. Biosci. **176:** 35-51.

DRAGHICI, S., 2005 *Pathway Express*, http://vortex.cs.wayne.edu/projects.htm.

HOESCHELE, I., and H. LI, 2005 A note on joint versus gene-specific mixed model analysis of microarray gene expression data. Biostatistics **6:** 183-186.

IRIZARRY, R. A., B. HOBBS, F. COLLIN, Y. D. BEAZER-BARCLAY, K. J. ANTONELLIS *et al.*, 2003 Exploration, normalization, and summaries of high-density oligonucleotide array probe level data. Biostatistics **4:** 249-264.

KAEBERLEIN, M., M. MCVEY and L. GUARENTE, 1999 The SIR2/3/4 complex and SIR2 alone promote longevity in Saccharomyces cerevisiae by two different mechanisms. Genes Dev. **13:** 2570-2580.

KOUBOVA, J., and L. GUARENTE, 2003 How does calorie restriction work? Genes Dev. **17:** 313–321.

LI, C., and W. H. WONG, 2001 Model-based analysis of oligonucleotide arrays: model validation, design issues and standard error applications. Genome Biol. **2:** 1-11.

LI, C., and W. H. WONG, 2003 DNA-Chip Analyzer (dChip) in *The analysis of gene expression data: methods and software*, edited by G. PARMIGIANI, E. GARRETT, R. IRIZARRY and S. ZEGER. Springer.

LIN, S. J., P. A. DEFOSSEZ and L. GUARENTE, 2000 Requirement of NAD and SIR2 for life-span extension by calorie restriction in Saccharomyces cerevisiae. Science **289:** 2126-2128.

MASORO, E. J., 1988 Food restriction in rodents: an evaluation of its role in the study of aging. J. of Gerontol. **43:** B59.

SOHAL, R. S., and R. WEINDRUCH, 1996 Oxidative stress, caloric restriction, and aging. Science **273:** 59-63.

STOREY, J. D., 2002 A direct approach to false discovery rates. J. Roy. Statist. Soc. Ser. B **64:** 479-498.

STOREY, J. D., and R. TIBSHIRANI, 2003 Statistical significance for genomewide studies. Proc. Natl. Acad. Sci. USA **100:** 9440-9445.

TAYLOR, A., A. M. ZULIANI, R. E. HOPKINS, G. E. DALLAL, P. TREGLIA *et al.*, 1989 Moderate caloric restriction delays cataract formation in the Emory mouse. FASEB J. **3:** 1741-1746.

WU, Z., and R. A. IRIZARRY, 2005 *A Statistical Framework for the Analysis of Microarray Probe-Level Data, Working Papers 73*. Johns Hopkins University, Dept. of Biostatistics Working Papers.

WU, Z., R. A. IRIZARRY, R. GENTLEMAN, F. M. MURILLO and F. SPENCER, 2004 A model based background adjustment for oligonucleotide expression arrays. J. Amer. Stat. Assoc. **99:** 909-917.

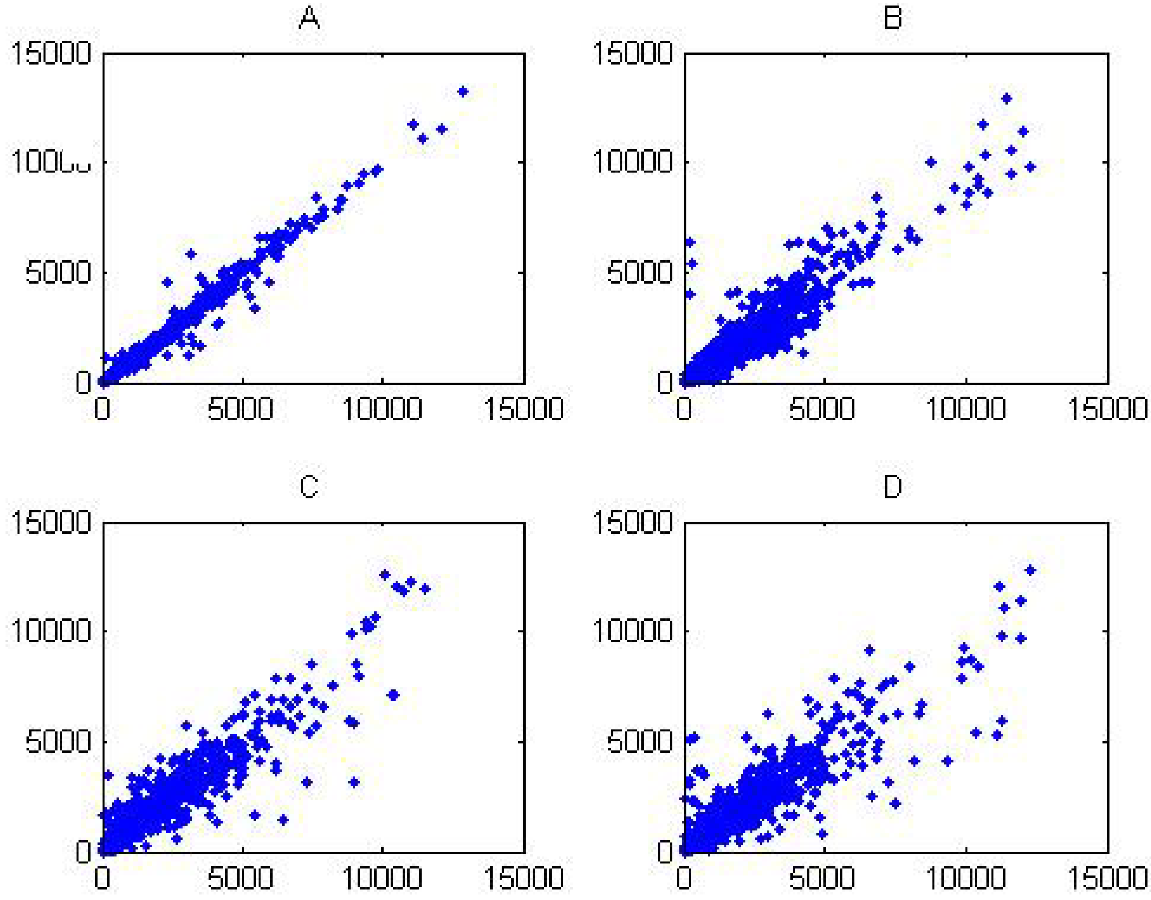


FIGURE 2.1.— Bivariate correlation plots showing correlations of summarized intensities of two arrays

The two axes show summarized intensities of two of the arrays in the mouse study. A: correlation of two arrays from the same treatment group, same experiment. B: two arrays from the same treatment group, different experiments. C. Two arrays from the same experiment but different treatment groups. D: Two arrays from different treatment groups, different experiments.

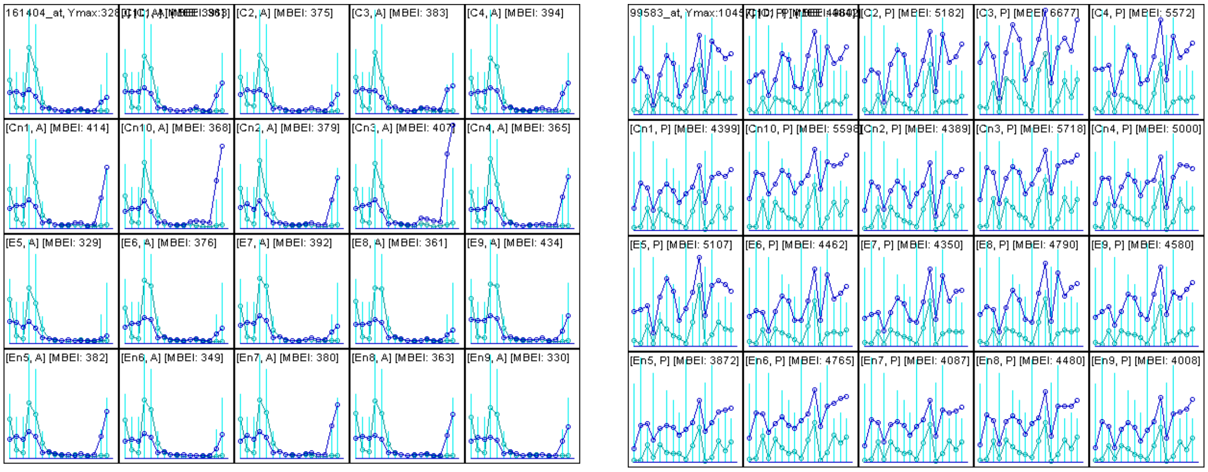


FIGURE 2.2.— Expression profiles of two genes

The plots were produced using dChip (LI AND WONG 2001). Left: Expression of the 20 probes of AV279130. The 10 grids at the top show the 10 arrays for the control mice, and the 10 grids at the bottom are for the CR mice. Green lines are for the mismatch data; blue lines are for the perfect match data. Right: Expression of X53451.

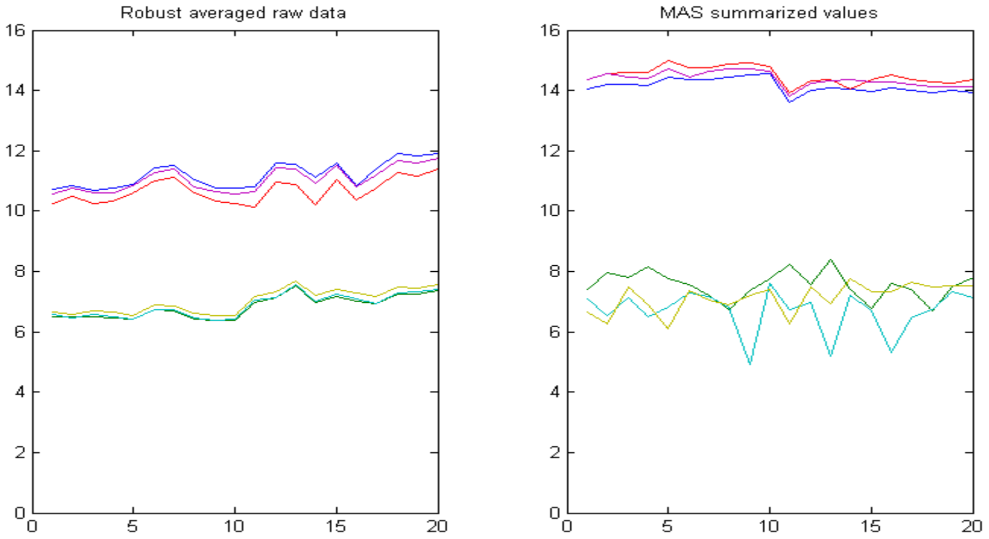


FIGURE 2.3.— Expression profiles of GAPDH on the 20 microarrays

Left: robust averaged raw data. Right: MAS summarized values. MAS summarized values added variability to the data.

TABLE 2.1 Number of differentially expressed genes detected by the different methods

# of Differentially Expressed Genes Detected Using Bonferroni Adjustment

	Experiment 1					Experiment 2					Combined analysis				
	PLMMA	DChip	GCRMA	MASPM	MAS	PLMMA	DChip	GCRMA	MASPM	MAS	PLMMA	DChip	GCRMA	MASPM	MAS
PLMMA	24	8	14	9	8	9	4	4	5	4	272	176	182	176	157
DChip		20	6	6	5		21	5	5	4		277	154	143	130
GCRMA			24	9	8			9	5	4			236	146	146
MASPM				15	5				11	6				256	150
MAS					12					8					216
Percentage of genes in the combined analysis predicted:															
	3.3	6.9	8.5	5.5	5.6	7.7	6.5	3.4	3.9	3.7					
Common ones from both experiments:						1	3	1	1	0					

# of Differentially Expressed Genes Detected Using Q-value FDR Adjustment

	Experiment 1					Experiment 2					Combined analysis				
	PLMMA	DChip	GCRMA	MASPM	MAS	PLMMA	DChip	GCRMA	MASPM	MAS	PLMMA	DChip	GCRMA	MASPM	MAS
PLMMA	726	461	429	457	371	545	393	242	305	252	1,927	1,460	1,337	1,464	1,206
DChip		788	376	417	326		820	224	294	237		2,175	1,230	1,404	1,110
GCRMA			542	363	328			264	203	177			1,495	1,243	1,070
MASPM				837	420				437	234				2,081	1,328
MAS					568					341					1,510
Percentage of genes in the combined analysis predicted:															
	36.3	14.5	33.2	39.1	34.7	27.6	33.7	12.9	20.5	21.7					
Common ones from both experiments:						319	378	217	276	186					

The values on the diagonal are the number of genes detected by that method. The values on the off diagonal are the number of genes detected simultaneously by two methods. The number of genes detected simultaneously by experiment 1 or 2 and the combined analysis divided by the number of genes detected in combined analysis give the percentages in the table. The last rows in the tables show the number of genes that were reproducible from both experiments.

TABLE 2.2 Some genes detected with the Q-value FDR control that may be related to aging

Gene Name	Log Fold change	PLMMA	dChip	GC RMA	MAS PM	MAS
sirtuin 3 (silent mating type information regulation 2, homolog) 3 (S. cerevisiae)	0.3479	X**	X**	X*	X**	X
ubiquitin B	0.3518	X	X		X	
insulin-like growth factor 1	0.2283		X			
serum amyloid A 2	-0.3614	X**	X**	X*	X**	X**
serum amyloid A 2	-0.9303	X**	X*	X**	X**	X**
serum amyloid A 3	-0.7	X*	X**	X**	X**	X*
serum amyloid A 4	-0.3856	X**	X**	X**	X**	X**
glucose 6 phosphatase, catalytic, 3	-0.0712					X*
proteosome (prosome, macropain) subunit, beta type 8 (large multifunctional protease 7)	-0.1776	X*	X*	X*	X*	X
proteosome (prosome, macropain) subunit, beta type 9 (large multifunctional protease 2)	-0.3627	X**	X**	X*	X**	X*

For each method, an "X" indicates that the gene was detected as significant by that method. "X*" indicates that the gene was detected as significant in two of the three analyses (experiment 1, experiment 2 and combined) for that method. "X**" indicates that the gene was detected as significant in all three analyses.

TABLE 2.3 Results from genes of the analyzed proteins

Gene Name	Log Fold change	PLMMA	dChip	GCRMA	MASPM	MAS
betaine-homocysteine methyltransferase	-0.6066	**X**	**X**		**X**	**X**
betaine-homocysteine methyltransferase	-0.1242	**X**			**X**	**X**
carbonic anhydrase 3	-0.4751	**X***	**X**	**X**	**X**	**X***
glutathione S-transferase, pi 1	-0.1832	**X**	**X**		**X**	**X**

The first two genes are for the same protein. For each method, an "X" indicates that the gene was detected as significant by that method. "X*" indicates that the gene was detected as significant in two of the three analyses (experiment 1, experiment 2 and combined) for that method.

TABLE 2.4 Variances of summarized expression values of GAPDH from the 20 microarrays

	Averaged raw data	**dChip**	**GCRMA**	**MASPM**	**MAS**	**PLMMA**
Variance	0.160741	0.0168	0.0259	0.041	0.199	0.041

The numbers were averaged from the six probe sets.

TABLE 2.5 Pathways associated with genes detected by the PLMMA method

Pathway name	# gene in pathway	# Input gene in pathway	# Pathway genes on chip	% pathway genes	Diff ratio
Adherens junction	76	18	60	30.0	1.9
Huntington's disease	27	7	22	31.8	2.1
Tight junction	125	19	78	24.4	1.6
Dentatorubropallidoluysian atrophy (DRPLA)	14	3	11	27.3	1.8
Complement and coagulation cascades	70	13	59	22.0	1.4
Notch signaling pathway	50	7	30	23.3	1.5

"# gene in pathway" denote the total number of genes in the pathway. "# Input gene in pathway" denote the number of genes both in the pathway and among the genes detected as differentially expressed by PLMMA. "# Pathway genes on chip" denote the number of pathway genes on the microarray. "% pathway genes" denotes percentages of pathway genes on the chips that were significant. "Diff ratio" denotes the ratio of "# input gene in pathway"/total input genes divided by the ratio of pathway genes/total genes on the chips. The diff ratios show that the pathways in the table were over-represented in the genes that were detected as significant.

Chapter 3

From genetics to gene networks:

Evaluating approaches for integrative analysis of genetic marker and gene expression data for the purpose of gene network inference

Bing Liu[¶§], Alberto de la Fuente[§] and Ina Hoeschele[§¶]

[¶]Department of Statistics, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061

[§]Virginia Bioinformatics Institute (0477), Virginia Polytechnic Institute and State University, Blacksburg, VA 24061

ABSTRACT

In a genetical genomics experiment a segregating population of hundreds of individuals is expression profiled for thousands of genes and genotyped for hundreds or thousands of genetic markers. Expression Quantitative Trait Locus (eQTL) mapping treats gene expression levels as quantitative traits and has the goal of identifying genomic regions causally affecting gene expression levels. With the identified eQTL regions, using DNA sequence information, genes co-located with an eQTL on the chromosome can be identified as the candidate causal regulators of the gene expression levels of the affected gene. After using local structural models to identify such candidate regulators in each eQTL, an Encompassing Directed Network (EDN) of causal relationships among genes can be constructed.

The objective of the present work is to evaluate several eQTL mapping approaches and local structural models in their ability for constructing an EDN. Several eQTL mapping approaches were evaluated: Single Profile Analysis (SPA), Principal Components (PC) mapping, cis-eQTL and trans-eQTL mapping. Results on a genetical genomics dataset from yeast showed that the PC-mapping, cis-eQTL and trans-eQTL mapping greatly increased power for the eQTL detection as compared to the SPA. The combined eQTL mapping results from the PCA, cis and trans mapping detected most eQTLs found in SPA and much more, and therefore should be a good starting point for EDN construction. For regulator-target pair identification, our local structural models performed well on the simulated data set for identifying regulators within one eQTL region, except for a case where some genes have extremely high and other genes low heritability. For the yeast data set, an EDN constructed based on the combined results from SPA, cis and trans-mapping included 28,609 regulator-target pairs. It still

contains many direct regulations that are actually indirect, as well as multiple candidate regulators for some eQTLs and targets. It is therefore important to perform further sparsification of the network.

3.1 INTRODUCTION

Gene networks can be represented as graphical models in which nodes are genes, and undirected and directed edges correspond to interactions and causal influences between the nodes, respectively. Gene networks are coarse-grained descriptions of cellular physiology in the sense that only relationships between gene expression levels are modeled, and many other components such as proteins and metabolites are not explicitly taken into account. Nevertheless, gene networks are system-level descriptions of cellular physiology and will improve our understanding of the genetic architecture of complex traits and diseases. Gene networks have many practical applications (BRAZHNIK *et al.* 2002), including the discovery of direct drug targets (DI BERNARDO *et al.* 2005; GARDNER *et al.* 2003). It has been shown that some classical concepts in genetics, such as dominance and epistasis, can be understood in terms of networks and their properties (KACSER and BURNS 1981; OMHOLT *et al.* 2000). Many strategies have been proposed to obtain gene networks from gene expression data, such as probabilistic models, (e.g., FRIEDMAN 2004; FRIEDMAN *et al.* 2000) , time series analysis using linear models, (e.g., D'HAESELEER *et al.* 1999; VAN SOMEREN *et al.* 2000), partial correlation analysis, (e.g., DE LA FUENTE *et al.* 2004) and several perturbation approaches (e.g., DE LA FUENTE *et al.* 2001; DE LA FUENTE *et al.* 2002; GARDNER *et al.* 2003; WAGNER 2001). Different approaches to gene network inference are reviewed by BRAZHNIK *et al.* (2002), D'HAESELEER *et al.* (2000), DE JONG (2002), and GARDNER and FAITH (2005).

Causal inference for genes can be achieved with a strategy of creating targeted perturbations (interventions) and measuring the responses of gene expression levels to those perturbations. It has been shown that such an approach can provide a reliable identification of gene networks

(DE LA FUENTE *et al.* 2001; DE LA FUENTE *et al.* 2002; GARDNER *et al.* 2003). There are two major types of targeted perturbation experiments: one uses one-at-a-time, specific perturbations (e.g., CARPENTER and SABATINI 2004; HUGHES *et al.* 2000; SACHS *et al.* 2005), and the other uses naturally occurring multi-factorial perturbations in segregating populations (genetical genomics, JANSEN 2003; JANSEN and NAP 2001; JANSEN and NAP 2004). We focus our network inference on genetical genomics data. In a genetical genomics experiment, a segregating population of hundreds of individuals is expression profiled for thousands of genes and genotyped for hundreds or thousands of genetic markers. The variation in the expression levels of genes is influenced by the variation in many polymorphisms (genotypes) across the genome. The genotypes can thus be regarded as natural multifactorial perturbations (JANSEN 2003; JANSEN and NAP 2001; JANSEN and NAP 2004) resulting in different gene expression "phenotypes", and a relationship can be established between the measured genotypes and the measured gene expression phenotypes. In contrast to the approaches using specific experimental perturbations, in genetical genomics we do not know where the perturbations arise and we must identify their origin. This can be achieved by expression Quantitative Trait Locus (eQTL) mapping, which treats the gene expression profiles in a segregating population as quantitative traits and performs Quantitative Trait Locus (QTL) mapping on those traits. QTL mapping identifies the polymorphic genomic regions having significant effects on a quantitative trait. Compared to the traditional QTL mapping, eQTL mapping is performed on a much larger scale: there are thousands to ten thousands of correlated expression traits (etraits). The result of eQTL analysis is the knowledge that certain genomic regions likely have causal effects on the expression levels of particular genes. Then, by using DNA sequence information, genes located in an eQTL region can be identified as candidate causal regulators of the genes whose expression levels are affected by that eQTL. After the identification of the candidate regulatory genes in each eQTL, an Encompassing Directed Network (EDN) of causal regulatory relationships among genes can be constructed. The constructed EDN can be represented as a graph consisting of gene nodes and eQTL nodes. The directed edges in the EDN correspond to causal relationships or regulations among pairs of genes. A set of sparser networks well supported by the data can be found by searching within the space defined by the EDN. The main purpose of the present work is to evaluate several eQTL mapping approaches and local structural models for the selection of regulator-target pairs in their ability to construct an EDN from the eQTL results. Sparsification of the EDN using a global structural modeling approach will be addressed in another contribution.

In a "traditional" eQTL mapping approach, eQTL analysis is performed on each etrait separately and adjusted for multiple testing using the false discovery rate (FDR) (BENJAMINI and HOCHBERG 1995) control, after retaining only one eQTL per etrait or per etrait and chromosome (BING and HOESCHELE 2005; BREM and KRUGLYAK 2005; BREM *et al.* 2002). This approach will be referred to as Single Profile Analysis (SPA). In this contribution, we repeat the SPA, but retain multiple separated, significant test statistics peaks on each chromosome.

SPA overlooks the fact that etraits are correlated and therefore does not have optimal power for the detection of pleiotropic eQTLs (*i.e.,* eQTLs affecting multiple etraits). It has been shown that multi-trait mapping is more powerful than single trait mapping for detecting pleiotropic QTLs (JIANG and ZENG 1995). However, it is not computationally feasible to perform multi-trait mapping on thousands of etraits. For small sets of correlated traits, QTL mapping of Principal Components (PC) has been shown to be equivalent to multi-trait mapping, without the drawback of increased computational complexity (MANGIN *et al.* 1998). Several groups have used PCs for QTL or eQTL mapping in different experimental settings (BOOMSMA 1996; CHASE *et al.* 2002; COMUZZIE *et al.* 1997; LAN *et al.* 2003; LIU *et al.* 1996; WELLER *et al.* 1996; ZENG *et al.* 2000). However, only a small number of traits were analyzed in those studies. Here, we perform the Principal Component Analysis (PCA) on the entire set of 4,589 filtered yeast genes (BREM and KRUGLYAK 2005), as well as on subsets of genes obtained by gene clustering.

Expression QTL mapping and regulator gene identification can be performed much more effectively by taking into account two distinct types of genetic regulation: cis-regulation and trans-regulation. In the case of cis-regulation, the cis-eQTL affects a particular etrait X (expression level of gene X) and is located at the physical location of gene X on the chromosome. The polymorphism of a cis-eQTL likely corresponds to a promoter region polymorphism of the gene (e.g. DOSS *et al.* 2005; RONALD *et al.* 2005; JANSEN and NAP 2001). If gene X regulates the expression of some other genes, the eQTL that cis-affected gene X will have an indirect effect on the expression of those genes through gene X (KULP and JAGALUR 2006; DOSS *et al.* 2005). Such indirect effects have been referred to as cistrans effects (KULP and JAGALUR 2006). Trans-eQTLs influence the expression levels of genes, but do not need to be co-located with any of these genes. The polymorphism of a trans-eQTL likely comes from a coding region polymorphism in a regulator gene located at the eQTL (e.g. YVERT *et al.* 2003, JANSEN and NAP 2001). While a trans-eQTL does not affect the expression level of the regulatory gene, the coding region polymorphism affects the kinetic properties of the regulatory protein encoded by that gene, which in turn affects the expression levels of the targets.

Since by definition the location of a cis-eQTL must physically coincide with the location of the gene whose etrait is affected, only the marker(s) closest to the location of an etrait's gene is tested to detect cis-eQTLs (e.g. DOSS *et al.* 2005; RONALD *et al.* 2005). For network inference, such cis-linked etraits are not very informative. As shown on mouse data (DOSS *et al.* 2005), the secondary targets of the cis-eQTLs, the so-called cistrans regulated etraits, can be obtained by testing the effects of the identified cis-eQTL regions on other etraits.

Trans-affected etraits are affected by the eQTL genotype and the etrait of the candidate regulator gene simultaneously. Therefore, it was proposed (KULP and JAGALUR 2006) that in order to specifically detect trans-eQTLs, mapping is best performed by including candidate "regulatory" etraits in the QTL model. KULP and JAGALUR (2006) performed interval mapping on any etrait *i* with a model including the effects on etrait *i* of another etrait *j*, the genotype at the physical location of gene *j*, and the etrait-by-genotype interaction. We performed mapping of trans-eQTLs also by including the candidate etrait in the model, but with a regression model and the intersection-union-test (IUT) (CASELLA and BERGER 1990; ROY 1957) to test whether the eQTL genotype and the etrait of the candidate regulator gene both significantly affected the target etrait.

The problem of identifying candidate regulatory genes from eQTL confidence regions has been approached by using partial correlation tests (BING and HOESCHELE 2005), analysis of the between-strain Single Nucleotide Polymorphisms (SNPs) (LI *et al.* 2005), assessing the extent of eQTL overlap between any two etraits (ZHU *et al.* 2004), and more recently using a stochastic model incorporating protein-protein interaction data (TU *et al.* 2006). We used local structural models separately for each eQTL to identify the regulator-target pairs, taking into account that an eQTL may affect a target through cis, trans or cistrans regulation.

In contrast with previous work (e.g. KULP and JAGALUR 2006, DOSS *et al.* 2005), in this contribution we consider cis, cistrans and trans regulations jointly with the goal of reconstructing an EDN that defines the network search space for a network reconstruction method which we report on in a separate contribution. This method is capable of reconstructing networks with cycles or feedback loops, an advantage over Bayesian networks that are currently used (LIU *et al.* 2006).

3.2 METHODS

The methodology we discuss here can be applied to any organism where a segregating population is extensively marker genotyped and expression profiled, and where DNA sequence information is available. Currently several such datasets have been produced, most noteworthy for yeast (BREM and KRUGLYAK 2005) and mouse (SCHADT *et al.* 2003). For evaluation purposes we analyzed the yeast genetical genomics dataset (BREM and KRUGLYAK 2005). After removing the 20% of genes with the lowest etrait variability from the original data, our dataset contained etraits for 4,589 genes and genotypes for 2,956 genetic markers on

112 haploid offspring from a cross between a laboratory and a wild strain. Observations with missing marker genotypes were excluded.

3.2.1 Single Profile Analysis

Marker linkages were tested using the Kruskal-Wallis test (LEHMANN 1975). QTL confidence intervals (CIs) were obtained by searching for markers at either side of the significant QTL marker that corresponds to a decrease in the logarithm-of-odds (LOD) score of at least 1 (LANDER and BOTSTEIN 1989). LOD 1 intervals are approximately equivalent to a 96.8% CI (MANGIN *et al.* 1994). The Kruskal-Wallis test statistic follows an approximate chi-square distribution under the null hypothesis, and approximate LOD scores were computed by dividing the Kruskal-Wallis test statistic by 2*ln(10). If multiple eQTLs on the same chromosome have significant effects on the same etrait, they have to be separated by at least two consecutive, insignificant markers to be regarded as different eQTLs. Any two eQTL regions with less than 50% overlap were treated as separated eQTL for any particular etrait. To identify chromosomal regions affecting multiple etraits, the eQTL regions of two different etraits were combined into a single region if the two eQTLs were located at the same marker or their CIs overlapped by over 80%.

The nominal p-values were calculated based on normality assumptions. Rebaï (REBAÏ 1997) showed that even if the data are not normally distributed, a normal approximation will not give misleading results if the distribution is not too extreme. We verified this assumption using permutation tests and observed that the nominal p-values were very close to the permutation-based p-values and were slightly more conservative. Therefore, nominal p-values were used in this study. The p-values were adjusted for multiple testing by controlling the FDR using the BH-procedure (BENJAMINI and HOCHBERG 1995).

Sliding three marker regression (BING and HOESCHELE 2005; THALLER and HOESCHELE 2000) was performed to fine map the LOD CIs. A marker to be tested was fitted together with its flanking markers in a regression model. The marker of interest has an expected nonzero partial regression coefficient if and only if at least one QTL is located between the flanking markers (ZENG 1993). To select the appropriate flanking markers, we chose the closest markers having at least 20 recombinants with the tested marker. The number of recombinants required was determined based on two criteria: sufficient power for the regression models to distinguish between the tested marker and the flanking markers, and sufficient proximity of the flanking markers to block the effects of the linked eQTL. A test statistic profile was obtained for all the markers in a LOD CI. New confidence regions were identified by looking for significant regions separated by at least two non-significant markers inside the LOD CI. Since the eQTL regions were already detected as significant, a p-value cutoff of 0.05 was used.

3.2.2 Principal Components Mapping

PCs were first computed on the total set of 4,589 etraits. Subsequently, to detect eQTLs affecting smaller subsets of genes, we clustered genes and applied PCA separately to the clusters. We used k-means with absolute correlation as the distance measure to cluster genes into 100 subsets with the software Cluster 3.0 (downloaded from http://bonsai.ims.u-tokyo.ac.jp/ ~mdehoon/software/cluster/software.htm). The number of genes in each cluster varied from 3 to 232, with an average of 46 genes per cluster. An eigen value cutoff of 1.5 was used to determine how many PCs to retain for each cluster, so that the PCs from different clusters contained a similar amount of information. Then, eQTL mapping was performed on these “composite etraits” (PCs) rather than the individual etrait in the same way as in the SPA. An eQTL affecting a PC is assumed to be a common regulator of all the etraits with high loadings on the affected PC. However, there was no clear cutoff for “high” loadings (Figure 3.1). Therefore, instead of choosing an arbitrary cutoff, all etraits were individually tested as in the SPA for the identified PC-eQTL regions to establish the influence of PC-eQTLs on individual etrait. As in the SPA, the tested PC-eQTLs on the same chromosome were required to be local maxima separated by at least two consecutive, insignificant markers.

3.2.3 Cis-eQTL Mapping

Cis-acting QTL effects were tested using the same non-parametric test as in the SPA. For those etraits with a significantly linked cis-marker, LOD confidence intervals were obtained using the LOD results from the SPA. We searched both sides of the tested marker for a LOD 1 drop and recorded the maximum LOD marker. If the gene of the tested etrait fell outside the CI and was more than 10 kb away from the maximum, we discarded the eQTL. The cis-eQTLs were combined in the same way as in the SPA. In the cistrans analysis, we tested all etraits for the effects of the identified cis-eQTL. As in the SPA, the tested cis-eQTLs were required to be local maxima separated by at least two consecutive, insignificant markers to be regarded as different eQTLs affecting that etrait.

3.2.4 Trans-eQTL Mapping

For the trans-eQTL mapping, we used a regression approach, as the Kruskal-Wallis test (LEHMANN 1975) cannot incorporate the etrait of a candidate regulator gene. Our regression model for the etrait of any gene *i* includes the effects on etrait *i* of a candidate regulator $j \neq i$, the genotype of the marker closest to the physical location of gene *j* and the etrait by genotype interaction term:

$$y_{in} = b_1 y_{jn} + b_2 x_{jn} + b_3 y_{jn} x_{jn} + \varepsilon_{in} \qquad (3.1)$$

where y_{in} is the deviation of etrait value *i* in observation *n* from its mean; y_{jn} is the deviation of the potential regulator etrait *j* in observation *n* from its mean; x_{jn} is the deviation of the genotype value (0 or 1) of the marker closest to candidate regulator gene *j* from its mean, and ε_{in} represents the residual. Regression coefficients b_1 and b_2 must both be significantly different from zero for gene j to be declared as a trans-regulator of gene *i*, as determined by the IUT. The null hypothesis of the IUT is that either b_1 or b_2 is zero, or both are zero, and the IUT rejects the null hypothesis if and only if all H_{0k} (H_{01}: b_1=0; H_{02}: b_2 =0) have been rejected. We did not consider a candidate regulator if its closest marker had a recombination rate of less than 0.25 with the marker closest to the target etrait, to prevent false discoveries due to strong cis-eQTLs, while KULP and JAGALUR (2006) excluded all candidate regulators on the same chromosome as the target.

We tested all candidate regulators on all etraits, retained the maximum p-value, corresponding to either b_1 or b_2 as the IUT procedure prescribes, and used the BH procedure for multiple testing adjustment.

3.2.5 Identification of regulator-target pairs for SPA, PCA and cis-mapping

We used local structural models to select regulator genes in each of the identified QTL CIs. The candidate regulator selection was performed in three steps: 1) Identification of the detected cis-linked etraits that were most likely cis-linked and those that were probably secondary (cistrans) effects, 2) Identification of the detected trans-affected etraits that were probably cistrans-affected and those that were more likely trans-affected, and 3) Search for the candidate regulator among the genes physically located in the eQTL confidence interval for each of the likely trans-affected etraits.

3.2.5.1 Distinguishing cis from cistrans

Some of the etraits that were found to be cis, based on the fact that they were affected by an eQTL whose CI overlapped with the physical location of the gene on the chromosome, may not be truly cis-affected. Such gene may be cistrans regulated through a cis-affected gene, or trans regulated by some coding region polymorphism in a gene located near the target gene. We tested whether a potentially cis-affected gene was likely cis-affected using model (3.1) but omitting the interaction term, letting y_{in} be the value of the potential cis-affected target etrait *i*, y_{jn} the value of another potential cis-affected regulator *j* of *i*, and x_{jn} the genotype of the marker at which the peak test statistic of the eQTL CI occurs. If y_i is actually cistrans-affected through y_j, then b_2 should not be significantly different from zero when y_j is included in the regression equation. These tests were carried out for all identified cis-affected etraits in an eQTL CI. If for an etrait *i,* b_2 remained significant for all etraits *j*, then it was identified as a “true” cis-affected etrait. Since the effects were already identified as significant in the eQTL analysis, no multiple testing adjustment was applied and a p-value cutoff of 0.05 was used.

3.2.5.2 Distinguishing trans from cistrans

After having identified the likely cis-affected etraits, we focused on the etraits detected as trans-affected by the eQTL CI. The trans-affected etraits can be either truly trans-affected or cistrans- affected through a cis-affected regulator. Using model (3.1) again, y_{in} is now a trans-affected etrait and y_{jn} is a cis-affected etrait identified in step 1. Cistrans regulation is indicated by b_2 not being significantly different from zero. If b_2 remains significant for all cis-affected etraits *j,* then the gene *i* is identified as a likely trans-affected etrait.

3.2.5.3 Selecting candidate trans-regulators in the same eQTL region

To find the candidate regulators for a likely trans-affected etrait *i* among all the genes physically located in the eQTL region, for the target etrait *i* we fitted model (3.1) with any candidate regulator etrait *j* located in the eQTL region and the eQTL marker (without the interaction term), and any additional candidate etrait *p*. The additional candidate etrait was included to examine whether the regulator-target correlation was due to some indirect mechanism. For each candidate-target pair *i* & *j*, the null hypothesis is that at least one of the *b* coefficients of *j* is not significantly different from zero after having a different candidate etrait *p* in the model. Therefore, we retained the maximum p value of all the *b* coefficients of each candidate regulator of each target as in WILLE and BUHLMANN (2006) and WILLE *et al.* (2004). We used a p-value cutoff of 0.05/number of candidate regulators to control the family-wise error rate at 0.05 for all tests performed for each eQTL-target pair. A candidate regulator etrait with all of its *b* coefficients significantly different from zero was retained as a regulator of its target.

3.2.6 Identification of regulator-target pairs for trans-mapping

In the trans-eQTL mapping analysis based on model (3.1), some of the identified regulator genes may be false positives, because the regulator etrait might be significant due to some correlating mechanisms other than the hypothesized direct causal relationship between the regulator and the target, and/or the marker might be significant due to the linkage with another polymorphism. For example, regulator *j* might directly affect another regulator *k*, which in turn affects target *i*, without direct causal relationship between *j* and *i*. To eliminate such cases, we performed some "local sparsification" for each target etrait. For each target etrait *i* with at least two identified regulators, for each identified regulator *j* of etrait *i*, we included another regulator etrait and its nearest marker (gene *k*) in the regression model:

$$\begin{aligned} y_{in} = &\left(b_{1j} y_{jn} + b_{2j} x_{jn} + b_{3j} y_{jn} x_{jn}\right) + \\ &\left(b_{1k} y_{kn} + b_{2k} x_{kn} + b_{3k} y_{kn} x_{kn}\right) + \varepsilon_{in} \end{aligned} \quad (3.2)$$

Gene *k* is another regulator of etrait *i* identified as significant in the trans-mapping step, and the marker closest to it was required to have a recombination rate of at least 0.25 with the marker closest to gene *j*. If the recombination rate was less than 0.25 between the two markers, only one marker (x_{jn}) was kept in the model. The interaction terms were included only if they were significant in the analysis based on model (3.1). The terms for *i* and *j* are the same as in model (3.1), with additional etrait, marker and interaction terms for gene *k*. Since the candidate regulators included in the model were already identified as significant in the trans-mapping, no multiple testing adjustment was performed and the IUT was applied at a p-value cutoff of 0.05. If the IUT for regulator *j* was not significant, we discarded gene *j* as a regulator.

3.2.7 EDN construction

The EDN consists of two types of nodes: continuous nodes for the genes, and discrete nodes for the eQTLs. Edges in the EDN correspond to causal influences between these nodes. To construct an encompassing network, we simply assembled all the identified and retained regulator-target relationships, which consist of directed edges from eQTLs to cis-regulated target genes, from cis-regulated genes to cistrans regulated target genes, from trans-regulator genes to target genes and from trans-eQTLs to target genes.

3.3 RESULTS

3.3.1 Single Profile Analysis

With a 5% FDR p-value threshold of 0.000264, a total of 666 significant combined eQTL regions and 6,264 individual eQTL-target pairs were detected. The sizes of the eQTL CI regions were relatively wide (median 84 kb), which in some cases can be due to multiple linked QTL. The median size of the eQTL CI regions from three-marker regression decreased to 43 kb, the number of eQTL regions increased to 797, and the number of significant eQTL-target pairs increased to 6,729.

3.3.2 Principal Components Mapping

First, PCA was performed on all 4,589 etraits. Based on the scree plot which shows the fraction of total variance in the data as explained by each PC, 20 PCs were selected for eQTL mapping. The plot of the sorted gene loadings of the first 10 PCs (Figure 3.1) shows that many genes contribute to each PC. Therefore, PC mapping based on PCA of all genes was only able to detect major eQTL affecting a relatively large number of genes. With PCA on all 4,589 filtered genes, 38 combined eQTL regions were detected (median CI 84 kb), including all major eQTL regions affecting relatively large number of genes identified earlier with analyses on single profiles or clustered profiles (YVERT *et al.* 2003).

When analyzing PCs computed from separate PCA of the gene subsets corresponding to the 100 clusters, after three-marker regression, a total of 250 combined eQTL regions (median CI 37 kb) were detected. The eQTL regions detected with PCA on all etraits were also detected here. Next, SPA was performed on these 250 eQTLs, with a FDR adjusted p-value cutoff of 0.00012. A total of 10,316 eQTL-target pairs were detected.

3.3.3 Cis-eQTL Mapping

For cis-mapping, as expected, controlling FDR at the 5% level resulted in a considerably less stringent p-value threshold (0.0139), compared with the SPA threshold. After three-marker regression, a total of 578 combined cis-eQTL regions (median CI 36 kb) were detected. We then searched for cistrans-affected etraits of these eQTLs. The FDR-adjusted p-value cutoff at this stage was 0.000412. A total of 7,481 eQTL-target pairs were found.

3.3.4 Trans-eQTL Mapping

Trans-mapping appeared to greatly increase the power to detect eQTL. Using the IUT with the 5% level FDR control, 41,309 significant candidate regulator-target pairs were identified. Figure 3.2 presents representative profiles of an etrait on two chromosomes. Red lines represent the SPA profile and its threshold, and blue lines represent the trans-mapping profile and its threshold. The trans-mapping profile is raised considerably above the SPA profile and this increase more than compensates for an increase in the threshold value.

The interactions between eQTLs and candidate regulator genes did not appear to be important. Out of all tests performed, only 0.08% tests had significant eQTL by regulator gene interactions with FDR control at the 5% level for this term. Out of the tests with significant IUT, 4.94% had p-values smaller than 0.01, and 0.43% had p-values smaller than the FDR cutoff from all tests.

3.3.5 Comparing the eQTL detection power

A wide eQTL region detected by one mapping method may correspond to two eQTL regions detected by another method. Therefore, we compared the eQTL mapping methods by counting the number of overlapping eQTLs. More precisely, we counted the overlap of any two eQTLs detected with two different methods as affecting the same etrait. We considered any overlap, 50% overlap, and 99% overlap of the eQTL regions. For comparing trans-mapping results with other methods, an overlap or agreement was counted when the trans-eQTL marker was inside of an eQTL region detected by the other methods, or located immediately next to it with no other marker in between.

The percentages of eQTLs that had any overlap between different methods are shown in Figure 3.3. The percentages were not very different when a 50% or 99% overlap between eQTL regions was required (results not shown).

Most eQTL-target pairs detected in SPA (83%) overlapped with eQTLs identified in cis-mapping. These effects were likely either cis or cistrans. Another 13% were found in PC-mapping, and these eQTLs probably were pleiotropic eQTLs. Only 31% of the SPA eQTL regions contained or were next to markers identified by trans-mapping, an expected finding given that the SPA results included many cis- and cistrans linkages. Only 3% of the SPA eQTL-target pairs were not detected by the other methods.

For cis-mapping, 9% of the eQTL-target pairs were not found in SPA. These probably were the less significant cis-eQTLs, which have relatively fewer cistrans-affected etraits. For PC-mapping, 24% of its eQTLs did not overlap with eQTLs from the other methods, indicating the strength of this pleiotropic approach.

Only 10% of the eQTLs from trans-mapping overlapped with the eQTLs from SPA. Of all trans-eQTLs, 87% did not overlap with the eQTLs from the other methods, which indicated the high power of trans-eQTL mapping, and supported the fact that the other methods mostly find cis and cistrans effects.

3.3.6 Regulator-target pair identification

A total of 6,723 eQTL-target pairs were detected using SPA. After searching for regulators of those eQTLs, 6,276 regulator-target pairs involving 3,050 genes were found, including 1,192 regulators and 2,544 targets. From PC-mapping, for the 10,316 eQTL-target pairs, 9,843 regulator-target pairs were retained, involving 3,581 genes, with 1,143 regulators and 3, 262 targets. A total of 7,481 eQTL-target pairs were found by cis-mapping, and after search for regulators, 6,090 regulator-target pairs involving 3,034 genes were found, including 1,099 regulators and 2,562 targets. After "local sparsification" of the trans-mapping results, the 41,309 candidate regulator-target pairs reduced to 15,835 pairs involving 3,858 genes, including 1,433 regulators and 3,682 targets.

The percentages of common regulator-target pairs between different methods are shown in Figure 3.4. The combined results from the SPA and cis-mapping included 57% of the SPA results. Most (95%) regulator-target pairs detected by the trans-mapping method is not detected by any other methods. That is because for the detection of trans regulations, the methods that did not include the regulator etraits in the model had limited power. The comparison based on the eQTL regions (Figure 3.3) showed that 87% of the eQTLs detected in the trans-mapping method did not fall within the eQTL regions detected by the other methods.

3.3.7 EDN construction

Since the combined eQTL mapping results from the PCA, cis and trans mapping detected most eQTLs found in SPA and much more, we constructed an EDN based on the combined results from PCA, cis and trans-mapping, which included 28,609 regulator-target pairs. The network consisted of 4,274 genes nodes. The remaining 315 genes did not receive any inputs nor were they affecting any other genes. A total of 2,118 genes were regulators of at least one target, among which 158 did not receive any inputs. A total of 4,116 genes were targets having at least one regulator, among which 2,156 did not affect any other genes in the network. A total of 1,960 genes occurred both as regulators and targets. There were 135 instances of reciprocal regulation present (i.e. gene A affects gene B and gene B affects gene A). Gene PHM7 had the most targets: 468, while gene YLR152C had the most regulators: 32.

The confirmed regulators or strong candidate regulator genes for the 13 eQTLs with widespread transcriptional effects identified in YVERT *et al.* (2003) were investigated in this EDN. Amn1, a confirmed regulator gene with widespread influence (YVERT *et al.* 2003), was found to be a top cistrans regulator with 408 cistrans targets. The strong candidate regulator MAK5 with five coding region polymorphisms between the two parental strains (YVERT *et al.* 2003) had 110 trans targets. Another confirmed regulator gene GPA1 (YVERT *et al.* 2003) had 60 targets, about half of them are trans-targets. The genes LEU2 and URA3 (auxotrophic markers deleted in one of the parental strains) (YVERT *et al.* 2003) had 98 (most were cistrans) and 32 (most were cistrans) targets in the cis-trans network, respectively. The heme-dependent transcription factor HAP1, which has a Ty insertion in one of the parental strains (BREM *et al.* 2002; YVERT *et al.* 2003), had 141 (100 cistrans, others were trans) targets.

The in- and out-degree distributions of the EDN are plotted in Figure 3.5. The out-degree distribution was approximately linear in the log scale, similar to scale free networks. The in-degree and overall degree distributions did not follow a power law as in scale free networks (BARABASI and ALBERT 1999).

3.3.8 Simulation results on regulator-target pair identification

We evaluated the ability of our local structural modeling approach to retain the correct regulator-target pairs within one eQTL region with the simulated data. For a population of 112 individuals (as in the yeast data), we simulated an eQTL region containing three eQTL causal polymorphisms and several candidates and targets. This local network is depicted in Figure 3.6. The target list for the eQTL region is T = [G2, G3, G4, G5, G6, G7, G8]. Gene G1 is the only trans-candidate regulator, while genes G3, G4, G6 and G7 are cis-candidate regulators. There are four types of regulations: one true trans-regulation (from G1 and Q1 to G2); two true cis-linkages (Q2 to G3 and Q3 to G6); two true cis-regulations of genes located in the eQTL region (Q2 to G3 to G4 and Q3 to G6 to G7); and two true cis-regulations of targets not located in the eQTL region (Q2 to G3 to G5 and Q3 to G6 to G8). Data were simulated with linear regression models with regression coefficients fixed at the value of 1 and residual standard deviations (SD) set to 0.125, 0.25 or 0.5 (one value for all genes, or for genes with odd numbers SD = 0.5 or 0.25, and for genes with even numbers SD = 0.125 or 0.25). For a gene directly regulated by an eQTL, the model is $y = bx + e = x + e$, where x is QTL genotype (0/1), variance due to the eQTL is equal to 0.25, and heritability = $0.25/(0.25+SD^2) = 0.941$, 0.80 or 0.50 for the three SD values, respectively. For a gene indirectly regulated by an eQTL, the model is $y = b(bx + e_1) + e_2 = x + e_1 + e_2$, and heritability = $0.25 / (0.25 + 2SD^2) = 0.889$, 0.667, and 0.333. Several scenarios were considered with different values for the recombination rate and SD. A total of 1000 data replicates were simulated and analyzed for each scenario. The results are summarized in Table 3.1 in terms of power and false positive rate for the four types of regulations described above, which demonstrate that the procedure works well, with the exception of a case where some genes have extremely high and other genes low heritability (column 5 in Table 3.1). This problem was actually due to one of the cis-linked genes (G3) having very small residual variance and being assigned as a regulator for other genes incorrectly.

3.4 DISCUSSION

Several different methods for eQTL mapping and regulator gene selection were evaluated in terms of their ability to construct an EDN for gene network inference. The combined eQTL mapping results from the PCA, cis and trans mapping detected most eQTLs found in SPA and much more, and therefore should be a good starting point for EDN construction. The PC-mapping based on PCA of all genes identified only major eQTLs with widespread effects, while the PC-mapping based on the separate PCA of gene clusters detected many eQTL-target pairs that were not detected by the other methods. Cis and trans-mapping greatly increased power for eQTL detection as compared to SPA. Cis-mapping detected much more cis regulations due to a much less stringent p-value cutoff after the FDR control. The power for detecting cistrans effects was also somewhat increased. By including the candidate regulator genes in the model, the trans-mapping method detected many eQTL-target pairs, more than six times the number found with SPA, and more than four times the number found with PC-mapping. Of the eQTLs detected by trans-mapping, 87% were not detected by the other methods. However, the number of regulator-target pairs was only increased by a factor of less than four over SPA and two over PC-mapping. For trans-mapping, the number of regulator-target pairs was less than half the number of identified eQTL-target pairs, indicating that many of these effects were indirect or more distant and were identified due to the greater sensitivity of this method.

Since PC-mapping exhibited very high power, combining PC-mapping with cis-mapping and trans-mapping to detect pleiotropic eQTLs should be explored. Combining cis-mapping and trans-mapping of individual etraits and PCs may be a better approach for EDN construction, which can be further improved by the incorporation of “external” biological information such as SNP presence in candidate regulators (LI *et al.* 2005) or information on protein-protein interactions as recently proposed (TU *et al.* 2006).

We do not think that our use of the BH procedure for multiple testing control had a major impact on our results, but further research on efficient and effective multiple testing control for eQTL mapping is still desirable. While various authors have used the FDR criterion for multiple testing control in genome-wide QTL mapping, there are convincing arguments that FDR is not the best criterion to use in this context. It should be valid to use FDR control for the p-values of the peak statistics on all chromosomes over all etraits, however this approach misses multiple eQTLs on the same chromosome. One possibility is to extend this approach to retaining the peak statistics in each of several (equal) chromosome partitions. STOREY *et al.* (2005) presented a sequential method retaining two most important eQTLs for each etrait. However, identifying only one or two eQTLs per etrait may miss information important to gene network reconstruction. Chen and Storey (2006) proposed a different criterion for multiple testing but relied on data permutation for its implementation, which is computationally very demanding when based on common quantile (as recommended, e.g. (CARLBORG *et al.* 2005)) rather than common cut-off.

Sample size calculations (via simulation) should be performed for genetical genomics experiments with the most efficient methods for eQTL mapping and regulator-target pair identification, as in KIM *et al.* (2005), to ensure sufficient power while containing the large expense of these experiments.

An EDN constructed as described in this study still contains many direct regulations that are actually indirect, as well as multiple candidate regulators for some eQTLs and targets. It is therefore important to perform further sparsification of the network by a search within the (constrained) network space defined by the EDN. Bayesian network analysis has been used for this purpose (LI *et al.* 2005; ZHU *et al.* 2004), although it does not permit the reconstruction of networks with cycles. In another contribution, we therefore report on the use of Structural Equation Modeling (SEM) to reconstruct cyclic networks based on a genetical genomics experiment (LIU *et al.* 2006).

3.5 APPENDIX: DATA PREPROCESSING

The data set used in this study is from a yeast segregating population of 112 segregants (BREM and KRUGLYAK 2005). This yeast experiment creates natural, multi-factorial perturbation in a segregating population by crossing two strains of yeast – haploid derivatives of a standard laboratory strain (BY) and a wild isolate (RM). The population is genotyped at 2,956 genetic markers and genome wide expression profiled.

From the raw data downloaded from the Gene Expression Omnibus database, we performed background subtraction (foreground – background), calculated the mean of each array and dye, and used the mean to replace the null data. The resulting data were then lowess transformed using the MAANOVA package (http://www.jax.org/staff/churchill/labsite/software/anova/index.html).

The average sample/reference log ratios of the technical replicates were used as data for the analysis. The data were then normalized by subtracting the mean of each sample. From the normalized data, we removed the 496 ORFs rejected by Kellis *et al.* (2003) and another 4 ORFs marked as "deleted" in the GEO database. Finally, we filtered 20% genes with low variance from the data. The resulted data set has expression profiles (etraits) for 4,589 genes and 2,956 genetic marker genotypes for 112 samples.

AUTHORS' CONTRIBUTIONS

Bing Liu carried out the analysis for SPA, cis-mapping, PC-mapping, comparison of the results and drafted the manuscript. Dr. Alberto de la Fuente carried out the analysis on the trans-mapping, regulator-target pair identification, EDN construction, and a simulation study for the regulator-target pair identification. Dr. Ina Hoeschele directed the work.

REFERENCES

BARABASI, A., and R. ALBERT, 1999 Emergence of scaling in random networks. Science **286:** 509-512.

BENJAMINI, Y., and Y. HOCHBERG, 1995 Controlling the false discovery rate—a practical and powerful approach to multiple testing. J. Roy. Stat. Soc. B **57:** 289-300.

BING, N., and I. HOESCHELE, 2005 Genetical genomics analysis of a yeast segregant population for transcription network inference. Genetics **170:** 533-542.

BOOMSMA, D., 1996 Using multivariate genetic modeling to detect pleiotropic quantitative trait loci. Behavior Genet. **26:** 161-166.

BRAZHNIK, P., A. DE LA FUENTE and P. MENDES, 2002 Gene networks: how to put the function in genomics. Trends Biotechnol. **20:** 467-472.

BREM, R. B., and L. KRUGLYAK, 2005 The landscape of genetic complexity across 5,700 gene expression traits in yeast. Proc. Natl. Acad. Sci. USA **102:** 1572-1577.

BREM, R. B., G. YVERT, R. CLINTON and L. KRUGLYAK, 2002 Genetic dissection of transcriptional regulation in budding yeast. Science **296:** 752-755.

CARLBORG, O., D. J. DEKONING, K. F. MANLY, E. CHESLER, R. W. WILLIAMS *et al.*, 2005 Methodological aspects of the genetic dissection of gene expression. Bioinformatics **21:** 2383-2393.

CARPENTER, A. E., and D. M. SABATINI, 2004 Systematic genome-wide screens of gene function. Nat. Rev. Genet. **5:** 11-22.

CASELLA, G., and R. L. BERGER, 1990 *Statistical inference*. Wadsworth, Pacific Grove, CA.

CHASE, K., D. R. CARRIER, F. R. ADLER, T. JARVIK, E. A. OSTRANDER *et al.*, 2002 Genetic basis for systems of skeletal quantitative traits: Principal component analysis of the canid skeleton. Proc. Natl. Acad. Sci. **99:** 9930-9935.

CHEN, L., and J. D. STOREY, 2006 Relaxed Significance Criteria for Linkage Analysis. Genetics **173:** 2371-2381.

COMUZZIE, A. G., M. C. MAHANEY, L. ALMASY, T. D. DYER and J. BLANGERO, 1997 Exploiting pleiotropy to map genes for oligogenic phenotypes using extended pedigree data. Genet. Epidemiol. **14:** 975-980.

D'HAESELEER, P., S. LIANG and R. SOMOGYI, 2000 Genetic network inference: from co-expression clustering to reverse engineering. Bioinformatics **16:** 707-726.

D'HAESELEER, P., X. WEN, S. FUHRMAN and R. SOMOGYI, 1999 Linear modeling of mRNA expression levels during CNS development and injury. Pac. Symp. Biocomputing **4:** 41-52.

DE JONG, H., 2002 Modeling and simulation of genetic regulatory systems: a literature review. J. Comput. Biol. **9:** 67-103.

DE LA FUENTE, A., N. BING, I. HOESCHELE and P. MENDES, 2004 Discovery of meaningful associations in genomic data using partial correlation coefficients. Bioinformatics **20:** 3565-3574.

DE LA FUENTE, A., P. BRAZHNIK and P. MENDES, 2001 A quantitative method for reverse engineering gene networks from microarray experiments using regulatory strengths., pp. 213-221 in *2nd Int. Conf. Syst. Biol.*, edited by T. M. YI, M. HUCKA, M. MOROHASHI and H. KITANO. Omnipress, California Institute of Technology, Pasadena, CA.

DE LA FUENTE, A., P. BRAZHNIK and P. MENDES, 2002 Linking the genes: inferring quantitative gene networks from microarray data. Trends Genet. **18:** 395-398.

DI BERNARDO, D., M. J. THOMPSON, T. S. GARDNER, S. E. CHOBOT, E. L. EASTWOOD *et al.*, 2005 Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. Nat. Biotechnol. **23:** 377-383.

DOSS, S., E. E. SCHADT, T. A. DRAKE and A. J. LUSIS, 2005 Cis-acting expression quantitative trait loci in mice. Genome Res. **15:** 681-691.

FRIEDMAN, N., 2004 Inferring cellular networks using probabilistic graphical models. Science **303:** 799-805.

FRIEDMAN, N., M. LINIAL, I. NACHMAN and D. PE'ER, 2000 Using Bayesian networks to analyze expression data. J. Comp. Biol. **7:** 601-620.

GARDNER, T., D. DI BERNARDO, D. LORENZ and J. COLLINS, 2003 Inferring genetic networks and identifying compound mode of action via expression profiling. Science **301:** 102-105.

GARDNER, T. S., and J. FAITH, 2005 Reverse-engineering transcription control networks. Phys. Life Rev. **2:** 65-88.

HUGHES, T. R., M. J. MARTON, A. R. JONES, C. J. ROBERTS, R. STOUGHTON *et al.*, 2000 Functional discovery via a compendium of expression profiles. Cell **102:** 109-126.

JANSEN, R. C., 2003 Studying complex biological systems using multifactorial perturbation. Nat. Revi. Gen. **4:** 145-151.

JANSEN, R. C., and J. P. NAP, 2001 Genetical genomics: the added value from segregation. Trends Genet. **17:** 388-391.

JANSEN, R. C., and J. P. NAP, 2004 Regulating gene expression: surprises still in store. Trends Genet. **20:** 223-225.

JIANG, C., and Z. B. ZENG, 1995 Multiple trait analysis of genetic mapping for quantitative trait loci. Genetics **140:** 1111-1127.

KACSER, H., and J. A. BURNS, 1981 The molecular basis of dominance. Genetics **97:** 639-666.

KELLIS, M., N. PATTERSON, M. ENDRIZZI, B. BIRREN and E. S. LANDER, 2003 Sequencing and comparison of yeast species to identify genes and regulatory elements. Nature **423:** 241-254.

KIM, H. Y., J. M. WILLIAMSON and C. M. LYLES, 2005 Sample-size calculations for studies with correlated ordinal outcomes. Stat. Med. **24:** 2977-2987.

KULP, D., and M. JAGALUR, 2006 Causal inference of regulator-target pairs by gene mapping of expression phenotypes. BMC Genomics **7:** 125.

LAN, H., J. P. STOEHR, S. T. NADLER, K. L. SCHUELER, B. S. YANDELL *et al.*, 2003 Dimension reduction for mapping mRNA abundance as quantitative traits. Genetics **164:** 1607-1614.

LANDER, E., and D. BOTSTEIN, 1989 Mapping mendelian factors underlying quantitative traits using RFLP linkage maps. Genetics **121:** 185-199.

LEHMANN, E., 1975 *Nonparametrics: Statistical Methods Based on Ranks*. Holden-Day, Inc, San Francisco.

LI, H., L. LU, K. MANLY, E. CHESLER, L. BAO *et al.*, 2005 Inferring gene transcriptional modulatory relations: a genetical genomics approach. Hum. Mol. Genet. **14:** 1119-1125.

LIU, B., A. DE LA FUENTE and I. HOESCHELE, 2006 Gene network inference via structural equation modeling in genetical genomics experiments

LIU, J., J. M. MERCER, L. F. STAM, G. C. GIBSON, Z. B. ZENG *et al.*, 1996 Genetic analysis of a morphological shape difference in the male genitalia of *Drosophila simulans* and *D. mauritiana*. Genetics **142:** 1129-1145.

MANGIN, B., B. GOFFINET and A. REBAI, 1994 Constructing confidence intervals for QTL location. Genetics **138:** 1301-1308.

MANGIN, B., P. THOQUET and N. H. GRIMSLEY, 1998 Pleiotropic QTL analysis. Biometrics **54:** 88-99.

OMHOLT, S. W., E. PLAHTE, L. OYEHAUG and K. XIANG, 2000 Gene regulatory networks generating the phenomena of additivity, dominance and epistasis. Genetics **155:** 969-980.

REBAÏ, A., 1997 Comparison of methods of regression interval mapping in QTL analysis with non-normal traits. Genet. Res. **69:** 69-74.

RONALD, J., R. B. BREM, J. WHITTLE and L. KRUGLYAK, 2005 Local regulatory variation in *Saccharomyces cerevisiae*. PLoS Genet. **1:** e25.

ROY, S. N., 1957 *Some aspects of multivariate analysis*. Wiley, New York.

SACHS, K., O. PEREZ, D. PE'ER, D. A. LAUFFENBURGER and G. P. NOLAN, 2005 Causal protein-signaling networks derived from multiparameter single-cell data. Science **308:** 523-529.

SCHADT, E. E., S. A. MONKS, T. A. DRAKE, A. J. LUSIS, N. CHE *et al.*, 2003 Genetics of gene expression surveyed in maize, mouse and man. Nature **422:** 297-302.

STOREY, J., J. AKEY and L. KRUGLYAK, 2005 Multiple Locus Linkage Analysis of Genomewide Expression in Yeast. PLoS Biol. **3:** e267.

THALLER, G., and I. HOESCHELE, 2000 Fine-mapping of quantitative trait loci in half-sib families using current recombinations. Genet. Res. **76:** 87–104.

TU, Z., L. WANG, M. N. ARBEITMAN, T. CHEN and F. SUN, 2006 An integrative approach for causal gene identification and gene regulatory pathway inference. Bioinformatics **22:** e489-496.

VAN SOMEREN, E. P., L. F. WESSELS and M. J. REINDERS, 2000 Linear modeling of genetic networks from experimental data, pp. 355-366 in *Proc. Int. Conf. Intell. Syst. Mol. Biol.*

WAGNER, A., 2001 How to reconstruct a large genetic network from n gene perturbations in fewer than n(2) easy steps. Bioinformatics **17:** 1183-1197.

WELLER, J. I., G. R. WIGGANS, P. M. VANRADEN and M. RON, 1996 Application of a canonical transformation to detection of quantitative trait loci with the aid of genetic markers in a multi-trait experiment. Theor. Appl. Genet. **92:** 998.

WILLE, A., and P. BUHLMANN, 2006 Low-order conditional independence graphs for inferring genetic networks. Stat. Appl. Genet. Mol. Biol. **5:** Article 1.

WILLE, A., P. ZIMMERMANN, E. VRANOVA, A. FURHOLZ, O. LAULE *et al.*, 2004 Sparse graphical Gaussian modeling of the isoprenoid gene network in Arabidopsis thaliana. Genome Biol. **5:** R92.

YVERT, G., R. BREM, J. WHITTLE, J. AKEY, E. FOSS *et al.*, 2003 Trans-acting regulatory variation in *Saccharomyces cerevisiae* and the role of transcription factors. Nature Genet. **35:** 57-64.

ZENG, Z.-B., J. LIU, L. F. STAM, C.-H. KAO, J. M. MERCER *et al.*, 2000 Genetic architecture of a morphological shape difference between two *Drosophila* species. Genetics **154:** 299-310.

ZENG, Z., 1993 Theoretical basis for separation of multiple linked gene effects in mapping quantitative trait loci. Proc. Nat. Acad. Sci. **90:** 10972-10976.

ZHU, J., P. Y. LUM, J. LAMB, D. GUHATHAKURTA, S. W. EDWARDS *et al.*, 2004 An integrative genomics approach to the reconstruction of gene networks in segregating populations. Cytogenet. Genome Res. **105:** 363-374.

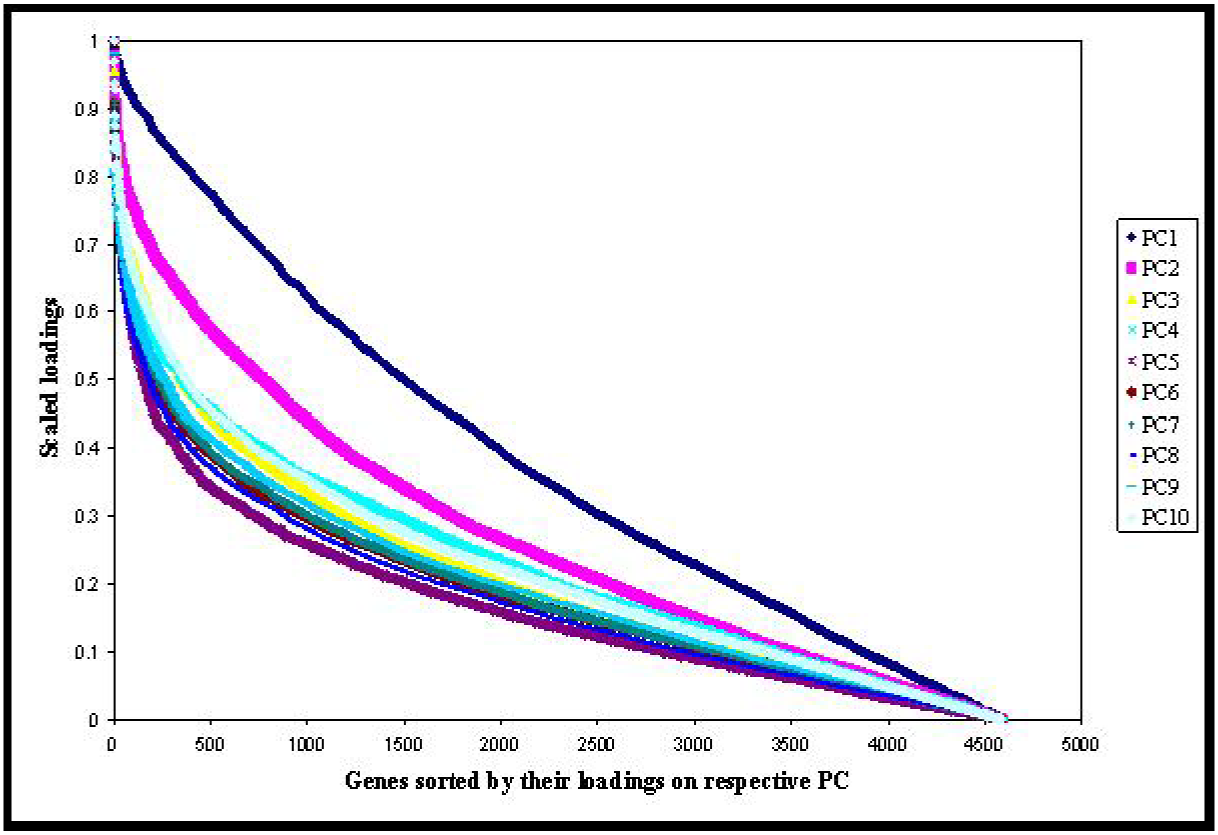


FIGURE 3.1.— Sorted gene loadings of the first 10 PCs of the filtered genes

For each of the first ten PCs of the filtered genes, the genes were sorted based on their loading on the PC. The loadings were scaled by been divided by the maximum loading for that PC.

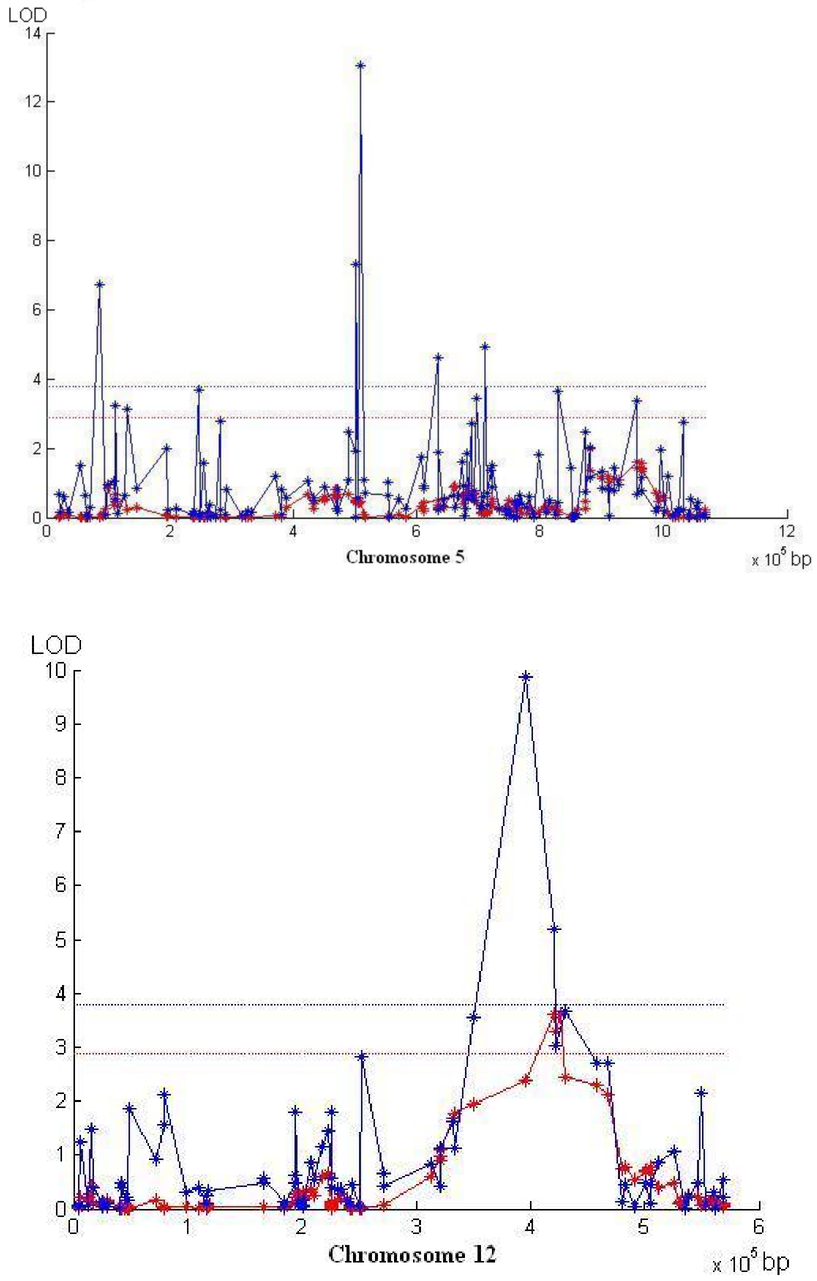


FIGURE 3.2.— Test statistics profiles of an etrait on two chromosomes

Red lines: SPA profile and its threshold. Blue lines: Trans-mapping profile and its threshold. Approximate LOD test statistics were computed by dividing the test statistics by 2*ln(10).

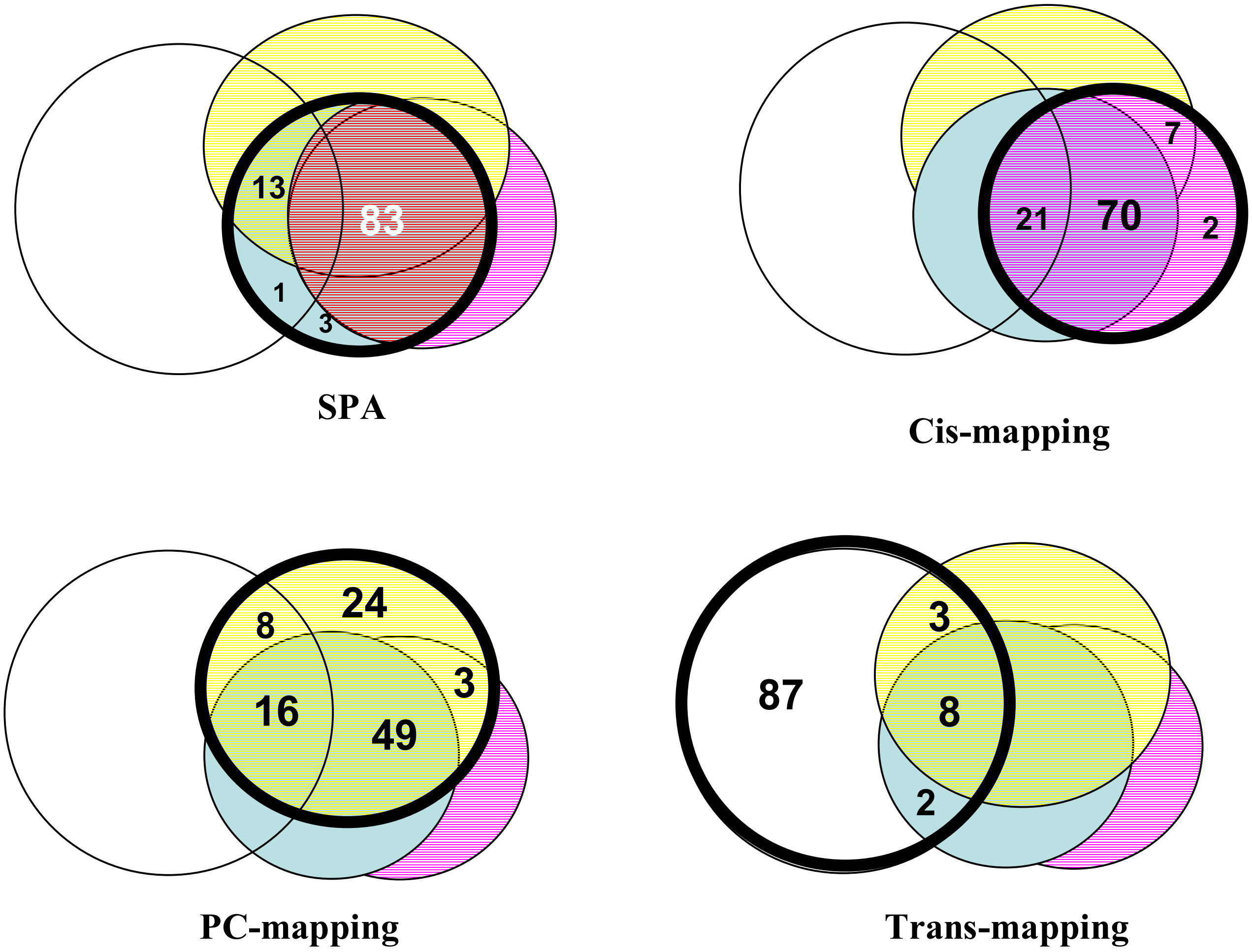
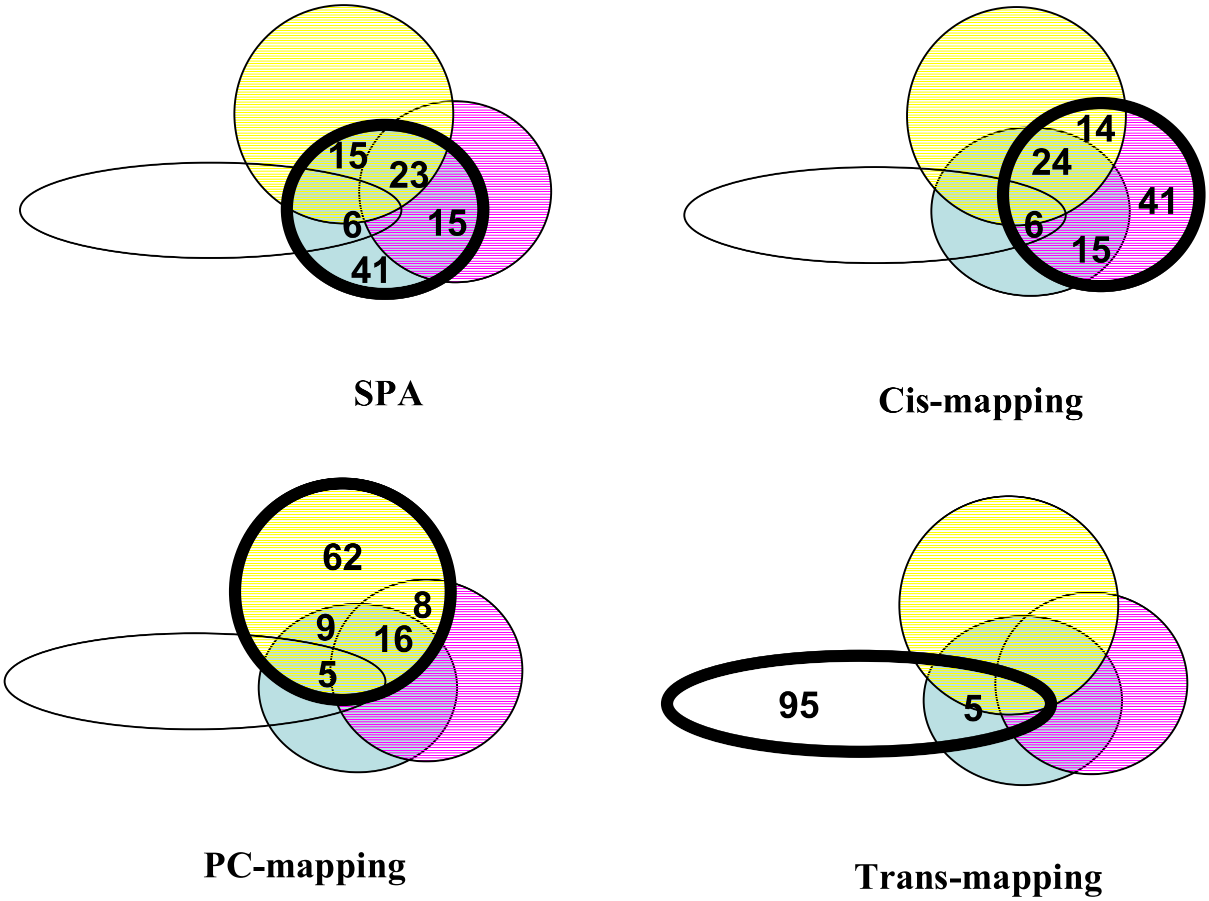


FIGURE 3.3.— Comparison of detected eQTLs of SPA, PC-mapping, cis-mapping and trans-mapping

The percentages of the overlapping eQTLs detected with different methods as affecting the same etrait. Two eQTL regions are regarded overlapping if their regions have any overlap on the chromosome. For comparing trans-mapping results with the other methods, an overlap was counted when the trans-eQTL marker was inside of an eQTL region detected by the other methods, or located immediately next to it with no other marker in between.



SPA

Cis-mapping

PC-mapping

Trans-mapping

FIGURE 3.4.— Comparison of regulator-target pairs of SPA, PC-mapping, cis-mapping and trans-mapping

The percentages of common regulator-target pairs detected with different methods.

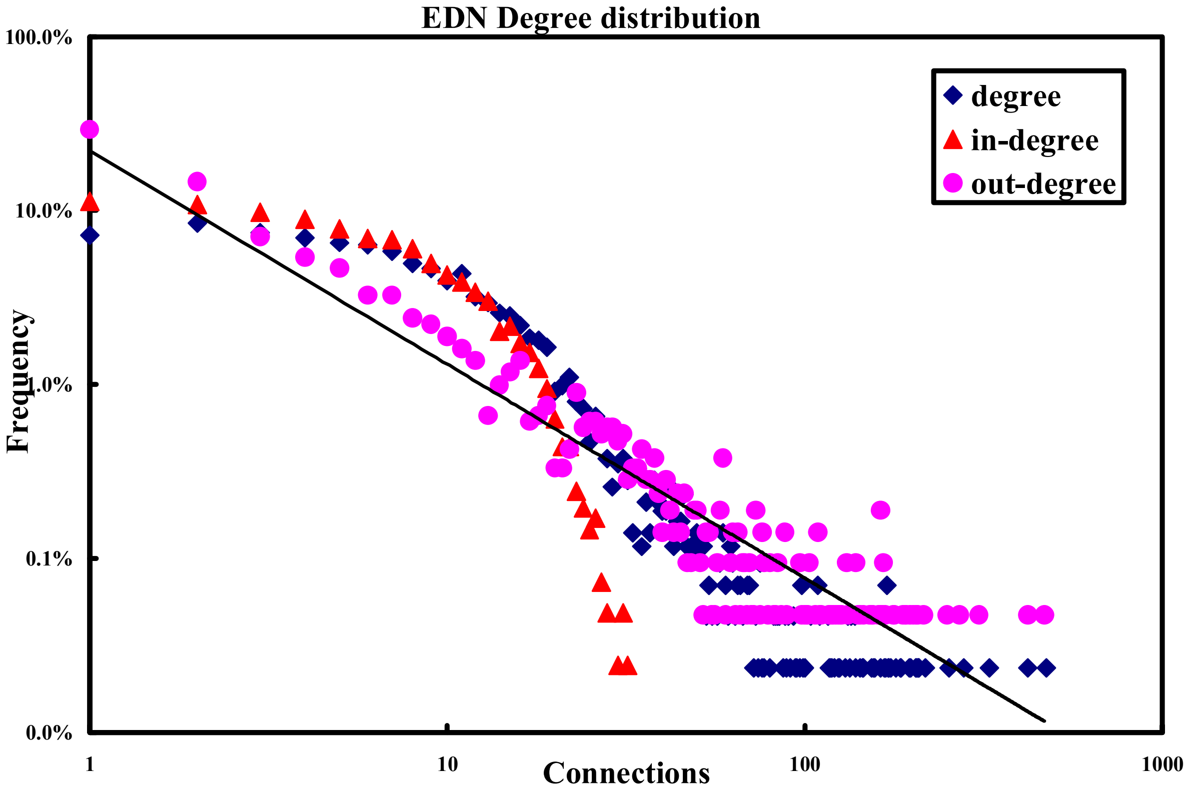


FIGURE 3.5.— Degree distributions of the EDN plotted in the log scale

Connections: the number of in (out/total) edges; Frequency: the number of genes having the corresponding number of in (out/total) connections divided by the number of genes having at least one in (out/total) connections. In-degree: the number of incoming edges; out-degree: the number of outgoing edges; degree: the total number of connections.

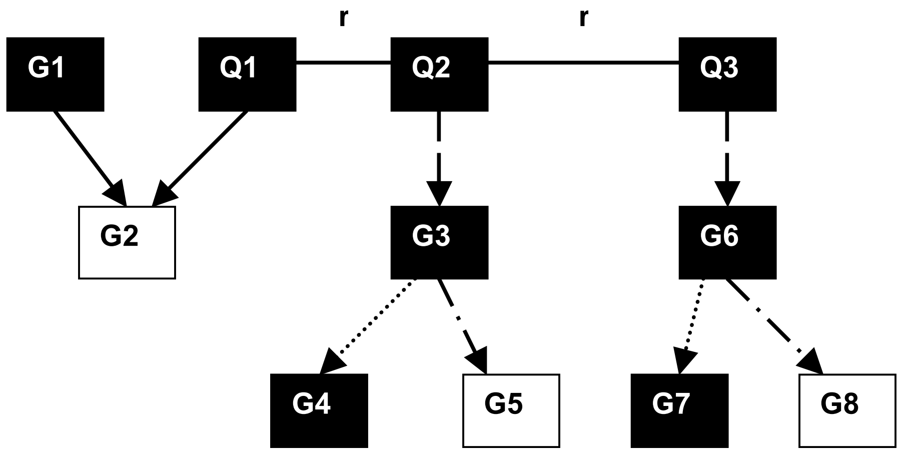


FIGURE 3.6.— The network model used in the simulation study

Black squares with starting letter Q: eQTLs. eQTL squares connected by lines: tightly linked eQTLs. Squares with starting letter G: genes. Black squares with starting letter G: genes located in the eQTL region. White squares with starting letter G: genes not located in the region but are affected by the eQTL. Solid arrows: true trans-regulation; dashed arrows: true cis-linkages; dotted arrows: true cistrans regulations on genes located in the eQTL region; dashed-dotted arrows: true cistrans regulations on targets not located in the eQTL region.

TABLE 3.1 Results from a simulation study on regulator-target pair identification

	SD=0.5	SD=0.25	SD=0.125	SD=0.5/0.125	SD=0.5/0.25	SD=0.25/0.125
Cis-link Power	100, 100	100, 100	100, 100	55.3, 59.85	89.4, 98.65	97.8, 98.5
Cis-link FDR	0.6, 0.9	0.7, 0.67	0.67, 0.57	0.48, 0.53	0.6, 0.72	0.62, 0.57
Cis-reg cis Power	99, 99	99, 99	99, 99	54.8, 59.4	88.6, 97.8	97.2, 97.8
Cis-reg cis FDR	0.35,0.13	0 , 0	0, 0	38.6, 0.25	3.27, 0.15	1.8, 0.1
Cis-reg Power	99, 98.8	99, 98.9	98, 98.5	54.9, 59.2	88.2, 97.4	96.9, 97.5
Cis-reg FDR	0.93, 0.4	0, 0	0, 0	45, 25.82	4.82, 1.33	2.85, 1.3
Trans-reg Power	99, 99.2	100, 100	100, 100	41.8, 45.1	92.9, 96.1	96.3, 96.5
Trans-reg FDR	0.85, 1.1	1.1, 1.15	1.57, 1.52	26.95,62.52	10.92, 2.52	2.3, 2.77

Power: percentage of simulations in which the regulation type was found; FDR: percentage of simulations in which a regulation of a certain type was found that did not exist in the underlying network; Cis-link: regulation of target in eQTL region; Cis-reg: cis-regulation of target not in eQTL region; Trans-reg: trans-regulation. For the last three columns, even numbered gene nodes (Figure 3.6) receive the left amount of error variance and odd number nodes the right amount. The two numbers in each cell correspond to 0% recombination and 9% recombination (10 recombinants) among eQTLs, respectively. A p value cutoff of 0.01 was used.

Chapter 4

Gene network inference via structural equation modeling in genetical genomics experiments

Bing Liu[¶§], Alberto de la Fuente[§] and Ina Hoeschele[§¶]

[¶] Department of Statistics, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061

[§] Virginia Bioinformatics Institute (0477), Virginia Polytechnic Institute and State University, Blacksburg, VA 24061

ABSTRACT

In genetical genomics, a segregating population is expression profiled and DNA marker genotyped. An Encompassing Directed Network (EDN) of causal regulatory relationships among genes can be constructed with expression Quantitative Trait Locus (eQTL) mapping and selection of candidate causal regulators. An EDN is likely to contain cycles or feedback loops. In this work, we implement Structural Equation Modeling (SEM) to sparsify the EDN by producing a set of sub-models containing fewer edges and being well supported by the data. Typically, SEM has been implemented for only tens of variables. Based on a factorization of the likelihood and a strongly constrained search space, our algorithm can construct networks involving several hundred genes. Parameters are estimated based on the method of maximum likelihood, and structure inference is based on a penalized likelihood ratio and an adaptation of the Occam's Window model selection. The likelihood function is factorized into a product of conditional likelihoods of individual genes (not contained in a cycle) as in acyclic Bayesian Networks, and conditional likelihoods of subsets of genes that compose cyclic components. The likelihood of a cyclic component is maximized using genetic algorithms. The SEM algorithm was evaluated using simulated data having known underlying network topologies. For the simulated networks, the SEM approach had an average detection power of around ninety percent, and an average false discovery rate of ten percent. The algorithm was also applied to a sub-network of an EDN obtained from a yeast data set. Our implementation of SEM permits the reconstruction of networks of several hundred genes, and future research will likely improve upon the efficiency of the current implementation.

4.1 INTRODUCTION

System biologists are interested in understanding how DNA, RNA, proteins and metabolites work together as a complex functional network. The gene network is a projection of such network on the gene space (BRAZHNIK *et al.* 2002), in the sense that only relationships between genes are modeled, while the physical interactions between them may be acted through other components. While networks including genes, RNA, proteins and metabolites would be more informative, gene networks are system level descriptions of cellular physiology and provide an understanding of the genetic architecture of complex traits (e.g. complex diseases).

Bayesian Networks are currently a popular tool for gene network inference (FRIEDMAN *et al.* 2000; HARTEMINK *et al.* 2002; IMOTO *et al.* 2002; PE'ER *et al.* 2001; YOO *et al.* 2002). Bayesian networks use partially directed graphical models to represent conditional independence relationships among variables of interest and can describe complex stochastic processes. They are suitable for learning from noisy data (e.g. microarray data) (FRIEDMAN *et al.* 2000). Bayesian Networks are Directed Acyclic Graphical (DAG) models, which cannot represent structures with cyclic relationships. However, cyclic dependencies are ubiquitous in biology and are associated with many specific properties of living systems. Therefore, cyclic relationships are expected to be common in gene networks, which are hence better modeled as Directed Cyclic Graphs (DCGs). Based on the assumption that a cyclic graph represents a dynamic system at equilibrium (FISHER 1970), this problem can be theoretically resolved by including a time dimension, which produces causal graphs without cycles (DAG), which then could be studied using Bayesian Networks, an approach called Dynamic Bayesian Networks (HARTEMINK *et al.* 2002; MURPHY and MIAN 1999). However, such an approach requires the collection of time series data, which is difficult to accomplish, as it requires synchronization of cells and close time intervals not allowing for feedback (SPIRTES *et al.* 2000). Samples at wider time intervals represent near steady state data and hence require cyclic network reconstruction.

XIONG *et al.* (2004) were the first to apply Structural Equation Modeling (SEM) for gene network reconstruction using gene expression data. However, their application was limited to gene networks without cyclic relationships by using a recursive SEM, which has an acyclic structure and uncorrelated errors. These authors reconstructed only small networks with less than 20 genes. Here, we apply SEM in the context of genetical genomics experiments. In genetical genomics, a segregating population of hundreds of individuals is expression profiled and genotyped. An Encompassing Directed Network (EDN) of causal regulatory relationships among genes can be constructed with expression Quantitative Trait Locus (eQTL) mapping and selection of regulator-target pairs (LIU *et al.* 2006). In this study, we present an SEM implementation to search for a set of sparser structures within the EDN that are well supported by the data. The method is evaluated on the simulated data with known underlying network structure and on a real yeast data set. Typically, SEM analyses have included only tens of variables, but our implementation is capable of reconstructing networks of several hundred genes based on a factorization of the likelihood and a strongly constrained network topology search space.

4.2 METHODS

4.2.1 Encompassing Directed Network

Expression QTL mapping treats gene expression levels as quantitative traits, and identifies genomic regions causally affecting gene expression levels. It identifies a set of eQTL regions and for each eQTL a list of target genes whose expression profiles are affected. Furthermore, using DNA sequence information, genes located in an eQTL region can be identified as candidate regulators of the targets of that eQTL. Using local structural models, regulator-target pairs are identified for all eQTLs, taking into account that an eQTL may affect a target through cis, cistans or trans regulation. Then, an EDN is constructed by drawing directed edges from the regulator genes and eQTLs to the target genes. We have constructed an EDN using a genetical genomics dataset from yeast (LIU *et al.* 2006). Here, we implement the Structural Equation Modeling (SEM) to search within the EDN for a subset of sparser structures that are best supported by the data.

4.2.2 Structural Equation Modeling

4.2.2.1 A Structural Equation Model

SEM is widely used in econometrics, sociology and psychology, usually as a confirmatory procedure instead of an exploratory analysis for causal inference (e.g. BOLLEN 1989; JOHNSTON 1972; JUDGE *et al.* 1985). Shipley (2002) discussed the use of SEM in biology with an emphasis on causal inference. In general, an SEM consists of a structural model describing (causal) relationships among latent variables and a measurement model describing the relationships between the observed measurements and the underlying latent variables. A special case is the SEM with observed variables, where the variables in the structural model are directly observed, therefore there is no measurement model. Our model is a SEM with observed variables, which can be represented as

$$\boldsymbol{y}_i = \boldsymbol{B}\boldsymbol{y}_i + \boldsymbol{F}\boldsymbol{x}_i + \boldsymbol{e}_i\,;\;\; \boldsymbol{e}_i \sim (\boldsymbol{0}, \boldsymbol{E}) \qquad i = 1,...,\mathrm{N} \qquad (4.1)$$

In this model, for sample i (i = 1, . . . , N), $\boldsymbol{y}_i = (y_{i1},...,y_{ip})$` is the vector of expression values of all ($p$) genes in the network, and $\boldsymbol{x}_i = (x_{i1},...,x_{iq})$` denotes the vector of marker or QTL genotype codes. The $\boldsymbol{y}_i$ and $\boldsymbol{x}_i$ are deviations from means, $\boldsymbol{e}_i$ is a vector of error terms, and $\boldsymbol{E}$ is its covariance matrix.

Matrix $\boldsymbol{B}$ contains coefficients for the direct causal effects of the genes on each other. Matrix $\boldsymbol{F}$ contains coefficients for the direct causal effects of the eQTLs on the genes. The structure of matrices $\boldsymbol{B}$ and $\boldsymbol{F}$ corresponds to the path diagram or directed graph (in general a DCG) representing the structural model, in which vertices or nodes represent genes and eQTLs, and edges correspond to the non-zero elements of $\boldsymbol{B}$ and $\boldsymbol{F}$. Matrices $\boldsymbol{B}$ and $\boldsymbol{F}$ are sparse when the model represents a sparse network. When the elements in $\boldsymbol{e}$ are uncorrelated and matrix $\boldsymbol{B}$ can be rearranged as a lower triangular matrix, the model is recursive, there are no cyclic relationships, and the graph is a DAG. If the error terms are correlated ($\boldsymbol{E}$ is non-diagonal), or matrix $\boldsymbol{B}$ cannot be rearranged into a triangular matrix (indicating the presence of cycles or a DCG), the model is non-recursive.

The $\boldsymbol{x}_i$ may be fixed or random. In genetical genomics experiments, the $\boldsymbol{x}_i$ are random because individuals are sampled from a segregating population. However, the joint likelihood of the $\boldsymbol{y}_i$ and $\boldsymbol{x}_i$ can be factored into the conditional likelihood of the $\boldsymbol{y}_i$ given the $\boldsymbol{x}_i$ times the likelihood of the $\boldsymbol{x}_i$, and the latter does not depend on any of the network parameters in $\boldsymbol{B}$**,** $\boldsymbol{F}$ and $\boldsymbol{E}$*,* and therefore can be ignored. Thus, we only need to assume multivariate normality for the residual vectors.

An important issue in non-recursive SEM or DCG is equivalence. Models are equivalent when they cannot be distinguished in terms of overall fit. For DAGs, algorithms for checking the equivalence of two models or for finding the equivalence class of a given model in polynomial time are available (ANDERSSON *et al.* 1997; VERMA and PEARL 1991). Therefore, model search can be performed as a search among equivalence classes rather than among individual DAGs (CHICKERING 2002a). An equivalence class discovery algorithm for DCGs, which is polynomial time on sparse graphs (RICHARDSON 1996; RICHARDSON and SPIRTES 1999) is available, but there is no algorithm available for model search among equivalence classes as for DAGs. Two DAG models are equivalent if they have the same undirected edges but differ in the direction of some edges (edge reversal) (PEARL 2000). Two DCG model can be equivalent even if the differ in terms of undirected edges (RICHARDSON 1996; RICHARDSON and SPIRTES 1999). In our case, two models cannot be equivalent under edge reversal, because the directions of the edges are determined by the eQTLs. By using an information criterion for model selection (discussed below), if two equivalent models differ in the number of edges, we prefer the sparser model. Therefore, equivalence is of less concern in our case. Instead of selection among equivalence classes, we use a model search approach that identifies multiple models (discussed below).

4.2.2.2 Algorithms for likelihood maximization

A main concern about using SEM for gene network inference was about the limitations on the network size when using the existing SEM software (e.g. LISREL (Jöreskog and Sörbom 1989); Mx (Neale *et al.* 2003)) to perform SEM analysis. Typical applications of SEM models include only tens of variables. No existing software program can analyze models with a size relevant to genomics (hundreds or even thousands of variables). Even the SEM implementation of Xiong *et al.*(2004) which employed a genetic algorithm, was only applied to small networks of under 20 genes. Here, we implement SEM analysis in the context of genetical genomics, where the EDN provides a strongly constrained structure search space, allowing us to reconstruct networks of up to several hundred genes.

The most commonly used estimation method for SEM is the Maximum Likelihood (ML) method. Assuming a multivariate normal distribution of the residual vectors, or $\boldsymbol{e}_i$ ~ N (0, $\boldsymbol{E}$), the logarithm of the conditional likelihood of the $\boldsymbol{y}_i$'s given $\boldsymbol{x}_i$'s and given a particular structure is:

$$\begin{aligned} L(\boldsymbol{y_1},...,\boldsymbol{y}_N \mid \boldsymbol{B},\boldsymbol{F},\boldsymbol{E},\boldsymbol{x}_1,...,\boldsymbol{x}_N) &= \boldsymbol{\mathit{constant}} \\ + N\ln(|\boldsymbol{I}-\boldsymbol{B}|) + \frac{N}{2}\ln(|\boldsymbol{E}|^{-1}) - \frac{1}{2}\sum_{i=1}^{N}((\boldsymbol{I}-\boldsymbol{B})\boldsymbol{y}_i - \boldsymbol{F}\boldsymbol{x}_i)'\boldsymbol{E}^{-1}((\boldsymbol{I}-\boldsymbol{B})\mathrm{y}_i - \boldsymbol{F}\boldsymbol{x}_i) \end{aligned} \qquad (4.2)$$

This log likelihood is maximized with respect to the parameters in $\boldsymbol{B}$, $\boldsymbol{F}$ and $\boldsymbol{E}$.

Alternative models or structures were compared using information criteria. Information criteria combine the maximized likelihood with a penalty term to adjust for the number of free parameters, and some also adjust for sample size. The information criteria we investigated include the Bayesian Information Criterion (BIC) (Schwartz 1978) and a modification BIC(δ) (Broman and Speed 2002).

A non-recursive SEM model can be under-identified, while a recursive SEM is always identified. A model is "identified" if all parameters are independent functions of the data covariance matrix. Under regularity assumptions, an unidentified model can be equivalent to an identified model nested within it (BEKKER *et al.* 1994). Since we prefer the sparser model, our model selection based on an information criterion should arrive at identified models.

The likelihood function is non-linear in the parameters, and therefore an iterative optimization procedure is required for its maximization. With respect to the large number of parameters in an SEM for hundreds of genes, likelihood maximization is computationally very expensive or even infeasible. Fortunately, the likelihood can be factored into a product of local likelihoods which all depend on different sets of parameters, and which are maximized individually in analogy with Bayesian Network analysis. For directed acyclic graphs, the global directed Markov property permits the joint probability distribution of the variables to be factored according to the DAG (PEARL 2000). The factorization can be represented as p(V_1, V_2, ... V_n) $= \prod_{j=1}^{n}$ p(V_j | $\boldsymbol{V}$ (parents of *j*), $\boldsymbol{\theta}_j$), where $\boldsymbol{V}$ (parents of *j*) is a vector of *V*'s of the parent vertices of vertex *j*, and $\boldsymbol{\theta}_j$ is the parameter vector of the local likelihood f(V_j |.). A network with cyclic components (connected cycles, in which any gene can find a path back to itself through any other gene) becomes acyclic when a set of genes pertaining to the same cyclic component is collapsed into a single node, i.e. V_j represents either an individual gene or the set of genes involved in the same cyclic component. Then p(V_1, V_2, ... V_n) can be factored as above, thereby turning the optimization problem from one of thousands of dimensions into many of much smaller dimensions. For genes that are not involved in a cyclic component, the univariate conditional likelihood of a gene is maximized efficiently using linear regression. For the genes involved in a cyclic component, their joint multivariate conditional likelihood is maximized.

For a cyclic component, p(V_j | $\boldsymbol{V}$ (parents of j), $\boldsymbol{\theta}_j$) involves all equations having a gene in cyclic component j on the left hand side of Equation (4.1):

$$\boldsymbol{y}_{ic} = \boldsymbol{B}_{cp}\boldsymbol{y}_{icp} + \boldsymbol{F}_p\boldsymbol{x}_{ip} + \boldsymbol{e}_{ic}\,; \;\; \boldsymbol{e}_{ic} \sim (\boldsymbol{0}, \boldsymbol{E}_c) \qquad i = 1,\ldots,N \tag{4.3}$$

where $\mathbf{y}_{ic}$ is a vector with all genes in cyclic component j, $\boldsymbol{y}_{icp}$ is a vector with all genes in cyclic component j and all parents of genes in cyclic component j that are themselves not in cyclic component j; $\boldsymbol{B}_{cp}$ ($\boldsymbol{F}_p$) is obtained from the original $\boldsymbol{B}$ and $\boldsymbol{F}$ matrix by extracting all rows corresponding to the genes in j and all columns pertaining to parent effects of these genes, $\boldsymbol{x}_{ic}$ is all the QTL parents of j, and $\boldsymbol{e}_{ic}$ is the residual vector for all genes in j. The $\boldsymbol{B}_{cp}$ can be further partitioned into $\boldsymbol{B}_c$ and $\boldsymbol{B}_p$, corresponding to columns pertaining to genes in j and genes not in j, respectively, and $\boldsymbol{y}_{ip}$ includes the gene parents of j. Move the $\boldsymbol{B}_c$ matrix to the left,

$$(\boldsymbol{I} - \boldsymbol{B}_c)\boldsymbol{y}_{ic} = \boldsymbol{B}_p\boldsymbol{y}_{ip} + \boldsymbol{F}_p\boldsymbol{x}_{ip} + \boldsymbol{e}_{ic}\,; \;\; \boldsymbol{e}_{ic} \sim (\boldsymbol{0}, \boldsymbol{E}_c) \qquad i = 1,\ldots,N \tag{4.4}$$

In Equation (4.4), $\boldsymbol{y}_{ip}$ is a vector of exogenous variables (variables do not receive any inputs) just like $\boldsymbol{x}$. The likelihood function for this model is then

$$L(\boldsymbol{y}_{ic} \mid \boldsymbol{y}_{ip}, \boldsymbol{B}_c, \boldsymbol{B}_p, \boldsymbol{F}_p, \boldsymbol{E}_c, \boldsymbol{x}_{ip}) = \text{constant} + \text{N}\ln(|\boldsymbol{I} - \boldsymbol{B}_c|) + \frac{\text{N}}{2}\ln(|\boldsymbol{E}_c|^{-1})$$
$$-\frac{1}{2}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}((\boldsymbol{I} - \boldsymbol{B}_c)\boldsymbol{y}_{ic} - \boldsymbol{B}_p\boldsymbol{y}_{ip} - \boldsymbol{F}_p\boldsymbol{x}_{ip})'\boldsymbol{E}^{-1}((\boldsymbol{I} - \boldsymbol{B}_c)\boldsymbol{y}_{ic} - \boldsymbol{B}_p\boldsymbol{y}_{ip} - \boldsymbol{F}_p\boldsymbol{x}_{ip}) \quad (4.5)$$

The likelihood function of the genes in a cyclic component is maximized using a Genetic Algorithm (GA) based global optimization procedure. During the model search, local-likelihood *j* needs to be re-maximized with respect to $_j$ only if the set of parents of genes involved in the cyclic component has changed.

GA is a stochastic iterative optimization tool. It utilizes search and update techniques based upon principles of genetics, *e.g.* by means of selection, crossover and mutation (GOLDBERG 1989; HOLLAND 1975; HOLLAND 1992). We use GA with real number genome, and each parameter is coded as a real number "gene" located on a "chromosome" (a possible solution). GA creates many possible solutions in each population. New solutions (offspring) are generated by selection, crossover and mutation. The crossover-operator combines two chromosomes to produce an offspring. Mutation alters one or more genes in a chromosome. A scoring function is evaluated for each chromosome and used as a selection criterion for inclusion of that chromosome in the next generation's population. For the termination criterion, we require both a minimum number of generations to be reached, and the fitness score to converge.

GA finds a global or near-global optimum for high-dimensional problems. GA can search a very complex parameter space, and jump out of local optima. Though GA is computationally more expensive than the gradient based methods, it has been shown that GA is more successful for problems with very complex parameter spaces (MENDES 2001; MOLES *et al.* 2003).

In our model search algorithm, for re-maximization of the local likelihood of a cyclic component, we use four types of starting values simultaneously in the initial GA population: Random starting points; starting values obtained from Two Stage Least Squares (2SLS) (to be discussed below); starting values equal to the current parameter estimates; and starting values from the current parameter values for all genes except 2SLS estimates for the genes directly affected by the deletion or addition of an edge. We use current parameter values as starting values because we search the model space by removing and adding single or few edges at a time, and therefore most parameter estimates do not change or do not change much. However, the parameter values associated with the gene directly affected by the deletion or addition of an edge can change considerably and we hence initiated them by 2SLS. Using these starting values greatly increased the efficiency of the GA optimization. A GA C++ library GAlib (http://lancet.mit.edu/ga/) was used in our implementation.

GA evaluates the fit of a chromosome using the objective function, which in our case is the log likelihood function for genes in a cyclic component. With diagonal ***E*** matrix, the most computationally demanding part for evaluating the objective functions is the computation of the determinant of $(\boldsymbol{I-B})_c$. $(\boldsymbol{I-B})_c$ is a sparse matrix, and determinants are calculated using sparse LU decomposition as implemented in the C library UMFPACK, which applies the Unsymmetric MultiFrontal method for sparse LU factorization (DAVIS 2004a; DAVIS 2004b; DAVIS and DUFF 1997; DAVIS and DUFF 1999). Since the patterns of the matrices remain the same for a given structure, symbolic factorization is preformed only once and the result is used by all numerical factorizations for objective functions of that structure.

4.2.2.3 Starting values from two-stage Least-Squares

Two Stage Least Squares (2SLS; e.g, (GOLDBERGER 1991; JUDGE *et al.* 1985)) is a computationally efficient parameter estimation method for the SEM models. The 2SLS estimates are computed based on one portion of the model at a time, whereas the ML estimation takes the entire model into account. Therefore, ML is called a "full information" method, while 2SLS is a "partial information" method, and the ML estimates are generally better than the 2SLS estimates. However, since 2SLS is a non-iterative approach and computationally very efficient, we used it to generate starting values for the GA optimization of the cyclic components.

In 2SLS, the first step is to create predicted values of $\boldsymbol{y}$ using all of the exogenous variables in the system, i.e. solving the reduced form equations:

$$\boldsymbol{y_i} = (\boldsymbol{I} - \boldsymbol{B})^{-1}(\boldsymbol{F}\boldsymbol{x}_i + \boldsymbol{e}_i) = \boldsymbol{\Pi}\boldsymbol{x}_i + \boldsymbol{v}_i \tag{4.6}$$

Estimates of $\boldsymbol{\Pi}$ are obtained from this model by Ordinary Least Squares (OLS) and used to obtain predictions of $\boldsymbol{y}_i$ ($\hat{\boldsymbol{y}}_i$), which are then used in the original model, or

$$\boldsymbol{y_i} = \boldsymbol{B}\hat{\boldsymbol{y}}_i + \boldsymbol{F}\boldsymbol{x}_i + \boldsymbol{e}_i; \qquad i = 1,..., N \tag{4.7}$$

Estimates of $\boldsymbol{B}$ and $\boldsymbol{F}$ are then obtained by OLS. 2SLS may not work well for some genes with no suitable instrumental variables. An instrumental variable for prediction of an endogenous variable exists only under certain conditions in cyclic networks (e.g. HEISE

1975). These conditions are likely not met for all genes in a network. Only if each gene had a cis-linked QTL the conditions would always be met.

4.2.3 Network topology search

The EDN contains 2^d sub-models, where d is the number of edges. It is impossible to exhaustively search this space even for EDNs of moderate sizes. Therefore, we adapt a heuristic search strategy based on the principle of Occam's Window model selection (Madigan and Raftery 1994) which potentially selects multiple acceptable models. The search algorithm involves a down step and an up step. The down algorithm consists of the following steps:

0) Initialize set $\boldsymbol{A}$ = set of acceptable models as empty, set $\boldsymbol{C}$ = set of starting candidate models, and set $\boldsymbol{K}$ = set of models with minimum IC (the model selection criterion) as empty.
1) Select a model $\boldsymbol{M}$ from set $\boldsymbol{C}$, remove it from set $\boldsymbol{C}$ and add it to set $\boldsymbol{A}$. Let minIC=0.
2) Select a submodel $\boldsymbol{M}_0$ of $\boldsymbol{M}$ by removing an edge from $\boldsymbol{M}$.
3) Compute IC_{01}.
4) If $\mathrm{IC}_{01} < \mathrm{O}_t$ (some negative constant), remove $\boldsymbol{M}$ from set $\boldsymbol{A}$ and add $\boldsymbol{M}_0$ to set $\boldsymbol{C}$ if $\boldsymbol{M}_0 \notin \mathrm{C}$. Remove any model in set $\boldsymbol{K}$ and set minIC = -• (do not check for models with minimum IC anymore for this model).
5) If $\mathrm{O}_t < \mathrm{IC}_{01} <$ minIC, replace the model in set $\boldsymbol{K}$ with $\boldsymbol{M}_0$, and remove $\boldsymbol{M}$ from set $\boldsymbol{A}$.
6) If minIC $< \mathrm{IC}_{01} < 0$ and this model is chosen as a random start, remove $\boldsymbol{M}$ from set $\boldsymbol{A}$ and add $\boldsymbol{M}_0$ to set $\boldsymbol{C}$ if $\boldsymbol{M}_0 \notin \mathrm{C}$.

7) If there are more sub models of ***M***, go to 2. Otherwise, remove the model in set ***K*** and put it in set ***C*** if it is not already in set ***C.***
8) If ***C*** is not empty, go to 1.

Starting from all models accepted in the Down algorithm, the Up algorithm follows the same steps as in the Down algorithm, except every time an edge that was removed from the EDN is added back into the model. Once the Up algorithm is completed, the set A contains the set of potentially acceptable models.

For large networks with many removable edges, the original Occam's Window model selection (Madigan and Raftery 1994) approach may search a very large model space. In the worst case, it is equivalent to an exhaustive search. Therefore, we imposed a threshold O_t on the IC. Only if the IC of the sub-model strongly improved over the model it is nested in (IC smaller than the O_t), we kept the sub-model as a candidate. Otherwise, if no sub-model passed the threshold and the minimum IC was smaller than zero, we kept the model with minimum IC as a candidate model. The size of the search space depends on the value of O_t. If O_t= -•, the algorithm is similar to the Greedy Hill search. If -• <O_t < 0, then the algorithm searches a larger network space and possibly accepts multiple models. Because O_t requires that the sub-model strongly improves over the model it is nested in, it is likely that the search will accept only one final model. Therefore, we added some random start models in step 6 so that there may exist multiple search paths.

The model or structure search space is constrained to nested models within the EDN, and additionally, certain edges cannot be removed from the EDN, because their removal would contradict the results from the eQTL analysis. If a gene's expression profile is found to be influenced by an eQTL, then there must remain a direct or indirect path from the eQTL to that target gene in the network. For example, an edge for cis-regulation of a gene by an eQTL cannot be removed unless the eQTL has multiple cis-candidates, in which case one of the cis-edges needs to remain. Therefore, we identified those edges that cannot be removed without violating these path relations and fixed them in the model; they would not be removed during the model search. In our current implementation, we first sparsified the ***F*** matrix (eQTL → gene), and then the ***B*** matrix (gene → gene relations). Different approaches can be used for the structure update during the search. For example, multiple candidate regulators of the same eQTL may be tested first. Then, an eQTL and its candidate regulator(s) may be updated jointly. In addition, the eQTL analysis can suggest the sequence of edge deletion. For example, possible indirect effects may be tested first.

4.2.4 Data simulation

To evaluate the performance of linear SEM analysis on gene network inference, we simulated data with non-linear kinetic functions and cyclic topology in the context of genetical genomics experiments. We simulated QTL genotypes using the QTLcartographer software (Basten *et al.* 1996) and steady-state (equal synthesis and degradation rates and constant gene expression levels in time) gene expression profiles according to the simulated genotypes with the Gepasi software (Mendes 1993; Mendes 1997; Mendes *et al.* 2003) using a non-linear ordinary differential equation given by Equation (4.8):

$$\frac{dG_i}{dt} = V_i \cdot \prod_j \left(Z_j \left(\frac{K_{Ij}}{I_j + K_{Ij}} \right) \right) \times \prod_k \left(Z_k \left(1 + \frac{A_k}{A_k + K_{Ak}} \right) \right) - k_i G_i + \theta_i G_i \tag{4.8}$$

where G_i is mRNA concentration of gene i, V_i is its basal transcription rate, K_{Ij} and K_{Ak} are inhibition and activation rate constant, respectively. I_j and A_k are inhibitor and activator concentrations, respectively (the expression levels of genes in the network affecting the expression of gene i), and k_i is a degradation rate constant. Each gene has two genotypes, and the polymorphism is either located in its promoter region affecting its transcription rate (cis-linkage with V=1 for one genotype and V=0.75 for the other), or in the coding region of a regulatory gene changing the basal transcription rates of the target genes by multiplying V by a factor Z (Z=1 for one genotype and Z=0.75 for the other). Each gene has a 50% probability of having a promoter (cis) or coding region (trans) polymorphism. The error parameter θ_i represents the "biological" variance and was sampled from a normal distribution with a mean 0 and a standard deviation of 0.1 each time before the calculation of a steady state. All other parameters were set to 1. Lastly, we also added "experimental noise" to the generated data at 10% proportional to the variance of each gene's expression values. The parameters were chosen so that the estimated heritabilities were close to the real data. For a simulated data set, we calculated the heritabilities of the etraits by dividing the etrait variances from the data simulated without added biological and technical noise (i.e. variances came from genetical variance only) by the total variances of the etraits. The simulated etraits had an average heritability of 56%, and 60% of the etraits had heritabilities over 57%. The simulated etraits had somewhat lower heritabilities than the actual etraits in the yeast data set where 60% of the genes had estimated heritabilities > 69% (BREM and KRUGLYAK 2005). BREM and KRUGLYAK (2005) calculated heritabilities as (etrait variance in the segregants –pooled etrait variance among parental measurements)/ etrait variance in the segregants. The network topologies were generated as described by MENDES *et al.* (2003). For each generated network we created an EDN by adding links from each node *i* to node *j*, if node *j* was no more than two edges separated from node *i* in the true network. The results are reported as FDR and power using BIC (SCHWARTZ 1978) and BIC(δ) (Broman and Speed 2002) criteria.

4.3 RESULTS

The algorithm was tested on the simulated data and on a sub-network obtained from an EDN generated in LIU *et al.* (2006), using a real data set from a yeast segregating population (BREM and KRUGLYAK 2005).

4.3.1 Simulated data

Ten data sets with different random network topologies were analyzed. These networks had 100 genes, 100 eQTLs, and on average 148 gene → gene and 123 QTL → gene edges. Their EDN contained on average 360 gene → gene and 301 QTL → gene edges. On average 42 genes were involved in one to three cyclic components in each data set, with the biggest cyclic component involve on average 37 genes. The algorithm took around 24 hours for one data set. For these networks we used a very small O_t in the search, therefore only one final model was obtained. We report the results in terms of FDR and detection power. The FDR is expressed as the number of wrongly identified edges divided by the total number of identified edges. The power is defined as the number of edges correctly inferred as a fraction of the total number of edges in the true network. In Table 4.1, we compared results obtained using BIC with penalty term ln(N)*df, and BIC(δ) with penalty term d*ln(N)*df. We used the recommended d=2*LOD threshold / log10(N) (BROMAN and SPEED 2002), and an LOD cutoff of 3. The results showed that for the simulated data sets, BIC was not stringent enough for the

QTL edges, with an average power of 99% and an average FDR of 22%. For the gene edges, the average FDR was 8%, with some loss of power (average 88%). For the QTL edges, the average FDR with BIC(δ) was 9% while the average power was 99%. For the gene edges, with BIC(δ) the average FDR was only 1%, while the power was reduced to on average 78%. Overall, the algorithm had good performance and showed that the linear SEM approach seems to be robust under violation of the linearity assumptions.

We also tested one data set with 20 random start points, and sixteen very similar final models were obtained. Out of an average of 134 detected QTL → gene edges, average number of edges different from the best model was 4.4. Out of an average of 153 detected gene → gene edges, the average number of edges different from the best model was 7.9. The average BIC different from the best model was 26. The average absolute likelihood difference was 12, while the mean likelihood was 26,969. Two models had the same likelihood, while having six different eQTL → gene edges and seven different gene → gene edges. Another four sets of two models had likelihood difference smaller than one. They have on average four different eQTL → gene edges and on average 7.3 different gene → gene edges.

4.3.2 Yeast data analysis

We performed SEM analysis on a sub-network of an EDN obtained from the yeast dataset (LIU *et al.* 2006). To obtain this sub-network, we started out with 168 genes involved in a cycle component and included the genes connected to these genes by up to three edges, and all the eQTLs parents of these genes. The sub- network obtained had 265 genes, 241 QTLs, 832 gene → gene edges, and 640 QTL → gene edges. After sparsification using our SEM implementation, the resulted network contained 475 gene → gene edges and 468 QTL →

gene edges. Figure 4.1 shows the network topology of the network, with the dotted edges denoting the removed edges.

Table 4.2 shows the significant biological function groups of the genes in this network. About 41.6% of these genes are involved in catalytic activity, and another 18% are involved in hydrolase activity. All biological functions in Table 4.2 are significantly enriched in this network.

4.4 DISCUSSION

In this contribution, we present an initial evaluation of structural equation modeling for gene network reconstruction in the context of genetical genomics experiments. Previous investigations have used Bayesian networks (FRIEDMAN *et al.* 2000; HARTEMINK *et al.* 2002; IMOTO *et al.* 2002; PE'ER *et al.* 2001; YOO *et al.* 2002), but this methodology cannot reconstruct cyclic networks. Because cycles or feedback loops are expected to be common in genetic networks, it is imperative to investigate alternative methods such as the one we have presented here. Our implementation of SEM permits the reconstruction of networks of several hundred genes, and future research will likely improve upon the efficiency of the current implementation.

Maximum Likelihood is the predominant full-information method for parameter inference in structural equation models. It is therefore natural to perform a model (structure) search based on an information criterion that is a function of the maximized likelihoods of two competing models. While BIC and BIC(δ) performed satisfactorily in this study, further research into appropriate model selection criteria for large, very sparse networks is required. There is an interesting connection between classical model selection based on information criteria and bayesian model selection in the context of linear regression (CHIPMAN *et al.* 2001). Let $\boldsymbol{\gamma}$ be a vector of zero/one indicator variables (which defines a particular model), one for each regressor in a maximal model. Assume an independence prior on each γ_i, or

$$f(\boldsymbol{\gamma})=\prod_{i=1}^{p} f(\gamma_i)=w^{q_\gamma}(1-w)^{p-q_\gamma}\,; \qquad q_\gamma=\sum_{i=1}^{p}\gamma_i \tag{4.9}$$

and the following prior for the regression coefficients included in model $\boldsymbol{\gamma}$

$$f(\boldsymbol{\beta}_\lambda \mid \sigma^2,\boldsymbol{\lambda})=N_{q_\gamma}\left(\boldsymbol{0},c\sigma^2\left(\boldsymbol{X}_\gamma{}'\boldsymbol{X}_\gamma\right)^{-1}\right) \tag{4.10}$$

Then it can be shown that the marginal posterior probability density of the model is

$$f(\boldsymbol{\lambda} \mid \boldsymbol{y}) \propto exp\left[\frac{c}{2(1+c)}\left\{SS_\gamma/\sigma^2-F(c,w)q_\gamma\right\}\right]$$

where $SS_\gamma=\hat{\boldsymbol{\beta}}_\gamma'\boldsymbol{X}_\gamma'\boldsymbol{X}_\gamma\hat{\boldsymbol{\beta}}_\gamma$, $\hat{\boldsymbol{\beta}}_\gamma=\left(\boldsymbol{X}_\gamma'\boldsymbol{X}_\gamma\right)^{-1}\boldsymbol{X}_\gamma'\boldsymbol{y}$ and (4.11)

$$F(c,w)=\frac{1+c}{c}\left[2log\frac{1-w}{w}+log(1+c)\right]$$

The difference in {.} in the exponent in Equation (4.11) equals the BIC criterion, where *F(c,w)* is the penalty for BIC with *c* = N and *w* = 0.5. Using *w* = 0.5 implies that most of the prior probability is assigned to a model with p/2 parameters, and therefore for sparse models this value should not be a good choice.

We are currently implementing a full Bayesian analysis of the SEM for gene network reconstruction. Due to the presence of cycles in gene networks, an efficient empirical Bayes analysis does not seem to be available, requiring us to implement a full Bayesian approach via a Markov chain Monte Carlo (MCMC) algorithm. Our prior for the parameters in ***B (F)*** depends on hyper-parameters c_b and w_b (c_f and w_f), which are given non-informative priors and are included in the MCMC sampling to evaluate whether these parameters can be simultaneously inferred from the data. Although theoretically very appealing, this approach may have practical problems resulting from poor convergence of the sampler. It is possible that the ML method presented in this contribution may provide excellent starting values that facilitate convergence of the Bayesian analysis.

Our SEM model can be generalized to include certain types of interactions: those between an eQTL and a regulator gene jointly trans-regulating a target gene and epistatic interactions between eQTL found in the eQTL analysis and hence included in the EDN. This extended model can be represented as

$$\begin{aligned} \boldsymbol{y}_i &= \boldsymbol{B}\boldsymbol{y}_i + \boldsymbol{F}\boldsymbol{x}_i + \boldsymbol{H}(\boldsymbol{x}_i \circ \mathrm{y}_i) + \boldsymbol{\Psi}\boldsymbol{w}_i + \boldsymbol{e}_i \\ &= \boldsymbol{B}\boldsymbol{y}_i + \boldsymbol{F}\boldsymbol{x}_i + \boldsymbol{H}\boldsymbol{D}_{x_i}\boldsymbol{y}_i + \boldsymbol{\Psi}\boldsymbol{w}_i + \boldsymbol{e}_i; \quad \boldsymbol{i} = 1,\ldots,N; \quad Var(\boldsymbol{e}_i) = \boldsymbol{E} \end{aligned} \tag{4.12}$$

where:

$\boldsymbol{x}_i \circ \boldsymbol{y}_i$ is the Hadamard or element-wise product of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$; here we assume that there is a QTL for each gene (real or fictitious, so $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ have the same dimension) but we allow for interactions only between a regulator gene and its corresponding QTL in a trans-regulation; $\boldsymbol{H}$ is a matrix of etrait-by-QTL interaction effects; row g in $\boldsymbol{H}$ contains nonzero elements only in those columns which correspond to trans-regulations of gene g, where there is an interaction between the gene regulator and its trans-linked eQTL; $\boldsymbol{w}_n$ is a vector of products of the codes of two eQTL genotypes; $\boldsymbol{\Psi}$ is a matrix of effects of pairwise epistatic interactions among eQTL; $\boldsymbol{D_{xi}}$ is a diagonal matrix with vector $\boldsymbol{x_i}$ on the diagonal. With this model, we can again solve for $\boldsymbol{y_i}$ and assume a normal distribution for the residuals.

Lastly, in this study here we have considered a network with only causal, directed interactions or regulations. However, two genes may be correlated, but there may be no eQTL information available to determine causation. At least in theory such associations or undirected edges can be incorporated via correlations in the residual covariance matrix $\boldsymbol{E}$. One can then include these off-diagonal elements in $\boldsymbol{E}$ in the EDN and consider them as potentially present in the model search. However, this would pose a computational problem, as the presence of off-diagonal elements in $\boldsymbol{E}$ would hinder the factorization of the likelihood.

Our network inference algorithm was implemented in C++, and the essential programs are shown in the Appendix.

4.5 APPENDIX: THE C++ PROGRAM

This program sparsfies a given Encompassing Directed Network (EDN) based on estimated IC from Structural Equation Modeling (SEM). For likelihood maximization, the program proceeds as follows:

1. Determine the cycle components of the genes using the B matrix.
2. For all genes that are not part of a cycle, their maximum likelihoods are estimated separately using linear regression.
3. For each cycle components, new **B, F, X, Y** matrices was formed and their maximum likelihoods are estimated using Genetic Algorithms (GA). First, initial estimates are obtained using Two-stage Least-Squares (2SLS). GA uses four kinds of starting values: random starting points; starting values obtained from 2SLS; starting values

equal to the current parameter estimates; and starting values from the current parameter values for all genes except 2SLS estimates for the genes directly affected by the deletion or addition of an edge.

4. Triplet form (I-B) matrices (for the cycle components) are formed to calculate the determinant using sparse LU decomposition.

For the model search, only the gene or cycle affected by the deletion/addition of edges is re-estimated. QTL edges are removed first, then the gene edges. Path constraints are checked at the beginning of QTL/gene edge deletion. Using the estimates from SEM, the search algorithm proceeds as follows:

0) Initialize set ***A*** = set of acceptable models as empty, set ***C*** = set of candidate models, and set ***K*** = set of models with minimum IC (the model selection criterion) as empty.

1) Select a model ***M*** from set ***C***, remove it from set ***C*** and add it to set ***A***. Let minIC=0.

2) Select a submodel $\boldsymbol{M}_0$ of ***M*** by removing an edge from ***M.***

3) Compute IC_{01}

4) If $IC_{01} < O_t$ (some negative constant), remove ***M*** from set ***A*** and add $\boldsymbol{M}_0$ to set ***C*** if $\boldsymbol{M}_0 \notin C$. Remove any model in set ***K*** and set minIC = -• (do not check for models with minimum IC anymore for this model).

5) If $O_t < IC_{01} <$ minIC, replace the model in set ***K*** with $\boldsymbol{M}_0$, and remove ***M*** from set ***A.***

6) If minIC $< IC_{01} < 0$ and this model is chosen as a random start, remove ***M*** from set ***A*** and add $\boldsymbol{M}_0$ to set ***C*** if $\boldsymbol{M}_0 \notin C$.

7) If there are more sub models of ***M***, go to 2. Otherwise, remove the model in set ***K*** and put it in set ***C*** if it is not already in set ***C.***

8) If ***C*** is not empty, go to 1.

Starting from all models accepted in the Down algorithm, the Up algorithm follows the same steps as in the Down algorithm, except every time an edge that was removed from the EDN is added back into the model. Once the Up algorithm is completed, the set A contains the set of potentially acceptable models.

The following are the essential parts of the search program.

```
/* -----------------------------------------------------------------------------------------------------------------------

   DESCRIPTION: This program performs gene network model selections with SEM.
   ----------------------------------------------------------------------------------------------------------------------- */

#include <scsl_blas.h>
#include <ga/ga.h>
#include <ga/std_stream.h>
#include <ga/GARealGenome.h>

// C and fortran linear algebra liberaries
extern "C" {
#include "umfpack.h"
 void dgetri_(int *N, double *A, int *LDA, int *IPIV, double *WORK, int *LWORK, int *INFO);
 void dgetrf_ (int *M, int *N, double *A, int *LDA, int *IPIV, int *INFO);
 void dpotrf_(char *, int *, double *, int *, int *);
}

#define INSTANTIATE_REAL_GENOME


void myInitializer(GAGenome &);


int main(int argc, char** argv)
{

 for (int simudataid=0; simudataid<9; simudataid++) {

  double * xdata;
  double * ydata;

  int npar=0;
  int themodel=0;
  int nb=0, nf=0,ne=0;

  // This block create the y and x matrixs;
  int sizey=geneNum *samplesize;
  xdata =new (nothrow)  double [samplesize*numQTL] ;
  if (xdata  == 0) {
   cout << "Error: memory could not be allocated for xdata";
  }
  ydata =new (nothrow)  double [sizey] ;
  if (ydata  == 0) {
   cout << "Error: memory could not be allocated for ydata";
  }


  // this block read the number of non-zeros in B and F;
  ne=geneNum;
  double tempRead=0.;
  ifstream InFile1 (bfileName.c_str());
  nb=0;
  while(InFile1)
   {
         if(InFile1>>tempRead>>tempRead)
          nb++;
   }
  InFile1.close();

  ifstream InFile2 (ffileName.c_str());
  nf=0;

  while(InFile2)
   {
         if(InFile_1>>tempRead>>tempRead)
          nf++;
   }
  InFile2.close();


  npar=nb+nf+ne;
  int maxedge=0;

  if (nb>nf){
   maxedge=max(ne,nb);
  }else{
   maxedge=max(ne,nf);
  }

  if (nf==0) {
   numQTL=0;
  }

/* This matrix store the model space M. The first row: accepted(1), under consideration (0) or rejected (-1); number gene
edges;  number qtl edges; the level, it's topmodel. The rest: 1/0 show absense/presence of B edges in the EDN; constraint to
not removable (1) or 0;  same 2 col for F; Number gene edges; number qtl edges */

  int modelspace[maxmodelnumber][maxedge+1][6]; // model 0 is the EDN

  int  *** modelspace = new int ** [maxmodelnumber ];
  int  *** modelspacenew1 = new int ** [maxmodelnumber ];
  int  *** modelspacenew2 = new int ** [maxmodelnumber ];
  int  *** modelspacenew3 = new int ** [maxmodelnumber ];

  for(int i=0; i<maxmodelnumber ; i++){
   modelspace[i] = new  int * [maxedge+1 ];
   modelspacenew1[i] = new  int * [maxedge+1 ];
   modelspacenew2[i] = new  int * [maxedge+1 ];
   modelspacenew3[i] = new  int * [maxedge+1 ];
  }

  for(int i=0; i<maxmodelnumber ; i++){
   for(int j=0; j< (maxedge+1); j++){
         modelspace[i][j] = new  int[6];
         modelspacenew1[i][j] = new  int[6];
         modelspacenew2[i][j] = new  int[6];
         modelspacenew3[i][j] = new  int[6];
   }
  }
```

```
for(int i=0; i<maxmodelnumber; i++) {
  for(int j=0; j<(maxedge+1); j++){
        for(int k=0; k<6; k++) {
          modelspace[i][j][k] = 0;
          modelspacenew1[i][j][k] = 0;
          modelspacenew2[i][j][k] = 0;
          modelspacenew3[i][j][k] = 0;
        }
  }
}


// The first row: accepted(1) or under consideration (0);number gene edge in EDN; number qtl edge in EDN
// The rest: nonzero B row index;nonzero B col index ;  gene affected by QTL; affecting QTL; nonzero E row index;
//  nonzero E col index.
int edn[maxedge+1][6];

// The first row: model BIC compare to it's parent model; model likelihood;
// Columns: B estimates; F; E; likelihood for genes; likelihood for cycles; sigma2hat estimated from OLS
// BFE estimates are only for cycle components; likelihood for genes and sigma2hat are for all genes
double  *** modelspacepar = new double ** [maxmodelnumber ];
double  *** modelspaceparnew1 = new double ** [maxmodelnumber ];
double  *** modelspaceparnew2 = new double ** [maxmodelnumber ];
double  *** modelspaceparnew3 = new double ** [maxmodelnumber ];

for(int i=0; i<maxmodelnumber ; i++){
  modelspacepar[i] = new  double * [maxedge+1 ];
  modelspaceparnew1[i] = new  double * [maxedge+1 ];
  modelspaceparnew2[i] = new  double * [maxedge+1 ];
  modelspaceparnew3[i] = new  double * [maxedge+1 ];
}

for(int i=0; i<maxmodelnumber ; i++){
  for(int j=0; j< maxedge+1; j++){
        modelspacepar[i][j] = new  double[6];
        modelspaceparnew1[i][j] = new  double[6];
        modelspaceparnew2[i][j] = new  double[6];
        modelspaceparnew3[i][j] = new  double[6];
  }
}

for(int i=0; i<maxmodelnumber; i++) {
  for(int j=0; j<(maxedge+1); j++){
        for(int k=0; k<6; k++) {
          modelspacepar[i][j][k] = 0;
          modelspaceparnew1[i][j][k] = 0;
          modelspaceparnew2[i][j][k] = 0;
          modelspaceparnew3[i][j][k] = 0;
        }
  }
}


modelspace[0][0][1] =nb ;
```

```
modelspace[0][0][2] =nf ;

edn[0][0] =0 ; edn[0][1] =nb ; edn[0][2] =nf ;  edn[0][3] =ne ;

// This block store the encompassing network;
// B and F are sorted by the first column (targets);
int tempi=0; int lasttempi=0;
int tempindex=0;
int tempj;  int tempcount=0;
ifstream InFile4;
InFile4.open(bfileName.c_str());
InFile4>>lasttempi;
InFile4.close();

ifstream InFile3;
InFile3.open(bfileName.c_str());

while(InFile3) {
  if(InFile3>>tempi>>tempj){
          edn[tempindex+1][0] =tempi-1 ;
          edn[tempindex+1][1] =tempj-1;
          modelspacepar[themodel][tempindex+1][0] =changestartingvalue ;
          tempindex++;

          if (lasttempi!=tempi){
            modelspace[0][lasttempi][4] =tempcount ;
            tempcount=0;
          }
          lasttempi=tempi;
          tempcount++;
  }
}
modelspace[0][lasttempi][4] = tempcount;
InFile3.close();

tempindex=0;      tempcount=0;
ifstream InFile5;
InFile5.open(ffileName.c_str());
InFile5>>lasttempi;
InFile5.close();

ifstream InFile6;
InFile6.open(ffileName.c_str());

while(InFile6) {
          if (InFile6>>tempi>>tempj){
            edn[tempindex+1][2] =tempi-1 ;
            edn[tempindex+1][3] =tempj-1;
            modelspacepar[themodel][tempindex+1][1] =changestartingvalue ;
            tempindex++;

            if (lasttempi!=tempi){
              modelspace[0][lasttempi][5] = tempcount;
              tempcount=0;
            }
          lasttempi=tempi;
          tempcount++;
        }
  }
modelspace[0][lasttempi][5] = tempcount;
InFile6.close();

for (int j=0; j<geneNum ; j++){
  edn[j+1][4] =j;
  edn[j+1][5] =j;
  modelspacepar[themodel][j+1][2] =changestartingvalue ;
}

ifstream InFile7 (yfileName.c_str());

readDoubleM(samplesize, geneNum , InFile7, ydata);
InFile7.close();

ifstream InFile8 (xfileName.c_str());   // col: qtl; rows: samples
readDoubleM(samplesize, numQTL, InFile8, xdata);
InFile8.close();

int * cycIndex = new int[geneNum ];

for(int i=0; i<geneNum ; i++){
        cycIndex[i] = 0;
  }

 findcyclecomponents (cycIndex);

 // Create path matrix if needed

  int totalvariables=geneNum+numQTL;
  int ** PathPresMatBF = new int * [totalvariables];
  int ** tempPathPresMat = new int * [totalvariables];

  for(int i=0; i<totalvariables ; i++){
        PathPresMatBF[i] = new int [totalvariables];
        tempPathPresMat[i] = new int [totalvariables];
  }
  int * tempPathMat = new int [totalvariables*totalvariables ];
  int * tempAdjMat = new int [totalvariables*totalvariables ];

  for(int j=0; j<totalvariables*totalvariables ; j++){
        tempAdjMat[j] = 0;
        tempPathMat[j]=0;
  }

  for(int i=0;i<totalvariables ;i++){
        for(int j=0;j<totalvariables ;j++){
          PathPresMatBF[j][j]=0;
          tempPathPresMat[j][j]=0;
        }
  }
```

```
reconstructPath ( constraintonQTLorGene, tempAdjMat,  PathPresMatBF,  tempPathMat,
                              modelspace , edn, 0,  totalvariables,  geneNum, maxdistforpath ) ;

/////////////////////////////////// ORDINARY LEAST SQUARES

// This block create the matrices needed for the OLS;

double * yi;  yi =new (nothrow)  double [samplesize] ;
double * ui;  ui =new (nothrow)  double [samplesize] ;

for(int i = 0;i<geneNum ;i++){
        double   olssigma;
        int isqtl=0; int regulator=-1;
        double olsoutput=olsforonegene(olssigma, isqtl, regulator, geneNum, i, numQTL, samplesize, ydata,
                                             modelspace, 0,xdata,edn, yi, ui);

          modelspacepar[themodel][i+1][3]=olsoutput;
          modelspacepar[themodel][i+1][5]=olssigma;
}

double modellikelihood=0;

for (int k=0;k<geneNum ;k++){
        if (cycIndex[k]==0){
          modellikelihood=modelspacepar[themodel][k+1][3]+modellikelihood;
        }
}

////////////////////////////////////////////////////////////////////////////////////////////////////
// The following block estimates likelihood for genes in the cycles /////
/////////////////////////////////////////////////////////////////////////////////////////////////////

int * yinputforyi = new int [geneNum ];
int * xinputforyi = new int [numQTL];
int * ycnewidx = new int [geneNum ];
int * ypnewidx = new int [geneNum ];
int * xnewidx = new int [numQTL];
int isedn=1;

int *  bmodelidx;
int *  fmodelidx;
int *  emodelidx;

bmodelidx = new (nothrow) int [nb] ;
fmodelidx = new (nothrow) int [nf] ;
emodelidx = new (nothrow) int [ne] ;

for (int k=0; k<numcycles; k++){
        thecyclenumber=k+1;
        int isqtl=0;
        int regulator=-1;
        modelspacepar[themodel][k+1][4]=likelihoodforonecycle(isedn, 0, 0, k,yinputforyi, xinputforyi, ycnewidx,
                                                              ypnewidx, xnewidx, modelspace, modelspacepar,
                                                              geneNum, samplesize, numQTL,
                                                        ydata, xdata, edn,cycIndex, isqtl, regulator,
                                                        -1, bmodelidx, fmodelidx, emodelidx, isup);

        modellikelihood=modellikelihood+ modelspacepar[themodel][k+1][4]; // likelihoodforcycles[k];

}

cout<<"modellikelihood:     "<<modellikelihood<<endl;

modelspacepar[themodel][0][0]=0;
modelspacepar[themodel][0][1]=modellikelihood ;

OutFile << "Finished with EDN!  \n";

//////////////////////////////////////////////////////////////////////
// The following block search the model space within the edn /////////
//////////////////////////////////////////////////////////////////////

// Before the search, copy model spec of the EDN to the temp model spaces
for(int j=0; j<(maxedge+1); j++){
        for(int k=0; k<6; k++) {
          modelspaceparnew1[0][j][k] = modelspacepar[0][j][k] ;
          modelspaceparnew2[0][j][k] = modelspacepar[0][j][k] ;
          modelspaceparnew3[0][j][k] = modelspacepar[0][j][k] ;
          modelspacenew1[0][j][k] = modelspace[0][j][k] ;
          modelspacenew2[0][j][k] = modelspace[0][j][k] ;
          modelspacenew3[0][j][k] = modelspace[0][j][k] ;
        }
}

int totalnummodelaccepteddown=0;
int totalnummodelafterqtldown=0;
 int totalrandomstart=0;
 int minmodelidx=0;
 int searchlevel =0;
double minbic=9e+99;
int numberedgeremoved=1;
isup=0;
int donedownsearch=0;
int firstmodel=themodel;
int lastmodel=themodel;
int newfirstmodel=0;
int newlastmodel=0;
for (int e=1; e<=edn[0][1]+edn[0][2];e++){
        int isqtl=0;
        int targetgene=0;
        int regulator=0;
        int removedcheck=0;
        int topmodel=0;
        if (e<=edn[0][1]){
          targetgene=edn[e][0];
          regulator=edn[e][1];
          removedcheck= modelspace[topmodel][e][0];
        }else{
          targetgene=edn[e-edn[0][1]][2];
```

```
      regulator=edn[e-edn[0][1]][3];
      isqtl=1;
      removedcheck= modelspace[topmodel][e-edn[0][1]][2];
    }
}

int startingedge=edn[0][1]+edn[0][2];
int endingedge = edn[0][1]+1;
int donewithQTLsearch=0;

while (donedownsearch ==0){
    int isedn=0;
    newfirstmodel=lastmodel+1;
    newlastmodel=lastmodel;

    if (donewithQTLsearch==1){
      startingedge=edn[0][1];
      endingedge = 1;
    }

    for (int topmodel=firstmodel; topmodel<(lastmodel+1); topmodel++){ //for each starting model in the level
     if (modelspace[topmodel][0][0]!=-1){
      int storingmin =0;
      minbic=0;

      if (donewithQTLsearch==1 & searchlevel==0){
        reconstructPath  (constraintonQTLorGene,   tempAdjMat,  PathPresMatBF, tempPathMat ,
                                       modelspace , edn, topmodel,  totalvariables,  geneNum, maxdistforpath ) ;
      }

      for (int theedge= startingedge; theedge>=endingedge ; theedge--)  {
        int isqtl=0;
        int targetgene=0;
        int regulator=0;
        int removedcheck=0;

        if (theedge<=edn[0][1]){ // If it is a gene edge;
          targetgene=edn[theedge][0];
          regulator=edn[theedge][1];
          removedcheck= modelspace[topmodel][theedge][0];
        }else{      // For the QTL links
          targetgene=edn[theedge-edn[0][1]][2];
          regulator=edn[theedge-edn[0][1]][3];
          isqtl=1;
          removedcheck= modelspace[topmodel][theedge-edn[0][1]][2];
        }

        int constraintnoremove=0;

        if ( constraintonQTLorGene!=0){
          constraintnoremove=checkpathforconstriant( constraintonQTLorGene,  theedge, regulator,  targetgene,
                                                                                    isqtl, tempAdjMat, PathPresMatBF,tempPathMat,
                                                                                    tempPathPresMat, modelspace, edn,topmodel,
                                                                                    totalvariables, geneNum, maxdistforpath,searchlevel);
        }
```

```
if ( removedcheck==0 && constraintnoremove==0) { // if the edge is not removed already, and is removable

 if (cycIndex[targetgene]==0){ // If the edge going into a gene that is not part of a cycle

        double targetlikelihood=modelspacepar[topmodel][targetgene+1][3];
        double olssigma=0;

        double newtargetlikelihood =olsforonegene(olssigma, isqtl, regulator,geneNum,targetgene, numQTL,
                                                        samplesize, ydata,modelspace, topmodel,xdata,edn, yi, ui);

        double bic=getIC(ICtouse,newtargetlikelihood, targetlikelihood, numberedgeremoved,samplesize,
                        geneNum,numQTL,lodthresholdforbicdelta);

        if (bic<biccutoff){

           int isdupmodel= checkduplicatemodel(modelspace,edn, newfirstmodel, newlastmodel+1,topmodel,
           targetgene, isqtl, theedge, maxedge );

           if (isdupmodel==1){
             cout<<"duplicate model, no need to save"<<endl<<endl;
             if ( storingmin==1 ){ // If this is the one replacing the min model
                    storingmin =0;
                    modelspace[minmodelidx][0][0]=-1;
             }
           }else{
             if ( storingmin==1 ){ // If this is the one replacing the min model
                    storingmin =0;
             }else{
                    newlastmodel++; // one model into the space
                    minmodelidx=newlastmodel;
             }
             cout<<"another model in space:    "<<minmodelidx <<endl;
             storenestedmodel(modelspacepar, modelspace,edn,minmodelidx,topmodel,bic,newtargetlikelihood,
                              targetlikelihood , targetgene, isqtl,olssigma, searchlevel,theedge,maxedge, isup);
           }

          minbic=-9e+99; // no more check for min

        }else if (bic<minbic  ){
           minbic =bic;
           int isdupmodel= checkduplicatemodel(modelspace, edn, newfirstmodel,newlastmodel+1, topmodel,
                                                        targetgene, isqtl, theedge, maxedge );

           if (isdupmodel==1){
             cout<<"duplicate model, no need to save"<<endl<<endl;
             if (storingmin==1){ //since this min is already in space, leave out the space
                    storingmin =0;
                    modelspace[minmodelidx][0][0]=-1;
             }
           }else{
             if (storingmin==0){ //if no min of this model has been stored
                    storingmin =1;
                    newlastmodel++;
                    minmodelidx=newlastmodel;
```

```
       }
       cout<<"Store the min model in space:    "<<minmodelidx <<endl;
       storenestedmodel(modelspacepar, modelspace,edn,minmodelidx,topmodel,bic,newtargetlikelihood,
                      targetlikelihood , targetgene, isqtl,olssigma, searchlevel,  theedge,maxedge, isup);

      }

    } else if (bic<0){
     int arandomnumber= GARandomInt(1,  100);
     if ( arandomnumber<=( 100*randomperc ) && totalrandomstart<=maxrandomstart){
      int isdupmodel= checkduplicatemodel(modelspace, edn, newfirstmodel, newlastmodel+1,topmodel,
                                        targetgene, isqtl, theedge, maxedge );
      if (isdupmodel!=1){
       newlastmodel++; // one model into the space
       storenestedmodel(modelspacepar, modelspace,edn,  newlastmodel,topmodel,bic,newtargetlikelihood,
                      targetlikelihood , targetgene, isqtl,olssigma, searchlevel,theedge,maxedge, isup);
       totalrandomstart++;
      }
     }
    }
}
else{ // If going to a gene that is part of a cycle
    int thecycle =cycIndex[targetgene];
    int tempmodel=maxmodelnumber-1;
    double targetlikelihood=modelspacepar[topmodel][thecycle][4];

    double newtargetlikelihood =likelihoodforonecycle(isedn,topmodel, tempmodel,(thecycle-1),
                                        yinputforyi,  xinputforyi,ycnewidx,  ypnewidx, xnewidx,
                                        modelspace, modelspacepar, geneNum, samplesize, numQTL,
                                        ydata, xdata, edn, cycIndex, isqtl, regulator,
                                        targetgene, bmodelidx, fmodelidx, emodelidx, isup);

    double bic=getIC(ICtouse,newtargetlikelihood, targetlikelihood, numberedgeremoved,samplesize,
                   geneNum,numQTL, lodthresholdforbicdelta);

    if (bic<biccutoff){

    int isdupmodel= checkduplicatemodel(modelspace,edn, newfirstmodel, newlastmodel+1,topmodel,
    targetgene, isqtl, theedge, maxedge );

      if (isdupmodel==1){
       cout<<"duplicate model, no need to save" <<endl;

       if ( storingmin==1 ){ // If this is the one replacing the min model
           storingmin =0;
           modelspace[minmodelidx][0][0]=-1;
       }
      }else{
       if ( storingmin==1 ){ // If this is the one replacing the min model
           storingmin =0;
       }else{
           newlastmodel++; // one model into the space
           minmodelidx=newlastmodel;
       }
       cout<<"another model in space:    "<<minmodelidx <<endl;
```

```
         storenestedmodel(modelspacepar, modelspace,edn,minmodelidx,topmodel,bic,newtargetlikelihood,
                        targetlikelihood , targetgene, isqtl,olssigma, searchlevel,theedge,maxedge, isup);

        }
       minbic=-9e+99; // no more check for min

      }else if (bic<minbic){
       minbic =bic;
        int isdupmodel= checkduplicatemodel(modelspace, edn, newfirstmodel,newlastmodel+1, topmodel,
                                         targetgene, isqtl, theedge, maxedge );

        if (isdupmodel==1){
         cout<<"duplicate model, no need to save"<<endl<<endl;
         if (storingmin==1){ //since this min is already in space, leave out the space
             storingmin =0;
             modelspace[minmodelidx][0][0]=-1;
         }
        }else{
         if (storingmin==0){ //if no min of this model has been stored
             storingmin =1;
             newlastmodel++;
             minmodelidx=newlastmodel;
         }
         cout<<"Store the min model in space:    "<<minmodelidx <<endl;
         storenestedmodelcycle(modelspacepar, modelspace, edn, minmodelidx,topmodel, bic,
         newtargetlikelihood, targetlikelihood , targetgene, isqtl, searchlevel, theedge, maxedge, thecycle,
                      tempmodel,bmodelidx, fmodelidx, emodelidx, isup);

        }
      }
  } // end of if (cycIndex[targetgene]==0)
 } //end of if the edge is not removed
} // End of going through all QTL or gene linkes

 // If the min bic of the current model is larger than 0, the top model cannot be improved and therefore
 // change the status to accepted for the topmodel
if (minbic>=0){
 if (donewithQTLsearch==0){ // If searching through the QTL links
  totalnummodelafterqtldown++;

  // Results from the QTL search are starting point for the gene link search
  for(int j=0; j<(maxedge+1); j++){
      for(int k=0; k<6; k++) {
       modelspaceparnew1[totalnummodelafterqtldown][j][k] = modelspacepar[topmodel][j][k] ;
       modelspacenew1[totalnummodelafterqtldown][j][k] = modelspace[topmodel][j][k] ;
      }
  }

  int isdupmodel= checkduplicatemodel( modelspacenew1, edn, 1,  totalnummodelafterqtldown,
                                    totalnummodelafterqtldown,  1, -10,1,  maxedge );
  if (isdupmodel==1){
      cout<<"duplicate model, no need to save"<<endl<<endl;
      totalnummodelafterqtldown--; // Leave out the space
  }else{
      string outputfileName = getFileName(simudataid, "data_");
```

```
        outputfileName += "_model_";
        outputfileName = getFileName(topmodel-1, outputfileName);
        outputfileName+= "_downQTLsearch.txt";
        ofstream OutFile4(outputfileName.c_str());
        for(int j=0; j<(maxedge+1); j++){// Note: the top row is not for parameters
          for(int k=0; k<6; k++) {
           OutFile4<< modelspace[topmodel][j][k] <<'\t' ;
          }
          OutFile4 <<endl;
        }
        OutFile4.close();
  }else{
   cout <<"accepted one model for the down search:  "<<topmodel<<endl;
   totalnummodelaccepteddown++;
   cout << "the edges removed are:   "<<endl;

   // Results from the down search are starting point for up search
   for(int j=0; j<(maxedge+1); j++){
        for(int k=0; k<6; k++) {
          modelspaceparnew2[totalnummodelaccepteddown][j][k] = modelspacepar[topmodel][j][k] ;
          modelspacenew2[totalnummodelaccepteddown][j][k] = modelspace[topmodel][j][k] ;
        }
   }

   int isdupmodel= checkduplicatemodel( modelspacenew2, edn, 1, totalnummodelaccepteddown ,
                                        totalnummodelaccepteddown,  1, -10,1,  maxedge );
   if (isdupmodel==1){
        cout<<"duplicate model, no need to save"<<endl<<endl;
        totalnummodelaccepteddown--; // Leave out the space
   }else{
        cout <<"accepted one model for the down search:  "<<topmodel<<endl;
        modelspace[topmodel][0][0] =1;
        string outputfileName = getFileName(simudataid, "data_");
        outputfileName += "_model_";
        outputfileName = getFileName(topmodel-1, outputfileName);
        outputfileName+= "_downsearch.txt";
        ofstream OutFile5 (outputfileName.c_str());
         for(int j=0; j<(maxedge+1); j++){// Note: the top row is not for parameters
              for(int k=0; k<6; k++) {
                OutFile5 << modelspace[topmodel][j][k] <<'\t' ;
              }
              OutFile5 <<endl;
         }
         OutFile5.close();
         }
  }
 }
}
}

if (newlastmodel>lastmodel){ // if there're more models in the next level
 searchlevel++;
 addconstraintfromtopmodel (newfirstmodel, newlastmodel, modelspace, maxedge);

 firstmodel=newfirstmodel;
```

```
      lastmodel=newlastmodel;
      cout<< "new first model is " <<firstmodel << " and the new lastmodel: " <<lastmodel<<endl;

     }else if ( donewithQTLsearch==0) {// If working on the QTL search
      searchlevel=0; // Fist step in the gene search
      donewithQTLsearch=1;
      ICtouse=ICforgene; // If IC for qtl and gene are diff, switch here

      firstmodel=1;
      lastmodel=totalnummodelafterqtldown ;
      cout << "Finished with QTL links in the down search."<<endl<<endl;
      cout<< "new first model is " <<firstmodel << " and the new lastmodel: " <<lastmodel<<endl;

      for(int i=0; i<maxmodelnumber; i++){
       for(int j=0; j<(maxedge+1); j++){
        if( modelspace[i][j])
             delete[] modelspace[i][j];
        if (modelspacepar[i][j])
             delete[] modelspacepar[i][j];
       }
      }
      for(int i=0; i<6; i++){
       if( modelspace[i])
        delete[] modelspace[i];
       if (modelspacepar[i])
        delete[] modelspacepar[i];
      }
      if (modelspace)
       delete[] modelspace;
      if (modelspacepar)
       delete[] modelspacepar;
      modelspace=modelspacenew1; // now use the first newspace as the starting point.
      modelspacepar=modelspaceparnew1;

     }else{ // if nothing in the next level, done!!
      searchlevel=0;

      cout << "Finished with the down search."<<endl<<endl;
      modelspace=modelspacenew2; // now use the second temp newspace as the starting point for the up search
      modelspacepar=modelspaceparnew2;
      donedownsearch=1;
     } // End of if : there is more model in the next level

}  // End of the down search

cout<<"total number of model accepted in the down search:   "<< totalnummodelaccepteddown<<endl<<endl;

//////// End of down search, start up search     /////////////////////

cout<<"Start upward search.................................................   "<<endl;

int totalnummodelaftergeneup=0;
int totalnummodelacceptedup=0;
firstmodel=1;
lastmodel=totalnummodelaccepteddown ;
```

```
 double maxbic=0;
 int numberedgeadd=1;
 int donesearch=0;
 newfirstmodel=0;
 newlastmodel=0;
 isup =1;
startingedge=edn[0][1];
 endingedge = 1;
 int donewithGenesearch=0;

 while (donesearch ==0){
         int isedn=0;
         newfirstmodel=lastmodel+1;
         newlastmodel=lastmodel;

         if (donewithGenesearch==1){
           startingedge=edn[0][1]+edn[0][2];
           endingedge = edn[0][1]+1;
         }

         for (int topmodel=firstmodel; topmodel<=lastmodel; topmodel++){ //for each starting model in the level
           int storingmax =0;
           maxbic=0;

           for (int theedge= startingedge; theedge>=endingedge ; theedge--)  {
             int isqtl=0;
             int targetgene=0;
             int regulator=0;
             int removedcheck=0;

             if (theedge<=edn[0][1]){
               targetgene=edn[theedge][0];
               regulator=edn[theedge][1];
               removedcheck= modelspace[topmodel][theedge][0];
             }else{
               targetgene=edn[theedge-edn[0][1]][2];
               regulator=edn[theedge-edn[0][1]][3];
               isqtl=1;
               removedcheck= modelspace[topmodel][theedge-edn[0][1]][2];
             }

             if ( removedcheck==1) { // if the edge is  removed

               if (cycIndex[targetgene]==0){ // If the edge going into a gene that is not part of a cycle
                       double targetlikelihood=modelspacepar[topmodel][targetgene+1][3];
                       double olssigma=0;
                       double newtargetlikelihood =olsforonegeneup(olssigma, isqtl, regulator,geneNum,targetgene, numQTL,
                                                                            samplesize, ydata,modelspace, topmodel,xdata,edn, yi, ui);

                       double bic=getIC(ICtouse,targetlikelihood, newtargetlikelihood, numberedgeadd,samplesize,
                                             geneNum,numQTL, lodthresholdforbicdelta);

                       if (bic>biccutoffup){
                         if ( storingmax==1 ){
                           storingmax =0;
```

```
     }else{
      newlastmodel++;
     }

     int isdupmodel= checkduplicatemodelup(modelspace, edn, newfirstmodel, newlastmodel, topmodel,
                                          targetgene, isqtl, theedge, maxedge );

     if (isdupmodel==1){
      newlastmodel--;
     }else{
      storenestedmodel(modelspacepar, modelspace,edn,  newlastmodel,topmodel,bic,newtargetlikelihood,
                     targetlikelihood, targetgene,isqtl,olssigma, searchlevel, theedge,maxedge,isup);
     }

     maxbic=9e+99; // no more check for max

    }else if (bic>maxbic  ){
     maxbic =bic;

     if (storingmax==0){ //if no max of this model has been stored
      storingmax =1;
      newlastmodel++;
     }

     int isdupmodel= checkduplicatemodelup(modelspace,edn,newfirstmodel,newlastmodel, topmodel,
                                          targetgene, isqtl, theedge, maxedge );

     if (isdupmodel==1){
      cout<<"duplicate model, no need to save"<<endl<<endl;

      if (storingmax==1){ //since this max is already in space, leave out the space
       storingmax =0;
       newlastmodel--;
      }
     }else{
      cout<<"Store the max model in space:    "<<newlastmodel<<endl;
      storenestedmodel(modelspacepar, modelspace,edn,newlastmodel,topmodel,bic,newtargetlikelihood,
                     targetlikelihood, targetgene,isqtl,olssigma, searchlevel,theedge,maxedge,isup);
     }
    }

}
else{
    int thecycle =cycIndex[targetgene];
    int tempmodel=maxmodelnumber-1;
    double targetlikelihood=modelspacepar[topmodel][thecycle][4];

    double newtargetlikelihood=likelihoodforonecycle(isedn,topmodel, tempmodel,(thecycle-1),
                                          yinputforyi,  xinputforyi,ycnewidx,  ypnewidx, xnewidx,
                                          modelspace, modelspacepar, geneNum, samplesize, numQTL,
                                          ydata, xdata, edn, cycIndex, isqtl, regulator,
                                          targetgene, bmodelidx, fmodelidx, emodelidx, isup);

    double bic=getIC(ICtouse,targetlikelihood, newtargetlikelihood,
    numberedgeadd,samplesize,geneNum,numQTL, lodthresholdforbicdelta);
```

```
          if (bic>biccutoffup){
           if (  storingmax==1 ){
             storingmax =0;
           }else{
             newlastmodel++; // one model into the space
           }

           int isdupmodel=checkduplicatemodelup(modelspace, edn, newfirstmodel, newlastmodel, topmodel,
           targetgene, isqtl,theedge,  maxedge );

           if (isdupmodel==1){
             newlastmodel--;
           }

             storenestedmodelcycle(modelspacepar, modelspace,edn, newlastmodel,topmodel,
             bic,newtargetlikelihood, targetlikelihood , targetgene, isqtl, searchlevel,  theedge,maxedge,
                                         thecycle, tempmodel,bmodelidx, fmodelidx, emodelidx, isup );
           }

           maxbic=9e+99; // no more check for max

          }else if (bic>maxbic){
           maxbic =bic;
           if (storingmax==0){ //if no max of this model has been stored
             storingmax =1;
             newlastmodel++;
           }

           int isdupmodel= checkduplicatemodelup( modelspace,edn, newfirstmodel, newlastmodel, topmodel,
                                                               targetgene, isqtl, theedge, maxedge );
           if (isdupmodel==1){
             if (storingmax==1){ //since this max is already in space, leave out the space
               storingmax =0;
               newlastmodel--;
             }
           }else{
             cout<<"Store the max model in space:    "<<newlastmodel<<endl;
             storenestedmodelcycle(modelspacepar, modelspace, edn, newlastmodel,topmodel, bic, newtargetlikelihood,
                                         targetlikelihood, targetgene, isqtl, searchlevel, theedge, maxedge, thecycle,
                                         tempmodel,bmodelidx, fmodelidx, emodelidx, isup);
           }
          }
      }
     }
   }

   if (maxbic<=0){
     if (donewithGenesearch==0){ // If searching through the gene links
       totalnummodelaftergeneup++;

       // Results from the gene search are starting point for the QTL search
       for(int j=0; j<(maxedge+1); j++){
              for(int k=0; k<6; k++) {
                modelspaceparnew3[totalnummodelaftergeneup][j][k] = modelspacepar[topmodel][j][k] ;
```

```
         modelspacenew3[totalnummodelaftergeneup][j][k] = modelspace[topmodel][j][k] ;
        }
    }
    if (isdupmodel==1){
        cout<<"duplicate model, no need to save"<<endl<<endl;
        totalnummodelaftergeneup --; // Leave out the space
    }
   }else{
    cout <<"accepted one model for the up search:  "<<topmodel<<endl;
    totalnummodelacceptedup++;
    modelspace[topmodel][0][0] =1; // Accepted the top model
    string outputfileName = getFileName(simudataid, "data_");
    outputfileName += "_model_";
    outputfileName = getFileName(topmodel-1, outputfileName);
    outputfileName+= "_Upsearch.txt";
    ofstream OutFileUp (outputfileName.c_str());
    for(int j=0; j<(maxedge+1); j++){// Note: the top row is not for parameters
        for(int k=0; k<6; k++) {
         OutFileUp<< modelspace[topmodel][j][k] <<'\t' ;
        }
        OutFileUp<<endl;
    }
    OutFileUp.close();
   }
  }
}
if (newlastmodel>lastmodel){ // if there're more models in the next level
  searchlevel++;
  firstmodel=newfirstmodel;
  lastmodel=newlastmodel;
  cout<< "new first model is " <<firstmodel << " and the new lastmodel: " <<lastmodel<<endl;

}else if ( donewithGenesearch==0) {// If working on the gene search
  searchlevel=0;
  donewithGenesearch=1;
  ICtouse=ICforQTL; // Switch back to the IC for QTL
  firstmodel=1;
  lastmodel=totalnummodelaftergeneup ;
  cout << "Finished with gene links in the up search."<<endl<<endl;
  cout<< "new first model is " <<firstmodel << " and the new lastmodel: " <<lastmodel<<endl;

  for(int i=0; i<maxmodelnumber; i++){
   for(int j=0; j<(maxedge+1); j++){
    if( modelspace[i][j])
        delete[] modelspace[i][j];
    if (modelspacepar[i][j])
        delete[] modelspacepar[i][j];
   }
  }
  for(int i=0; i<6; i++){
   if( modelspace[i])
    delete[] modelspace[i];
   if (modelspacepar[i])
    delete[] modelspacepar[i];
  }
```

```
          if (modelspace)
            delete[] modelspace;
          if (modelspacepar)
            delete[] modelspacepar;

          modelspace=modelspacenew3; // now use the third temp newspace as the starting point.
          modelspacepar=modelspaceparnew3;

         }else{ // if nothing in the next level, done!!
          searchlevel++;

          cout << "Finished with the search!!!"<<endl<<endl;

          for(int i=0; i<maxmodelnumber; i++){
            for(int j=0; j<(maxedge+1); j++){
              if( modelspace[i][j])
                  delete[] modelspace[i][j];
              if (modelspacepar[i][j])
                  delete[] modelspacepar[i][j];
            }
          }
          for(int i=0; i<6; i++){
            if( modelspace[i])
              delete[] modelspace[i];
            if (modelspacepar[i])
              delete[] modelspacepar[i];
          }
          if (modelspace)
            delete[] modelspace;
          if (modelspacepar)
            delete[] modelspacepar;

          donesearch=1;
         } // End of if: there is more model in the next level

    }  // End of the search

    cout<<"total numember of model accepted   "<< totalnummodelacceptedup<<endl<<endl;
   //// End of search the model space within the edn ////
  } // End of the multiple data set loop

  return 0;
}


/* --------------------------------------------------------------------------------
This initializer uses four kinds of starting values for the individuals in a population:
      1.  Some percentage of the population have the 2sls starting values
      2.  Some percentage use the estimated values from the top model
      3.  Some percentage use estimates from the top model, except using 2sls results for all edges into the target gene
      4.  The other individual use randomized starting values
Note that a random number generator is used to assign the individuals to the four groups.
 ------------------------------------------------------------------------------- */

void  myInitializer(GAGenome & c)
```

```
{
 double changestartingvalue=1; // can be used to change the starting values. Use 1 by default.
 int anumber= GARandomInt(1,  popsize);

 int bidx=cyclenb+cyclep-1;
 GARealGenome &genome= (GARealGenome &)c;

 if (anumber<=( popsize*perc )){// Some percentage use 2sls starting values
   for(int i=genome.length()-1; i>=0; i--){
    if (i>=(cyclenb+cyclenf)){
        genome.gene(i,  ( emodelxforstartingvalues[i-(cyclenb+cyclenf)]*changestartingvalue));
    }
    else {
        if (i>=cyclenb){
         genome.gene(i,  (fmodelxforstartingvalues[i-cyclenb]*changestartingvalue));
        }
        else{

         while ( bmodeli[bidx]== bmodelj[bidx]){
           bidx--;
         }
         genome.gene(i,(-bmodelxforstartingvalues[bidx]*changestartingvalue));
         bidx--;
        }
    }
   }

 } else if (anumber<( 2*popsize*perc )){

   for(int i=genome.length()-1; i>=0; i--){
    if (i>=(cyclenb+cyclenf)){
        if ( anumber<( 0.5*popsize*perc)&& (targetforstartingvalues== emodeli[i-(cyclenb+cyclenf)])  ){
         genome.gene(i,  ( emodelxforstartingvalues[i-(cyclenb+cyclenf)] *changestartingvalue));
        }else{
         genome.gene(i,  ( emodelx[i-(cyclenb+cyclenf)]*changestartingvalue));
        }
    }
    else {
        if (i>=cyclenb){
         if ( anumber<( 1.5*popsize*perc)&& (targetforstartingvalues== fmodeli[i-cyclenb])  ){
           genome.gene(i,  ( fmodelxforstartingvalues[i-cyclenb]*changestartingvalue));
         }else{
           genome.gene(i,  ( fmodelx[i-cyclenb]*changestartingvalue));
         }
        }
        else{
         while ( bmodeli[bidx]== bmodelj[bidx]){
           bidx--;
         }

         if ( anumber<( 1.5*popsize*perc)&& (targetforstartingvalues== bmodeli[bidx])  ){
           genome.gene(i,(-bmodelxforstartingvalues[bidx]*changestartingvalue));
         }else{
           genome.gene(i,(-bmodelx[bidx]*changestartingvalue));
         }
```

```
                bidx--;
              }
        }
      }

    }  else{
      for(int i=genome.length()-1; i>=0; i--){
        genome.gene(i,genome.alleleset(i).allele());
      }
    }
  }
```

AUTHORS' CONTRIBUTIONS

Bing Liu worked on the maximum likelihood estimation with genetic algorithms, network topology search, carried out the analysis and drafted the manuscript. Dr. Alberto de la Fuente worked on two stage least squares regression, identification of cycle components, path identification, and simulated data used in this study. Dr. Ina Hoeschele directed the work.

REFERENCES

ANDERSSON, S. A., D. MADIGAN and M. D. PERLMAN;, 1997 A characterization of Markov equivalence classes for acyclic digraphs. Ann. Statist. **25:** 505-541.

BASTEN, C. J., B. S. WEIR and Z. B. ZENG, 1996 *QTL Cartographer: A Reference Manual and Tutorial for QTL Mapping*. Department of Statistics, North Carolina State University, Raleigh, NC.

BEKKER, P. A., A. MERCKENS and T. J. WANSBEEK, 1994 *Identification, equivalent Models, and computer algebra*. Academic Press, San Diego, CA.

BOLLEN, K., 1989 *Structural equations with latent variables*. Wiley-Interscience.

BRAZHNIK, P., A. DE LA FUENTE and P. MENDES, 2002 Gene networks: how to put the function in genomics. Trends Biotechnol. **20:** 467-472.

BREM, R. B., and L. KRUGLYAK, 2005 The landscape of genetic complexity across 5,700 gene expression traits in yeast. Proc. Natl. Acad. Sci. USA **102:** 1572-1577.

BROMAN, K. W., and T. P. SPEED, 2002 A model selection approach for the identification of quantitative trait loci in experimental crosses. J. Roy. Stat. Soc. B **64:** 641-656.

CHICKERING, D. M., 2002a Learning equivalence classes of bayesian-network structures. J. Mach. Learn. Res. **2:** 445 - 498.

CHICKERING, D. M., 2002b Optimal structure identification with greedy search. J. Mach. Learn. Res. **3:** 507-554.

CHIPMAN, H., I. E. EDWARDS and R. E. MCCULLOCH, 2001 The practical implementation of Bayesian model selection, pp. 65-116 in *Model Selection*, edited by P. LAHIRI. Institute of Mathematical Statistics, Beachwood, OH.

DAVIS, T. A., 2004a Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method. ACM Trans. Math. Soft. **30:** 196-199.

DAVIS, T. A., 2004b A column pre-ordering strategy for the unsymmetric-pattern multifrontal method. ACM Trans. Math. Soft. **30:** 165-195.

DAVIS, T. A., and I. S. DUFF, 1997 An unsymmetric-pattern multifrontal method for sparse LU factorization. SIAM J. Matrix Anal. Appl. **18:** 140-158.

DAVIS, T. A., and I. S. DUFF, 1999 A combined unifrontal/multifrontal method for unsymmetric sparse matrices. ACM Trans. Math. Soft. **25:** 1-19.

FISHER, F. M., 1970 A correspondence principle for simultanious equation models. Econometrica **38:** 73-92.

FRIEDMAN, N., M. LINIAL, I. NACHMAN and D. PE'ER, 2000 Using Bayesian networks to analyze expression data. J. Comp. Biol. **7:** 601-620.

GOLDBERG, D. E., 1989 *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, Reading, Mass.

GOLDBERGER, A. S., 1991 *A Course in Econometrics*. Harvard University Press, Cambridge, MA.

HARTEMINK, A., D. GIFFORD, T. JAAKKOLA and R. YOUNG, 2002 Combining location and expression data for principled discovery of genetic regulatory network models, pp. 437-449 in *Pac. Symp. Biocomput.*

HEISE, D. R., 1975 *Causal analysis*. John Wiley and Sons, New York.

HOLLAND, J. H., 1975 *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor.

HOLLAND, J. H., 1992 *Adaptation in natural and artificial systems: an introductory analysi with applications to biology, Control, and Artificial Intelligence* The MIT Press, Cambridge, Massachusetts, London.

IMOTO, S., K. SUNYONG, T. GOTO, S. ABURATANI, K. TASHIRO *et al.*, 2002 Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network, pp. 219-227 in *Proc. IEEE Comput. Soc. Bioinform. Conf.*

JOHNSTON, J., 1972 *Econometric methods*. McGraw-Hill, St. Louis.

JÖRESKOG, K. G., and D. SÖRBOM, 1989 *LISREL 7: A guide to the program and applications, 2nd Edn.* SPSS Inc.

JUDGE, G. G., W. E. GRIFFITHS, R. C. HILL, H. LÜTKEPOHL and T. C. LEE, 1985 *The Theory and Practice of Econometrics*. Wiley, New York.

LIU, B., A. DE LA FUENTE and I. HOESCHELE, 2006, From genetics to gene networks: Evaluating approaches for integrative analysis of genetic marker and gene expression data for the purpose of gene network inference

MADIGAN, D., and A. E. RAFTERY, 1994 Model Selection and Accounting for Model Uncertainty in Graphical Models Using Occam's Window. J. Am. Stat. Assoc. **89:** 1535-1546.

MENDES, P., 1993 GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems. Comput. Appl. Biosci. **9:** 563-571.

MENDES, P., 1997 Biochemistry by numbers: simulation of biochemical pathways with Gepasi 3. Trends Biochem. Sci. **22:** 361-363.

MENDES, P., 2001 Modeling large scale biological systems from functional genomic data: parameter estimation, pp. 163-186 in *Foundations of Systems Biology*, edited by H. KITANO. MIT Press, Cambridge, MA.

MENDES, P., W. SHA and K. YE, 2003 Artificial gene networks for objective comparison of analysis algorithms. Bioinformatics **19:** Suppl 2:II122-II129.

MOLES, C. G., P. MENDES and J. R. BANGA, 2003 Parameter Estimation in Biochemical Pathways: A Comparison of Global Optimization Methods. Genome Res. **13:** 2467-2474.

MURPHY, K., and S. MIAN, 1999 *Modelling gene expression data using dynamic Bayesian networks*. Technical report, Computer Science Division, University of California, Berkeley, CA.

NEALE, M. C., S. M. BOKER, G. XIE and H. H. MAES, 2003 *Mx: Statistical Modeling*. Department of Psychiatry, Richmond, VA.

PE'ER, D., A. REGEV, G. ELIDAN and N. FRIEDMAN, 2001 Inferring subnetworks from perturbed expression profiles. Bioinformatics **17:** 215-224.

PEARL, J., 2000 *Causality : Models, Reasoning, and Inference*. Cambridge University Press.

RICHARDSON, T., 1996 A Polynomial-Time Algorithm for Deciding Markov Equivalence of Directed Cyclic Graphical Models, pp. in *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, edited by E. HORVITZ and F. JENSEN, Portland, Oregon.

RICHARDSON, T., and P. SPIRTES, 1999 Automated discovery of linear feedback models, pp. 253-304 in *Computation, Causation, and Discovery*, edited by C. GLYMOUR and G. F. COOPER. MIT Press, Cambridge, MA.

SCHWARTZ, G., 1978 Estimating the dimension of a model. Ann. Stat. **6:** 461-464.

SHANNON, P., A. MARKIEL, O. OZIER, N. S. BALIGA, J. T. WANG *et al.*, 2003 Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. Genome Research **13:** 2498-2504.

SHIPLEY, B., 2002 *Cause and Correlation in Biology: A User's Guide to Path Analysis, Structural Equations and Causal Inference*. Cambridge University Press.

SPIRTES, P., C. GLYMOUR, R. SCHEINES, S. KAUFFMAN, V. AIMALE *et al.*, 2000 Constructing Bayesian Network Models of Gene Expression Networks from Microarray Data, in *Proc. Atlantic Symp. Comp. Biol., and Genome Inf. Syst. and Technol.*

VERMA, T., and J. PEARL, 1991 Equivalence and synthesis of causal models, pp. in *Proc. of the 6th workshop on uncertainty in Artificial Intelligence*, Cambridge, MA.

XIONG, M., J. LI and X. FANG, 2004 Identification of genetic networks. Genetics **166:** 1037-1052.

YOO, C., V. THORSSON and G. COOPER, 2002 Discovery of causal relationships in a gene-regulation pathway from a mixture of experimental and observational DNA microarray data, pp. 498-509 in *Pac. Symp. Biocomput.*

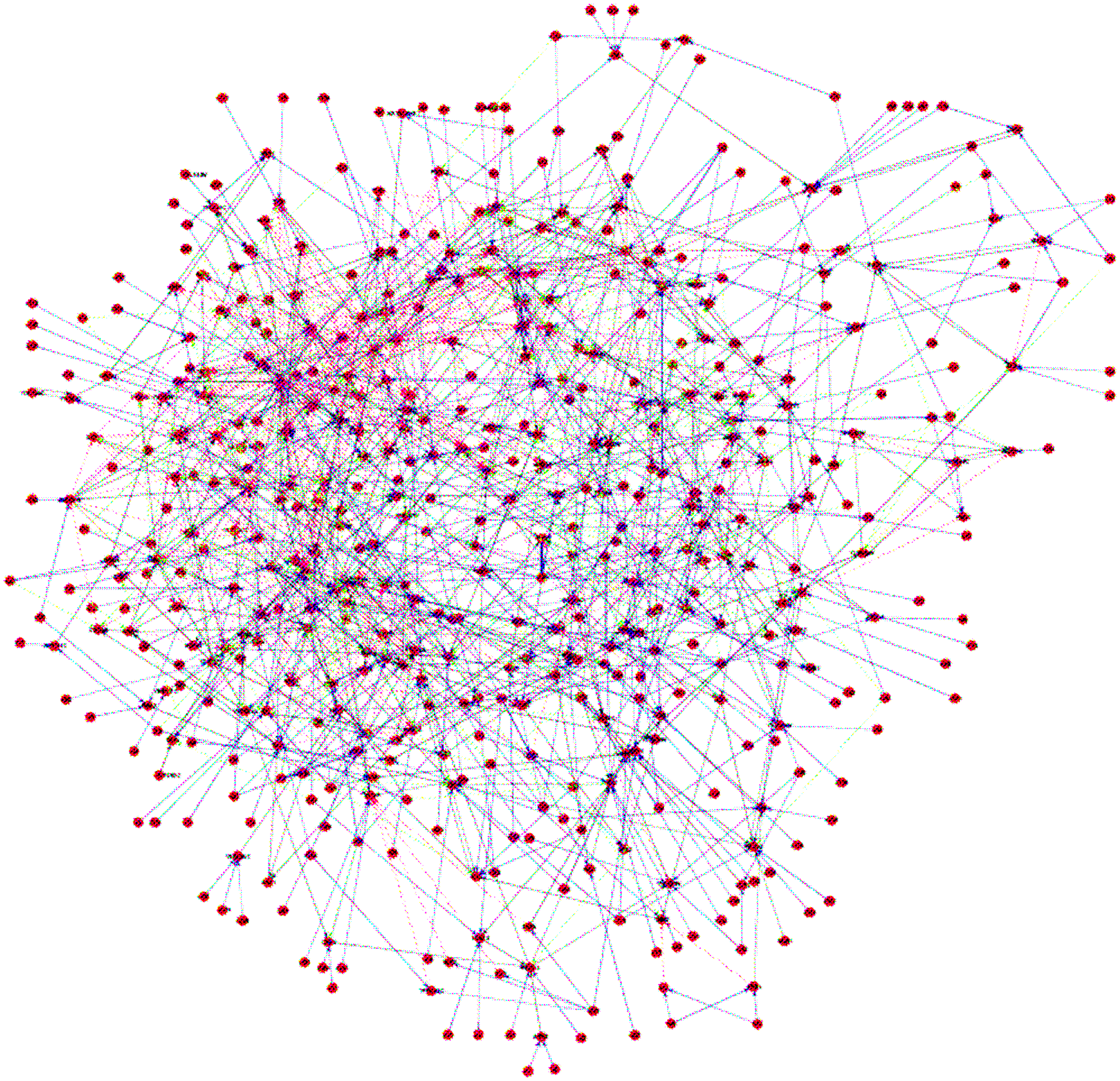


FIGURE 4.1.— Network topology of the yeast sub network

Produced with Cytoscape (SHANNON *et al.* 2003). Black edges are gene → gene edges in the sparsified network, and the blue edges are QTL → gene edges in the sparsified network. Red dotted edges are removed gene → gene edges, and the green doted edges are removed QTL → gene edges.

TABLE 4.1 Results of the SEM analysis on the simulated data

IC	Edge type	Model Measure	1	2	3	4	5	6	7	8	9	10
BIC	F	FDR	18.4	24.3	27.4	17.9	19.5	21.6	20.7	19.0	23.9	22.2
		Power	100.0	100.0	100.0	100.0	100.0	99.2	99.2	100.0	97.5	100.0
	B	FDR	6.6	7.1	7.6	7.6	5.7	8.5	3.8	15.3	9.5	11.0
		Power	87.6	89.7	89.9	89.3	89.3	88.4	85.9	85.8	88.7	87.2
BIC(δ)	F	FDR	7.5	7.9	7.7	5.1	8.1	7.1	6.3	14.8	11.9	14.5
		Power	100.0	100.0	99.2	99.2	100.0	96.7	100.0	98.4	100.0	100.0
	B	FDR	0.8	0.0	1.7	0.0	1.6	3.4	0.0	3.4	1.8	0.9
		Power	80.7	82.2	79.9	78.5	81.2	76.2	77.9	77.7	72.7	71.8

Percentages of FDR and Power are given for the ten models using BIC and BIC(δ) criteria, and for the QTL and gene edges.

TABLE 4.2 Significant biological function groups of genes in the yeast sub network

GO_term	Frequency	Genome Frequency	Probability	Genes
catalytic activity	41.6%	26.8%	1.50E-07	AAD14 AAD6 ACO1 AKL1 ALD6 AMD2 APN2 ARA1 ARD1 ARP5 AYR1 BDS1 CIT2 COQ5 COX5B DCP2 DIA4 DLD3 DUS3 ECM40 ERF2 EXG1 FET3 FET5 FRE2 GAB1 GCV3 GPA1 GRX5 HIS4 HIS5 HMG1 HMG2 HO HOS4 ICL2 ILV6 KCC4 KTR1 KTR6 LAT1 LEU2 LSC1 LYS2 LYS4 MAP1 MCM6 MET22 MKT1 MSH2 MSK1 MTQ2 MTR3 NFS1 NOP2 NUC1 NUG1 OST2 OST6 PDE1 PDR12 PHO8 PHO85 PLB2 PMA2 POL1 PPZ1 RAD16 RAD52 RAS1 RCK2 RFC4 RFC5 RHO2 RIB3 RPE1 RPM2 RPO41 SAP4 SCO1 SEN1 SHR5 SKM1 PAH1/SMP2 SPO11 SSA4 SUR1 THR4 TIP1 TOP2 TPS1 TRM7 TRP3 TYR1 TYS1 UBP14 UBP16 UGA2 URA3 WRS1 YAL061W RXT2 YEL077C YER138C YER160C YNL045W NMA111 YOL155C YPT53 YPT6
hydrolase activity	17.8%	10.5%	0.00026	AMD2 APN2 ARP5 BDS1 DCP2 EXG1 GAB1 GPA1 HIS4 HO HOS4 MAP1 MCM6 MET22 MKT1 MSH2 MTR3 NUC1 NUG1 PDE1 PDR12 PHO8 PLB2 PMA2 PPZ1 RAD16 RAS1 RFC4 RFC5 RHO2 RPM2 SAP4 SEN1 PAH1/SMP2 SPO11 SSA4 TIP1 UBP14 UBP16 RXT2 YER138C YER160C YNL045W NMA111 YOL155C YPT53 YPT6
transporter activity	9.0%	5.6%	0.01485	AAC1 AGP2 ALR1 AQR1 ATO2 ATR1 COX5B CRC1 DIC1 HXT2 ITR1 KAP114 LPE10 MCH4 MRS11 PDR12 PHO91 PMA2 POR1 SAL1 TAT1 UGA4 YFL054C YMC2
oxidoreductase activity	7.9%	3.5%	0.00066	AAD14 AAD6 ALD6 ARA1 AYR1 COX5B DLD3 FET3 FET5 FRE2 GCV3 GRX5 HIS4 HMG1 HMG2 LEU2 LYS2 SCO1 TYR1 UGA2 YAL061W
pyrophosphatase activity	6.8%	3.5%	0.00615	ARP5 DCP2 GPA1 HIS4 MCM6 MSH2 NUG1 PDR12 PMA2 RAD16 RAS1 RFC4 RFC5 RHO2 SEN1 SSA4 YPT53 YPT6
nucleoside-triphosphatase activity	6.0%	3.2%	0.01405	ARP5 GPA1 MCM6 MSH2 NUG1 PDR12 PMA2 RAD16 RAS1 RFC4 RFC5 RHO2 SEN1 SSA4 YPT53 YPT6

Obtained from the Saccharomyces genome database http://www.yeastgenome.org/. The columns are: significant GO terms; frequency of the terms in genes submitted; frequency of the terms in the whole genome; a score of significance of the terms in the genes submitted; genes involved in the biological process.

Chapter 5

Summary and future research

Gene network construction is an extremely complex task and probably never-ending research area. To be able to construct causal cyclic networks, we apply expression Quantitative Trait Locus (eQTL) analysis and Structural Equation Modeling (SEM) for the reconstruction of causal gene networks for genetical genomics experiments. Our network construction method exhibited very promising results, and should be further improved in terms of performance and scalability.

For large genetical genomics experiments, computation cost is a critical issue. Linear Mixed Model Analysis (LMMA) of large multifactorial experiments can be computationally very intensive. Efficient algorithm for LMMA would be necessary for such large genetical genomics experiments. Sample size calculations with the eQTL mapping methods should be performed as in Kim et al. (2005) to ensure sufficient power while containing the large expense of these experiments.

Since the PC-mapping exhibited very high power, combining PC-mapping with cis and trans-mapping to detect pleiotropic eQTLs should be explored. Combining cis-mapping and trans-mapping of individual etraits and PCs may be the best approach to EDN construction.

The regulator-target pair identification for EDN construction can be further improved. Without taking the whole network into account, there is only limited information available for this purpose. However, some methods that utilize information from neighboring genes or from another eQTL mapping method would benefit the regulator-target pair identification. Between-strain SNP information, or information on protein-protein interactions, will be helpful in identifying the regulators if available.

Maximum Likelihood is the predominant full-information method for parameter inference in structural equation models. It is therefore natural to perform a model search based on an information criterion that is a function of the maximized likelihoods of two competing models. While BIC and BIC(δ) performed satisfactorily in this study, further research on developing appropriate model selection criteria for large, very sparse networks is required.

How to efficiently update the network topologies during SEM model search needs more attention. The constructed EDN may include several candidate regulators for one eQTL, and they may or may not all be real regulators. Before a global search, searching among multiple candidate regulators for one eQTL can be an option. Then, an eQTL and its candidate regulator(s) can be updated jointly. Or, a local sparsification for each target may be performed before the global search. A model similar to the one used in the regulator-target pair identification for the trans-mapping, while including all regulators and eQTLs (not tightly linked) may be used. In addition, the eQTL analysis can suggest the sequence of edge deletion. For example, possible indirect effects may be tested first.

Our SEM model can be generalized to include certain types of interactions: those between an eQTL and a regulator gene jointly trans-regulating a target gene, and epistatic interactions between eQTL found in the eQTL analysis and hence included in the EDN. Furthermore, in this study we have considered a network with only causal, directed interactions or regulations. However, two genes may be correlated, but there may be no eQTL information available to determine causation. Such associations or undirected edges may be incorporated via correlations in the residual covariance matrix ***E***. One can then include these off-diagonal elements in ***E*** in the EDN and consider them as potentially present in the model search.

Lastly, besides the maximum likelihood implementation of SEM, a Markov chain Monte Carlo method for Bayesian SEM with prior information can be developed. The Bayesian approach can incorporate prior information about the network, and can account for the uncertainties in a complex problem.

REFERENCES

KIM, H. Y., J. M. WILLIAMSON and C. M. LYLES, 2005 Sample-size calculations for studies with correlated ordinal outcomes. Stat. Med. **24:** 2977-2987.

Vita

Bing Liu, daughter of Zhenhua Liu and Qinying Chen, was born in Hubei, China, in 1977. She received her Bachelor of Science in Biology from the Huazhong Normal University, China, in 1998. After that, she continued her graduate study in Plant Physiology in Zhejiang University, where she met and married her husband Xiaobo Zhou.

In 2000, Mrs. Liu started her study in Virginia Polytechnic Institute and State University (Virginia Tech), and received her first M.S. degree in Accounting and Information systems in 2002. She then joined Dr. Ina Hoeschele's Statistical Genetic group at Virginia Tech in 2002 as a graduate research assistant. She received her second M.S. degree in Statistics in 2004. Her current research interests are gene expression analysis, genetical genomics, and gene network inference.