Reverse Engineering of Biological Systems

A Thesis Submitted to the
College of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the degree of Doctor of Philosophy
in the Department of Mechanical Engineering
University of Saskatchewan
Saskatoon

By
Lizhi Liu

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Mechanical Engineering
University of Saskatchewan
57 Campus Drive
Saskatoon, Saskatchewan
Canada S7N 5A9

Abstract

Gene regulatory network (GRN) consists of a set of genes and regulatory relationships between the genes. As outputs of the GRN, gene expression data contain important information that can be used to reconstruct the GRN to a certain degree. However, the reverse engineer of GRNs from gene expression data is a challenging problem in systems biology. Conventional methods fail in inferring GRNs from gene expression data because of the relative less number of observations compared with the large number of the genes. The inherent noises in the data make the inference accuracy relatively low and the combinatorial explosion nature of the problem makes the inference task extremely difficult. This study aims at reconstructing the GRNs from time-course gene expression data based on GRN models using system identification and parameter estimation methods. The main content consists of three parts: (1) a review of the methods for reverse engineering of GRNs, (2) reverse engineering of GRNs based on linear models and (3) reverse engineering of GRNs based on a nonlinear model, specifically S-systems.

In the first part, after the necessary background and challenges of the problem are introduced, various methods for the inference of GRNs are comprehensively reviewed from two aspects: models and inference algorithms. The advantages and disadvantages of each method are discussed.

The second part focus on inferring GRNs from time-course gene expression data based on linear models. First, the statistical properties of two sparse penalties, adaptive LASSO and SCAD, with an autoregressive model are studied. It shows that the proposed methods using these two penalties can asymptotically reconstruct the underlying networks. This provides a solid foundation for these methods and their extensions. Second, the integration of multiple datasets should be able to improve the accuracy of the GRN inference. A novel method, Huber group LASSO, is developed to infer GRNs from multiple time-course data, which is also robust to large noises and outliers that the data may contain. An efficient algorithm is also developed and its convergence analysis is provided.

The third part can be further divided into two phases: estimating the parameters of S-systems with system structure known and inferring the S-systems without knowing the system structure. Two methods, alternating weighted least squares (AWLS) and auxiliary function guided coordinate descent (AFGCD), have been developed to estimate the parameters of S-systems from time-course data. AWLS takes advantage of the special structure of S-systems and significantly outperforms one existing method, alternating regression (AR). AFGCD uses the auxiliary function and coordinate descent techniques to get the smart and efficient iteration formula and its convergence is theoretically guaranteed. Without knowing the system structure, taking advantage of the special structure of the S-system model, a novel method, pruning separable parameter estimation algorithm (PSPEA) is developed to locally infer the S-systems. PSPEA is then combined with continuous genetic algorithm (CGA) to form a hybrid algorithm which can globally reconstruct the S-systems.

Acknowledgements

First and foremost, I owe innumerable thanks to my PhD supervisors, Prof. Fang-Xiang Wu and Prof. Wen-Jun Zhang for being great mentors, both professionally and personally. This thesis would never be possible without their constant encouragement and support. Their valuable and insightful advice not only encouraged me to keep learning and thinking, but also guide me to be an independent researcher. I am in particular indebted to them for their patience and allowing me with enough freedom to explore broad research topics.

I would also like to express thanks to other members of my advisory committee, Prof. Richard Burton, Prof. Daniel Chen and Prof. Sherif Faried, for their assistance and great advice during the PhD program.

My thanks also go to our group members, Bolin Chen, Yan Yan, Lin Wu, Yichao Shen, Amin Mbagheri, Wenjun Lin, Vivian Fan and Jinhong Shi, for their help both in my life and research.

I would also like to thank all my families for their continuous support and love.

Finally, I gratefully acknowledge the Natural Sciences and Engineering Research Council of Canada (NSERC) and College of Engineering at the University of Saskatchewan for the financial supports.

Dedicated to my parents,

Juqun Liu and Yuzhen Wang,

who give me their genes,

And my fiancée,

Hengyao Lu,

Without whom none of my success would be possible.

Contents

List of Tables

List of Figures

List of Abbreviations

DNA	Deoxyribonucleic acid
cDNA	Complementary DNA
RNA	Ribonucleic acid
mRNA	Messenger RNA
TF	Transcription Factor
RNAP	RNA Polymerase
ChIp	Chromatin Immunoprecipitation
PCR	Polymerase Chain Reaction
PPI	Protein Protein Interaction
GRN	Gene Regulatory Network
TRN	Transcription Regulatory Network
PWM	Position Weight Matrix
GWAS	Genome-Wide Association Study
eQTL	expression quantitative trait loci
SMD	Stanford Microarray Database
BST	Biochemical System Theory
PBN	Probability Boolean Network
DBN	Dynamic Bayesian Network
ODE	Ordinary Differential Equation
GGM	Gaussian Graphical Model
RNN	Recurrent Neural Network
DAG	Directed Acyclic Graph
SVD	Singular Value Decomposition
GA	Genetic Algorithm
CGA	Continuous GA
PSO	Particle Swarm Optimization
VAR	Vector Autoregressive
MCMC	Markov Chain Monte Carlo
AIC	Akaike Information Criterion
BIC	Bayesian Information Criterion
BNRC	Bayesian Nonparametric Regression Criterion
PC	Path Consistency
DREAM	Dialogue for Reverse Engineering Assessment and Methods
LASSO	Least Absolute Shrinkage and Selection Operator
SCAD	Smoothly Clipped Absolute Deviation
LQA	Local Quadratic Approximation
AR	Alternating Regression
AWLS	Alternating Weighted Least Squares
PLS	Partial Least Squares
SPEM	Separable Parameter Estimation Method
PSPEA	Pruning Separable Parameter Estimation Algorithm
AFGCD	Auxiliary Function Guided Coordinate Descent
ROC	Receiver Operating Characteristic
AUROC	Area Under ROC
AUPR	Area Under Precision Recall
FPR	False Positive Rate
TPR	True Positive Rate

Chapter 1

Introduction

1.1 Background

A biological system is composed of entities that can interact with each other. The system structure can be depicted by a graph in which nodes are entities and edges exist between interacting entities. Many mathematical models have been developed to describe various biological systems. Those models characterize the system structure as well as interactions between entities and are important tools for the study of biological systems. Most of them are parametric models, i.e., containing parameters whose values are unknown but have significant biological meanings, e.g., reaction rates, kinetic orders, etc. Generally, the values of the parameters cannot be obtained directly from experiments. With the development of the biological techniques, a large amount of various types of biological data are available. These data can be considered as the outputs of the biological systems and contain important biological information. To extract the information, structure identification and parameter estimation methods can be used to infer the structures and estimate the parameters of the models from the data.

In this thesis, the biological systems we focus on are gene regulatory networks (GRNs) and the data considered are gene expression data. Our purpose is to explore the information of the GRN, specifically, network topology and regulatory strengths between interacting genes, by fitting the gene expression data to the GRN models [2] via system identification and parameter estimation methods. In the following, the background of GRNs and gene expression data are introduced.

Molecules in the cell interact with each other and form a complex system to realize biological functions. DNA, RNA and protein are three most important molecules in the cell and their relationships can be summarized as the central "dogma" of molecular biology illustrated in Figure 1.1: DNA can copy itself and produce two identical copies of DNA in the replication process. DNA contains the information that can be used to produce RNA and proteins. In the transcription process, the coded information in stretches of DNA is transcribed to messenger RNAs (mRNAs). In the translation process, mRNA is further translated into a chain of polypeptides which is folded and modified into a biochemically active protein. Proteins are the "workhorse" of life, playing essential roles in every structure and activity of life. The DNA stretches that contain the necessary information to code for functional proteins are called genes. The process that transforms coded information in genes into functional proteins is known as gene expression. The expression

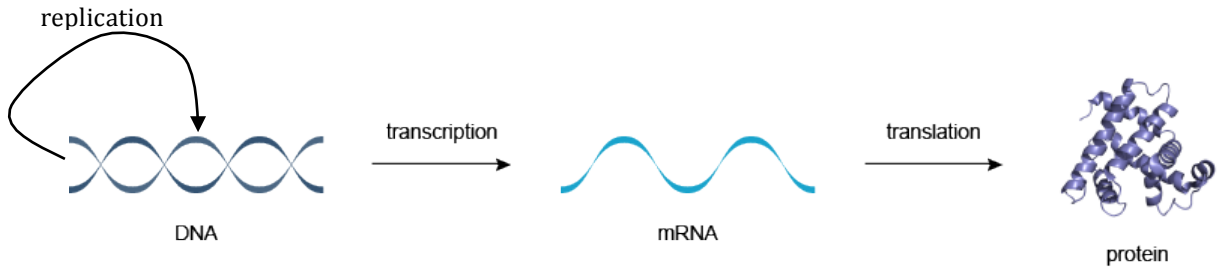


Figure 1.1: Central "dogma" of molecular biology.

of a specific gene may be regulated by other genes in the way that the proteins, also known as transcription factors (TFs), or other molecules produced from the regulatory genes bind the promoter region of the target gene and act positively to activate or negatively to repress its transcription. The way regulatory genes exert effects on the target gene via TFs or other molecules is called gene regulation. Each gene may be regulated by other genes and all the genes in the cell and their regulatory relationships make a complex system to realize the biological functions of the cell. A set of genes and their regulatory relationships is called a GRN [3], which can be obtained by mapping all the relations among various molecules onto the gene space as shown in Figure 1.2. GRN is a high level description of the molecular interactions in a cell and it has the ability to capture the characteristics of the system since all the interactions are abstracted as relations between genes.

Genes produce mRNAs under the regulations of other genes in the transcription. Therefore, the amount of mRNAs can reflect the activities of the genes and also implicate the regulatory information. Recently developed DNA microarray technology [4, 5, 6], also known as DNA chip or gene array, enables researchers to measure the mRNA expression levels of thousands of genes simultaneously. A DNA micorarray is a collection of microscopic DNA spots attached to a solid surface. Each spot represents a single gene and contains probes consisting of known sequences of the gene. The scheme of DNA microarray experiments is illustrated in Figure 1.3. First, after polymerase chain reaction (PCR) and purification, the templates of genes of interest are printed onto the microarray slide by a computer controlled high-speed robot. Complementary DNAs (cDNAs) labeled with two different fluorophores (cye3, cye5) are synthesized from mRNAs of the test (with cye3) and reference sample (with cye5) through the reverse transcription process. These two samples are hybridized to a single microarray which is then scanned to visualize the fluorescence of the two fluorophores excitated by lasers of different wavelengths. The scanned images of the microarray under two different lasers are merged in the computer software and then the gene expression data are obtained.

The data from a single microarray experiment are viewed as a normalized ratio (cye3 / cye5) which indicates increased (cye3 / cye5 > 1), decreased (cye3 / cye5 < 1) or unchanged (cye3 / cye5 ≈ 1) levels of gene expression relative to the reference sample. Data from multiple microarray experiments can be stored in the form of a matrix with rows representing genes and columns representing the experiments as shown in Figure 1.4. According to the ways of doing experiments, there are two types of gene expression data: static and time-course. For static data, the gene expressions are measured on different samples and the experiments

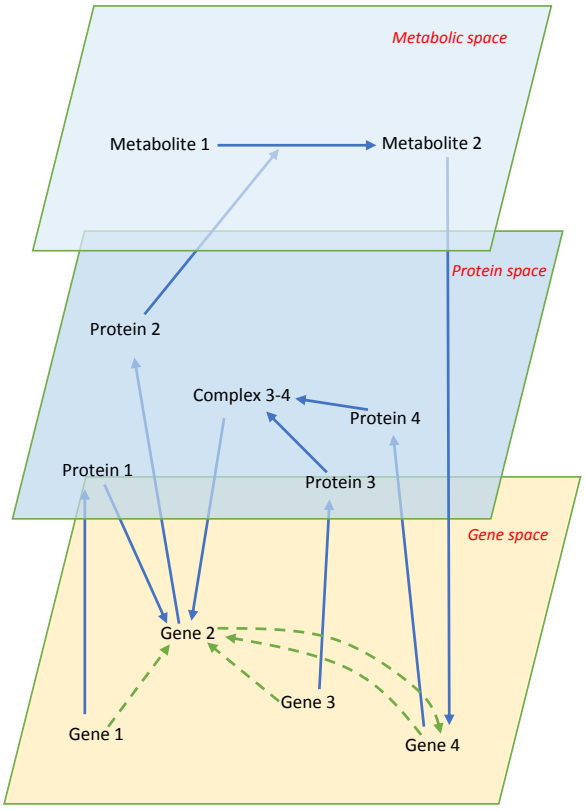


Figure 1.2: Genetic interactions in GRNs.

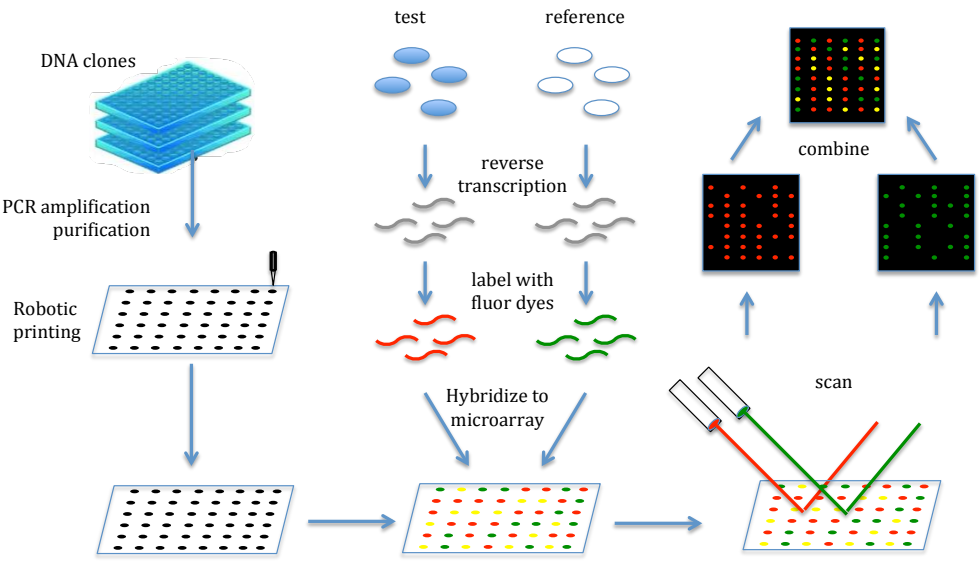


Figure 1.3: DNA microarray experiments.

genes	Microarray experiments			
	sample 1	sample 2	...	sample *n*
gene 1	x_{11}	x_{12}	...	x_{1n}
gene 2	x_{21}	x_{22}	...	x_{2n}
...	...	...	...	...
gene *p*	x_{p1}	x_{p2}	...	x_{pn}

(a)

genes	Microarray experiments			
	time 1	time 2	...	time *n*
gene 1	x_{11}	x_{12}	...	x_{1n}
gene 2	x_{21}	x_{22}	...	x_{2n}
...	...	...	...	...
gene *p*	x_{p1}	x_{p2}	...	x_{pn}

(b)

Figure 1.4: An illustration of gene expression data. (a) Static gene expression data. (b) Time-course gene expression data.

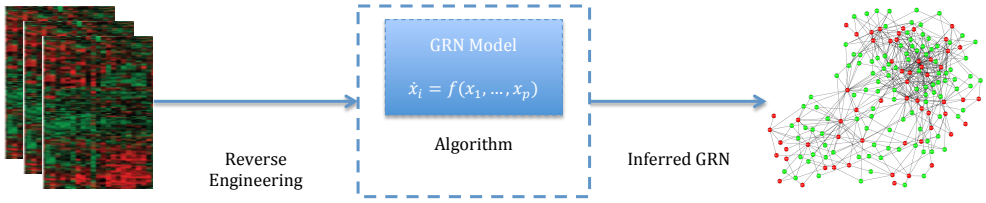


Figure 1.5: General scheme of the reverse engineering of GRNs.

are independent. For time-course data, the gene expressions are measured at consecutive time points, and the experiments are serially correlated [7]. Time-course gene expression data capture the dynamic information of systems and are good sources for the study of dynamic properties of GRNs. One defect of the gene expression data is the relatively small number of observations (i.e., the number of experiments) compared with the huge number of genes. This is mainly due to the high cost of the microarray experiment. The other defect is that the data contain a lot of noises which may be either introduced in the process of experiments or directly generated from the experiment devises.

The topology of the GRN can be depicted as a graph with nodes representing genes and edges representing regulatory relations. The research of this thesis aims at finding the GRN topology and interactions between genes from gene expression data, formally known as reverse engineering of the GRN [8, 9, 2]. The general idea, shown in Figure 1.5, is to use the GRN model as a template and develop algorithms or methods to fit the gene expression data to the template by system identification and parameter estimation methods. Then, the inferred GRN is obtained from the fitted model.

1.2 Motivations and Objectives

The GRN is a blueprint showing the way genes interact for making a living system. The network topology and interaction strengths between genes are quite difficult to obtain directly from experiments due to the

huge number of possible network connections, i.e., for p genes, the number of possible combination regulatory genes for each target gene is 2^p which is quite large even for a small-sized network.

As outputs of the GRNs, the gene expression data contain important network information from which the GRNs could be recovered to some extent. Reverse engineering of the GRNs is the basis for studying the properties of GRNs, e.g. controbility[10] and observability [11], and some practical applications, e.g. disease treatment and drug design.

The reverse engineering of GRNs from gene expression data is a challenging problem in systems biology. First, as mentioned above, for a specific target gene, finding its regulatory genes is a combinatorial explosion problem. The developed algorithm should be efficient and able to overcome the high dimensionality of the searching space. Second, the limited information contained in the gene expression data make the inference task difficult. To infer the relations between the variables (i.e., genes), conventional methods require the number of observations is no less than the number of variables. However, gene expression data tend to have relatively less observations compared with the huge number of genes. This prevents the direct application of many conventional methods and this is known as dimensionality problem or "large p small n" problem [12, 13, 14]. Gene expression data may contain large noises or outliers which affect the accuracy of the inferred network. Therefore, inference methods that are robust to the large noises and outliers are required. Third, the difficulty may also comes from the model structure. In this thesis, two types of models are considered: linear and nonlinear. For linear model, although the design of the inference algorithm is relatively easy, it can only approximately reconstruct the GRNs to some extent owning to nonlinearity of the real system. Nonlinear models can effectively capture the nonlinear behaviors of the GRNs and therefore are good candidates for the reverse engineering of the GRNs. However, their structure identification and parameter estimation are quite difficult due to their nonlinear and complex mathematical representations.

Based on these motivations, the objectives of this thesis are

1) Reviewing currently developed methods for reverse engineering of GRNs and discussing their advantages and disadvantages.

2) Studying the properties of GRN inference methods based on linear models and developing a method to reconstruct the network topology based on linear models with the consideration of improving the accuracy and robustness to large errors.

3) Developing methods for identifying the structure and estimating the parameters of the nonlinear GRN model. The nonlinear model studied in this thesis is S-system, which is an effective nonlinear ODE model for the GRN.

All objectives are achieved in the following chapters.

1.3 Organization of the Thesis

This thesis is organized in a manuscript-based style. Our works to achieve the objectives constitute the main content of this thesis. They are presented in the form of published or submitted manuscripts. In each chapter, a brief introduction is included to describe the connection of the manuscript to the context of the thesis. In addition, a general overview of the links of each manuscript to the thesis as a whole is provided in Chapter 8. The paper manuscripts have been formatted to be consistent with the requirements of the thesis.

The remainder of the thesis is organized as follows: Chapter 2 presents a comprehensive review of the methods for reverse engineering of GRNs. The models those methods are based on and the algorithms they use are surveyed respectively. The advantages and disadvantages of each methods are also discussed. As GRNs exhibit low connectivity, sparse penalties, e.g. LASSO [15], are usually used to obtain sparse GRNs. Chapter 3 studies the statistical properties of two famous penalties, adaptive LASSO [16] and SCAD [17], when based on a linear auto-regressive model and inferring GRNs from time-course gene expression data. It shows that under this setting, these two penalties enjoy the "Oracle" properties, i.e., can asymptotically reveal the network topology and the regulation strengths between genes. Chapter 4 presents a novel method, Huber group LASSO, to infer the network topology from multiple time-course gene expression data as well as to take the robustness to large errors or outliers into account. An efficient algorithm is developed and its convergence is analyzed. Compared with existing methods, the proposed method improves not only the inference accuracy but also the resistance to large noises. Chapter 5 proposes an alternating weighted least squares to estimate the parameters of S-systems from time-course gene expression data with the known system structure. This method takes advantage of the special mathematical structure of S-systems and reduces the nonlinear optimization problem into a series of weighted least squares problems which have analytical solutions. Chapter 6 develops an auxiliary function guided coordinate descent method to estimate the parameters in S-systems from time-course gene expression data. This method estimates the parameters by cyclically optimizing the parameters in the designed auxiliary function and its convergence is mathematically proved. Chapter 7 develops a pruning separable parameter estimation method which takes the special mathematical form of S-system into account and has the ability to locally reconstruct the S-system from time-course gene expression data with appropriate initial values. Then this method is combined with continuous genetic algorithm to globally infer the S-systems from time-course gene expression data. Finally, Chapter 8 summarizes the conclusions of the work in this thesis and point out some future work along this research. The copyright permissions of the manuscripts included are in Appendix A.

Chapter 2

Reverse Engineering of Gene Regulatory Networks from Biological Data

Published as: L.Z. Liu, F.X. Wu, and W.J. Zhang, "Reverse engineering of gene regulatory networks from biological data," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 5, pp. 365–385, 2012 [18].

This chapter gives a comprehensive review of the methods for reverse engineering of GRNs from biological data. It first introduces the necessary biological background and the overall scheme of GRN inference. Different models those methods are based on and algorithms they use are respectively surveyed. The pros and cons of each method are also discussed in the review. This chapter fulfills Objective 1 of the thesis.

Abstract

Reverse engineering of gene regulatory networks is one of the most challenging tasks in systems biology and bioinformatics. It aims at revealing network topologies and regulation relationships between components from biological data. Owing to the development of biotechnologies, various types of biological data are collected from experiments. With the availability of these data, many methods have been developed to infer gene networks. This paper firstly provides an introduction to the basic biological background and the general idea of gene network inferences. Then different methods are surveyed from two aspects: models that those methods are based on and inference algorithms that those methods use. The advantages and disadvantages of these models and algorithms are discussed.

2.1 Introduction

Biological systems are wondrously complex and involve many uncertainties. With the help of molecular biology, vast array of components in biological systems have been revealed. Now, one of the challenging tasks in molecular biology is to understand how interactions among these puzzle pieces determine the characteristics of organisms. To achieve this goal, a blueprint must be obtained to specify the manner that these components interact for making a whole living system. With the rapid advancements in biological technologies, researchers currently have access to a large amount of various types of data. For example, DNA microarray technology

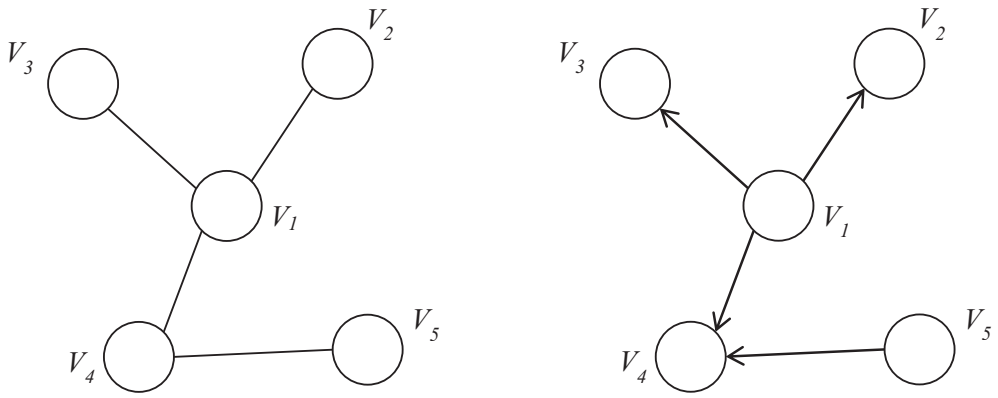


Figure 2.1: Examples of GRNs.

[5, 6] enables experimentalists to simultaneously measure concentrations of thousands of mRNA transcripts. With the availability of these data, many mathematical and computational approaches [7, 8, 9, 2] have been developed to make the goal closer.

Biomolecular networks consist of molecules in cells and their interactions. One important type is the gene regulatory networks (GRNs) [9] that aims at capturing the dependencies between interacting components, such as genes, RNAs, proteins and other various small molecules. Information flows in the network, e.g., from DNA, mRNA to protein. Regulations and controls of gene expressions can occur at different stages of the information transfer and may be caused by various factors [8]. The understanding of GRNs is very important for the understanding of fundamental cellular process.

GRNs are always represented as graphs or networks, in which nodes represent genes, proteins or metabolites and edges represent the relations between the components. Mathematically, a graph is expressed as $G(V, E)$, where V denotes a set of nodes, $V = \{V_1, \ldots, V_p\}$ and $E \subseteq V \times V$ denotes a set of edges. Edges can be undirected to capture the symmetric relations or directed to capture the asymmetric relations. The network edges in different contexts can have vastly different meanings and should be interpreted with care. As an example, Figure 2.1 shows two examples of GRNs: one with directed edges and the other with undirected edges and all the nodes in both GRNs denote genes. An undirected edge may represent that the genes are coexpressed or coregulated or share a common biological function, location or process. On the other hand, an directed edge between two genes may indicate a step in a metabolic pathway, signal transduction cascade or a causal control or regulatory relationship [19].

In the study of GRNs, one essential feature is to treat a GRN as a whole system rather than a collection of independent single cellular entities. Such a systematic view provides an insight into the control and optimization of parts of the system while considering the effects those may have on the whole system [8, 7]. The system-wide view and modeling may lead to valuable clues and new ideas in practical areas such as disease treatment and drug design [20].

Reverse engineering is to obtain the structure of a system by reasoning backwards from the observations of

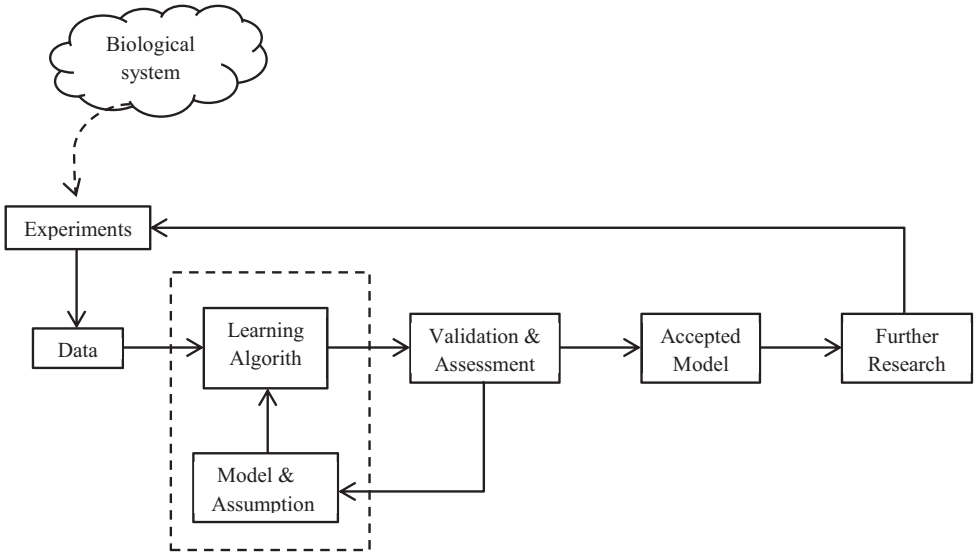


Figure 2.2: The general procedure of inferring GRNs and its role in biological research.

behaviors of the system. Reverse engineering of GRNs based on biological experimental data is a non-trivial problem, also known as network inference, or network identification. Many reverse engineering techniques have already been developed in the fields of computer science, statistics and engineering which are called machine learning, statistical learning and system identification, respectively [8]. Though these methods can be applied or extended to infer GRNs, reverse engineering of GRNs is still a challenging and active area of research. Challenges mainly come from the nature of data and the GRN itself [8]: The data are typically noisy, high-dimensional, and significantly under-sampled. The GRN models are usually very complicated, e.g., nonlinear and containing too many parameters. In addition, the lacks of methods or standards that can effectively evaluate the performance of algorithms also make it difficult.

The general procedure of reverse engineering of GRNs and its role in the whole biological research is shown in Figure 2.2. The scientists perform experiments based on hypotheses and experimental designs to obtain meaningful experimental data. Based on prior biological knowledge and some hypotheses, the GRN is described by a proposed general model, which is regarded as a template that assigns biological meanings and system structure to its unknown parameters. With a learning algorithm, a resulting model is obtained by fitting the model into the available data and then it is further validated and assessed. If the model is acceptable, it is applied in further research such as prediction and simulation. Otherwise, the assumptions are modified and an updated model template is proposed. Generally, the inferred models can provide more reasonable hypotheses or clues to guide future experiment designs and researches [8, 9, 7].

This paper provides an overview of reverse engineering methods for GRNs from two aspects: modeling of GRNs and learning algorithms. Further reviews are referred to Bar-Joseph [7] which discusses the analysis of gene expression time series data, Huang et al. [9] which is about the statistical methods of modeling GRNs,

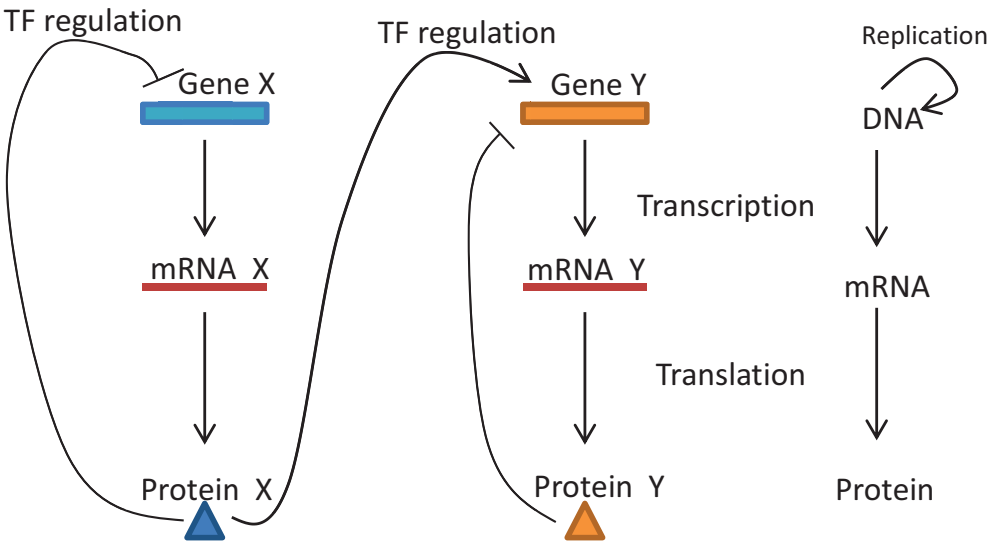


Figure 2.3: An illustration of TF regulation and the law of central "dogma".

Hecker et al. [21] which is a review about data integration in dynamic models, and more recent paper Mitra et al. [22] focusing on the soft computing methods. The remainder of this paper is organized as follows: In Section 2.2, some background knowledge of gene expression and regulation is introduced. In Section 2.3, we introduce the relevant biological experimental data that are used for reverse engineering. Section 2.4 surveys models for reverse-engineering of GRNs based on microarray data alone and on heterogeneous data, respectively. The advantages and disadvantages of each model are discussed. In Section 2.5, the optimization objective and optimization method of each inference algorithm are surveyed and discussed. Section 2.6 gives a summary of this paper.

2.2 Gene Expression and Regulation

Good models of GRNs should be built based on biological knowledge and reasonable hypotheses and be able to capture the essential mechanisms of GRNs. In this section, some background relating to the gene expression and regulation is introduced.

2.2.1 Gene Expression

There are mainly four types of molecules in a cell: DNA, RNA, protein and other small molecules. The relationships among the first three are described in Figure 2.3, which is known as the central "dogma" of molecular biology. A stretch of DNA that contains the necessary information to code for a type of protein is called a gene. Gene expression is the process that cells produce functional proteins from the instructions encoded in genes. In essence, this process is composed of three steps: First, a segment of DNA (a gene) is transcribed into an mRNA molecule (transcript). Second, the mRNA transcript is translated into a chain

of polypeptides. Third, the polypeptide chain is folded and modified into a biochemically active protein. The form and concentration of the product in each step is controlled by regulatory molecules which are usually proteins. Moreover, intermediate products such as mRNA, polypeptides, also may act as regulators of gene expression. After the whole process, the static genetic information encoded in genes is transferred to functional protein molecules.

2.2.2 Gene Regulation

The amount of mRNA produced during the transcription can reflect how active a gene is to some extent. Since mRNA levels in a cell can be measured by DNA microarray technology, the regulations of GRNs are usually studied at the transcriptional level [8].

A gene is comprised of a coding region (transcribed region) and a regulatory region (promoter region). The promoter is a control sequence of DNA upstream of the transcribed region. The coding region is the part that will be transcribed into mRNA and further translated into a protein. The mRNA transcription is controlled by the activity of RNA polymerase (RNAP), which is an enzyme that transfers genetic information from DNA to mRNA. Transcription begins with the binding of RNAP to the promoter region and then, RNAP unwinds the DNA double helix and slides along the DNA sequence. The mRNA message is elongated until the RNAP encounters a stop sequence in the DNA. A short, unique DNA sequence, called a motif that can be bound by a type of protein called the transcription factor (TF), is located in or near the promoter region. TFs can also control the transcription process by acting as an activator to promote, or acting as a repressor to block the recruitment of RNAP to specific genes. Thus, the transcriptional regulation of a gene is achieved by TFs that binds to the binding sites and exert their effects positively or negatively on binding of RNAP to the promoter region of the gene [8].

2.3 Data

Advances in high-throughput technologies have largely facilitated the study of GRNs. The most popular one used in this field is DNA microarrays, which is a large-scale gene expression monitoring technology [5, 6].

DNA Microarrays can detect the differences in mRNA levels of thousands of genes at a time. A DNA Microarray is typically a glass slide or silicon chip containing thousands, or millions of probes, each of which is complementary to a specific RNA species [4]. The concentration of an individual mRNA can be quantitatively measured by the probe. Because of the variations in sensitivities of probes, this technology is usually used to measure the relative changes in mRNA concentrations. Thus, the measurements from microarrays are typically used as the concentration ratios of each transcript relative to its baseline state [8].

There are mainly two types of microarray experiments: perturbation experiments and time-course experiments [7]. The former aims to capture the effects of a change or treatment on the cell and try to find out the genetic causes for differences between cell types or responses of pathways to disruptions. The latter aims to

	Experiment 1	Experiment 2	⋯	Experiment n
Gene 1	x_{11}	x_{12}	…	x_{1n}
Gene 2	x_{21}	x_{22}	…	x_{2n}
…	...	...	…	...
Gene p	x_{p1}	x_{p2}	…	x_{pn}

Figure 2.4: An illustration of gene expression matrix.

capture the changes of gene expression with time and are used to study the developmental process in the cell. These produce two types of data: static (or perturbation) expression data and time series expression data. An important difference between these two types of data is that static data from different assays are assumed to be independent and identically distributed (iid) while time series data exhibit a strong autocorrelation between successive time points [7].

The data from microarray experiments can be stored in the form of a matrix, called gene expression matrix. Figure 2.4 shows an example of gene expression matrix, in which each row represents a gene expression profile under different arrays and each column corresponds to the results of an array or gene-chip experiments. If it is a static data matrix, the experiments are treatments (for perturbation experiments), otherwise, the experiments are time points (for time-course experiments). A variety of microarray data from different tissues and conditions have been produced by the microarray technology and have been deposited in the online databases. One good source is the Stanford Microarray Database (SMD[1]) [23].

Although DNA microarrays can measure mRNA levels at a genome-wide level, due to the limitation of sources such as money and time, the number of experiments is usually quite small [12, 8, 13, 14]. Thus, the gene expression matrix is always narrow, i.e., $p \gg n$, where p and n are the number of rows and columns, respectively. In general, when inferring the relationships between the state variables, the number of observations is required to be larger than the number of variables. The lack of observations makes the inference task quite difficult for GRNs, which is known as the dimensionality problem or large p small n problem [12, 13, 14]. Another nature of the microarray data which make the analysis challenging is that they are typically noisy [8]. The noises come from various aspects such as the experimental devices and the procedure of doing the experiments.

In addition to the microarray data, there are other diverse data that can be used to study GRNs, including such as transcription factor binding sites (TFBS) or sequence motifs [24], ChIp-chip data [25], protein-protein

[1] http://smd.stanford.edu

interaction data [26], and even the existing large amount of information in scientific literatures. Genes regulated by the same TF share a binding consensus motif known as TFBS located in the upstream regions of genes. The TFBS can be obtained by using motif discovery algorithms to search for recurring patterns in a set of related sequences or in a position weight matrix (PWM). With the identified putative TFBS, target genes which may be regulated by that TF can be predicted. Chromatin Immunoprecipitation (ChIp) technology employs antibodies to isolate sequences directly bound by a specific TF and Microarrays are then used to scan these sequence to determine the potential locations for binding of that particular TF. The ChIp-chip results are usually transformed into p-values which indicate the significance of interactions between genes and TFs. The TF-DNA interactions characterized by ChIp-chip provide the evidence of physical TF binding but this binding is not necessarily functional. Each type of data gives the information from one aspect and they complement each other. Some researchers have reversely engineered GRNs by integrating these data with DNA microarray data [27, 26, 28, 21, 29, 30, 31].

2.4 Modeling of Gene Networks

GRNs are learned from biological data through a learning algorithm based on a network model. Depending on the degree of assumption and the availability of biological data, there are different levels of modeling of GRNs.To construct the regulatory map, GRNs are always studied at the transcriptional level. In such a network, two types of nodes, i.e., TFs and the mRNAs of the target genes, and two types of directed edge, i.e., transcriptional regulation and translation, are mainly focused on. For simplicity, many studies often combine TFs with genes encoding them and lead to an "influential GRN" [21], i.e., a GRN whose nodes are genes and edges are direct as well as indirect interaction between genes. The resulting "influential" network maps all interactions between components in a GRN onto the gene space, which can implicitly capture possible influences of TFs or metabolites on gene expressions and provide a global view of gene regulations [8, 21]. Another type of a GRN usually considered is the "transcriptional regulatory network (TRN)", which is a collection of TFs and genes with edges between TFs and genes representing TF-gene interactions. It seeks to identify true physical interactions between regulatory TFs and their target genes. The limitation of TRNs is that the regulations other than TF-gene interactions cannot be captured. For the modeling of gene networks, it is common to use "GRN" to stand for the "influential GRN". We also employ this convention in this paper.

As illustrated in Figure 2.5, different input data from experiments with particular designs are required for each reverse-engineering method. Not only can the inference methods use gene expression data and result in a GRN, but also can apply other biological data, such as ChIP-chip data, DNA sequences and protein-protein data, together with gene expression data to lead to networks at other levels such as a TRN.

Many models have been proposed to describe the GRNs. They can be distinguished and discussed from following perspectives: the state representations of nodes, the types of edges or relationships between nodes and the properties of models. The state of a node can be represented either by discrete values, such as

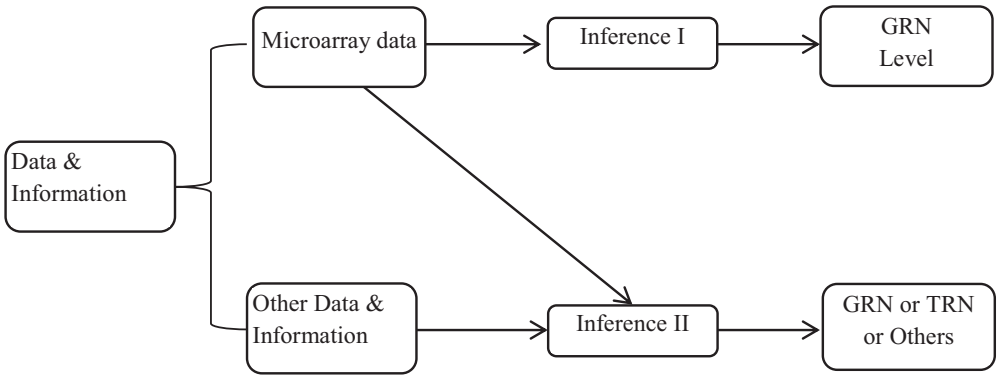


Figure 2.5: An illustration of different cases of reverse engineering of GRNs.

Boolean ("on" and "off"), or by continuous values. The edges between nodes could be directed (which may represents "causality" or "conditional dependence") or undirected. The relationships or regulations among these connected nodes may be linear or nonlinear. The properties of models refer to whether a model is dynamic or static, or deterministic or stochastic.

In the following, we review the GRN modeling methods based on two categories: models based solely on gene expression data and models based on heterogeneous data. The pros and cons of each model are discussed.

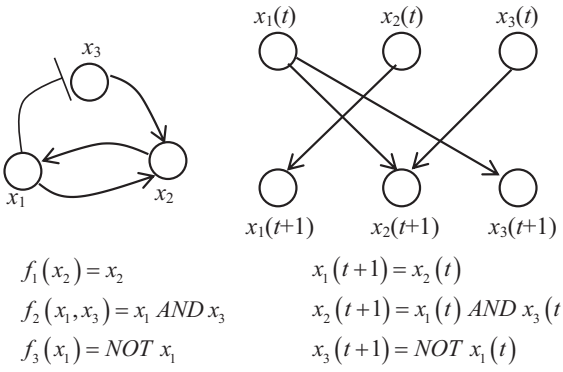
2.4.1 Models Based on Gene Expression Data

Inferring a GRN from microarray gene expression data has been a hot topic and addressed in past years. Various parameterized mathematical models have been proposed based on different assumptions. These models are mainly mathematical functions which describe the expressions of the target genes based on the activities of regulatory genes.

Boolean Networks

Boolean networks are discrete dynamic systems, which was introduced as a model of a GRN by Kauffman in 1970s [32]. When using the Boolean network to model GRNs, each gene is represented as a Boolean variable having only two states, "on" (or 1) and "off" (or 0). Here, "on" and "off" mean an active or over-expressed state and an inactive or under-expressed state of genes, respectively. The regulatory relations between genes are described by Boolean functions, which are functions of Boolean variables connected by logic operators, *AND*, *OR* and *NOT*. A Boolean network consisting of p genes is a directed graph G with a set of nodes $X = \{x_1, \ldots, x_p\}$ and a set of Boolean functions $F = \{f_1, \ldots, f_p\}$. For each node or gene x_i, its state at next step $t + 1$ is given by

$$x_i(t+1) = f_i(x_{i_1}(t), \ldots, x_{i_k}(t)), \tag{2.1}$$



transition pair	input t			output t+1		
	x_1	x_2	x_3	x_1	x_2	x_3
1	0	0	0	0	0	1
2	0	0	1	0	0	1
3	0	1	0	1	0	1
4	0	1	1	1	0	1
5	1	0	0	0	0	0
6	1	0	1	0	1	0
7	1	1	0	1	0	0
8	1	1	1	1	1	0

Figure 2.6: An example of a Boolean network.

where $x_i(t)$ is the discretized state of a gene at time t and $x_{i_j} \in X$, $1 \leq j \leq k$, $k \leq p$, are all parent nodes of x_i in G. The state of the network at a time point is represented by the vector of states of all nodes, $S(t) = (x_1(t), \ldots, x_p(t))$, and a state transition pair is denoted as $S(t) \rightarrow S(t+1)$.

Figure 2.6 illustrates an example of a Boolean network which consists of three nodes (genes), x_1, x_2 and x_3. The next state of each node is determined by the truth table in the figure. Boolean network is a dynamic model and its dynamics can be interpreted synchronously, i.e., all the states of nodes are updated simultaneously according to the previous state of the system.

To reversely engineer the GRNs with Boolean networks, we need to infer both the underlying topology and the Boolean functions of nodes from gene expression data. The Boolean networks can be inferred from both time-course gene expression data and perturbation data. For time-course data, gene expression data at two consecutive time points are treated as input state and output state, respectively. For a pair of perturbation data, the one without perturbation is considered as input and the other one after perturbation is output. Transition pairs are a collection of these input / output states, where the Boolean value of each state is obtained by discretization of continuous gene expression levels.

The number of possible Boolean networks grows exponentially with the number of nodes in the network. Thereby, exhaustive search for the optimal model is usually prohibited. It proved that the lower bound of the number of experiments required for identifying a network with p nodes is on the order of 2^{p-1} [33], which is hardly achievable in practice. However, Akutsu et al. [34] proved that if the in-degree of each node is bounded by a constant, then $O(\log p)$ transition pairs are necessary and sufficient for identifying the network correctly with high probability. Based on the bounded maximum in-degree assumption, Liang et al. [35] developed an algorithm called REVEAL to infer Boolean networks from state transition tables, which is based on the mutual information and tries to find the smallest set of input nodes that provides complete information for the output node.

Because of the ease of implementation and similar dynamics to the behaviors of biological systems, Boolean network is one of popular used models for reverse engineering of GRNs. However, they have some limitations:

In reality, the levels of gene expression are not just two states but continuous values. Hence, reducing the expression levels to just two states may be not enough and may miss the intermediate states. Information may be lost during the discretization of the continuous expression values. Besides, the updates of Boolean networks are synchronous, which may lose certain dynamic behaviors since the real biological system can update in an asynchronous way. Finally, the learning of a Boolean network is still a problem, which restricts it to only small scale networks.

Boolean networks are deterministic models. However, biological systems are known to be stochastic [36] and the data usually contain noises and measurement errors. Probability Boolean Networks (PBNs) [37, 38, 39, 40] have been proposed to introduce stochasticity to these models. In a PBN, for each node, there are several Boolean functions and each is assigned a probability to be chosen to predict the state of the node. One deficiency of the PBN is that there are too many parameters to be estimated in the model.

Bayesian Networks

Another popular GRN model is the Bayesian Network [41, 42], a graphical probabilistic model combing two areas: probability and graph theory. A Bayesian network model is a directed acyclic graph $G(X, E)$, where each node, $x_i \in X$, is a random variable representing expression level of a gene and the edges indicate the probabilistic dependent relations between nodes. If there is an edge from x_i to x_j, x_i is the parent of x_j and x_j probabilistically depends on x_i. The probability distribution of x_i conditioned on its parent nodes is denoted as $P(x_i|Pa(x_i))$, where $Pa(x_i)$ is the set of parent nodes of x_i. A Bayesian network is implicitly based on the Markov Assumption, i.e., given its parents, each variable is independent of its non-descendants. With this assumption, by applying the chain rule and properties of conditional independencies, the joint probability can be decomposed into the following product form:

$$P(x_1, \ldots, x_p) = \prod_{i=1}^{p} P(x_i|Pa(x_i)), \tag{2.2}$$

where p is the number of nodes in the networks.

In a Bayesian network, the states of genes are represented by a probabilistic distribution which can be either discrete or continuous. Figure 2.7 shows an example of a Bayesian network, in which each node has only two states, "0" ("off") and "1" ("on"). In this example, by equation (2.2), the probability that all nodes are "1" is $P(x_1 = 1, x_2 = 1, x_3 = 1) = P(x_1 = 1)P(x_2 = 1|x_1 = 1)P(x_3 = 1|x_1 = 1, x_2 = 1) = 0.6 \times 0.3 \times 0.6 = 0.108$.

Two Bayesian network structures are defined to be equivalent or indistinguishable if they represent the same set of conditional independencies. For example, the structures $x_1 \rightarrow x_2 \rightarrow x_3$ and $x_1 \leftarrow x_2 \leftarrow x_3$ both represent assertion that x_1 and x_3 are conditionally independent given x_2 [43]. From the given data, what one can learn is an equivalent class of Bayesian networks, but not an individual network. Bayesian networks in the same equivalent class are characterized as having the same underlying undirected graph structure.

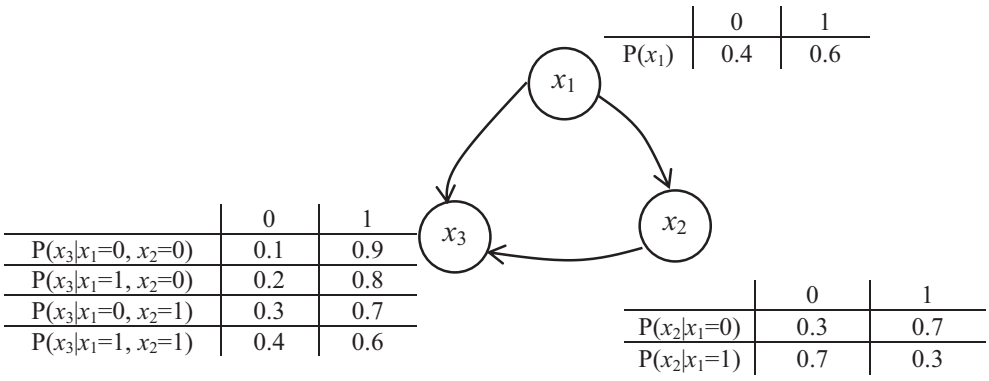


Figure 2.7: An example of a discrete Bayesian network.

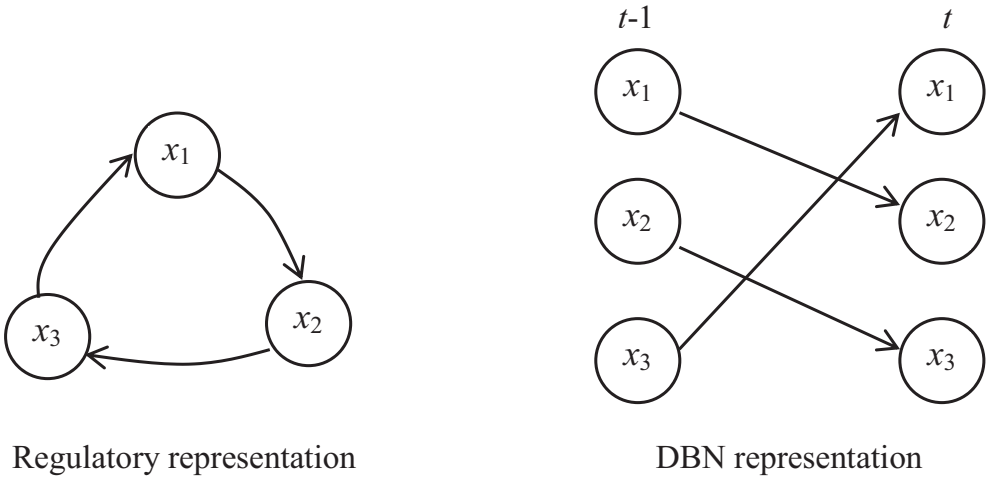


Figure 2.8: An example of a DBN.

The agreed directed edges in the equivalent class stand for causal relationships, whereas for the disagreed or undirected edges, the causality is unknown.

As a stochastic model, the Bayesian network can deal with noisy data and stochastic aspects of GRNs in a natural way [2, 36]. The Bayesian formalism enables straight-forward incorporation of prior biological information through application of Bayes rule. Furthermore, it can quantitatively describe the states of genes either in discrete or continuous and it provides an intuitive and easy way to understand visualization of GRNs. Nevertheless, the main disadvantage of Bayesian networks is that they require the network structure to be acyclic and the dynamic aspects are not considered in the models. Since feedback loops [44] and dynamics are important features of GRNs, these disadvantages limit the application of Bayesian network models.

Generally, Bayesian networks are learned from steady-state gene expression level measurements. With the time-course gene expression data as inputs, dynamic Bayesian networks (DBNs) are proposed to capture dynamic behaviors. DBN also represents the causal relationships between nodes and assume these relationships do not change over time. A simple DBN example is illustrated in Figure 2.8, which shows that DBNs

can model cyclic behaviors. Let $X(t) = [x_1(t), \ldots, x_p(t)]^T$ be a state vector of a DBN, where $x_i(t)$ is the expression level of gene i at time t. A DBN is assumed to have a Markov process, i.e.,

$$P(X(t)|X(1), \ldots, X(t-1)) = P(X(t)|X(t-1)). \tag{2.3}$$

Then, the joint probability can be decomposed as

$$P(X(1), \ldots, X(p)) = P(X(1)) \prod_{t=2}^{n} P(X(t)|X(t-1)), \tag{2.4}$$

where n is the number of time points. Thus, the regulations between genes can be modeled through the construction of $P(X(t)|X(t-1))$. Since the network structure does not change over all time points and only forward edges are allowed, the conditional probabilities can be further decomposed into the product of conditional probabilities of each gene given its parents. Therefore, given an initial state, the joint probability is

$$P(X(2), \ldots, X(n)|X(1)) = \prod_{t=2}^{n} \prod_{i=1}^{p} P(x_i(t)|Pa(x_i(t-1))), \tag{2.5}$$

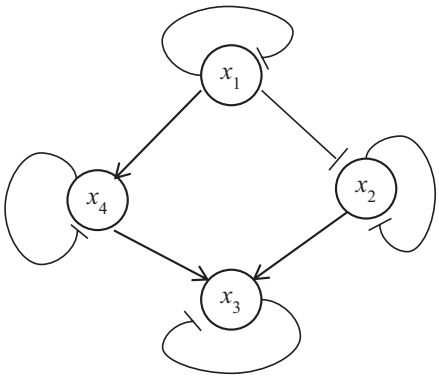
where $Pa(x_i(t-1))$ is the set of states of parent nodes of x_i at previous time step $t-1$.

Similar as Bayesian networks, DBNs can model the state of nodes in either discrete or continuous way. Ong et al. [45] used the discrete model to study the regulatory pathways in E.coli and some biological knowledge about the model structure is incorporated into their method. Kim et al. [46] proposed a continuous DBN method, which used the non-parametric regression method to capture more than linear dependencies. Perrin et al. [47] proposed a DBN model where hidden variables are genes modeled by a linear equation which is transformed from a differential equation and the output variables are observed genes. The parameters in the model are learned based on an Expectation-Maximization (EM) algorithm and missing variables can be handled in this model. One problem of this model is that it cannot be used for large-scale networks. Kim et al. [48] provided a comprehensive review of DBNs and proposed a general frame work for the DBN modeling.

Differential Equations

Differential equations are deterministic dynamical models of GRNs, in which the network topology and interactions between components are usually represented by parameters in the model. Generally, each differential equation model corresponds to a graph representation, where directed edges go from regulators to target nodes. The underlying meaning of these edges can be further explained by the corresponding parameters in the models. Differential equations quantify the change rate of each gene's expression level as a function of the expressions of regulatory genes. The general form for a GRN with p genes is

$$\frac{dx_i}{dt} = f_i(x_1, \ldots, x_p), \quad i = 1, \ldots, p, \tag{2.6}$$



W_{ij}	x_1	x_2	x_3	x_4
x_1	-	0	0	0
x_2	-	-	0	0
x_3	0	+	-	+
x_4	+	0	0	-

Figure 2.9: An illustration of the corresponding relationship between topology and the matrix.

where x_i is the expression level of gene i. In general, only a subset of $\{x_1, \ldots, x_p\}$ regulates gene i, i.e., some arguments of $f_i(.)$ have no effects on $\dot{x}_i$ and those effective arguments are regulators of x_i. The combined effects of regulators on x_i, including the biochemical effects of molecular interactions and degradations, are quantified by $f_i(.)$. Parameters of $f_i(.)$ are related to the network topology and regulation strength of each regulator. Thus, identifying gene networks requires identifying the form and estimating the parameters of $f(.)$ from data.

The expressions of $f_i(.)$'s are usually determined by considering the biochemical reactions and properties of gene networks. Because of the complexity of GRNs, these functions are usually nonlinear. However, considering the limited data and complexities of these nonlinear functions, $f_i(.)$ is usually assumed to be in the linear form.

One of the simple linear forms of $f_i(.)$ is the linear additive model [49, 20, 50, 51, 52, 53, 54], for which equation (2.6) becomes:

$$\frac{dx_i}{dt} = -\lambda_i x_i(t) + \sum_{j=1}^{p} w_{ij} x_j + u_i(t) + \epsilon_i(t), \quad i = 1, \ldots, p, \tag{2.7}$$

where λ_i is the self-degradation rate, u_i is the possible controlled external stimuli, and ϵ_i represents noises. The degradation rates are usually incorporated into w_{ij} to make the following form:

$$\frac{dx_i}{dt} = \sum_{j=1}^{p} W_{ij} x_j + u_i, \quad i = 1, \ldots, p, \tag{2.8}$$

where $W_{ij} = w_{ij}$ for $i \neq j$ and $W_{ii} = w_{ii} - \lambda_i$. W_{ij}'s represent the type and strength of influence of gene j on gene i, where a positive sign, $W_{ij} > 0$, indicates activation, a negative sign, $W_{ij} < 0$, indicates repression, and a zero means no interaction between them. Then the regulations among genes are mapped into a matrix consisting of W_{ij}'s. A simple example is shown in Figure 2.9. Equation (2.8) can be written in the matrix form. Let $\mathbf{x}^{(k)}$ and $\mathbf{u}^{(k)}$ be be the column vectors of gene expressions measurements and external influences

on each gene at experiment k, respectively. Then, the gene expression data matrix of n experiments is $\mathbf{X}_{p\times n} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$. Let $\mathbf{U}_{p\times n} = (\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(n)})$ be the matrix of external influences and $\mathbf{W}_{p\times p} = (W_{ij})$ be the regulation matrix. Then, equation (2.8) becomes

$$\frac{d}{dt}\mathbf{X}_{p\times n} = \mathbf{W}_{p\times p}\mathbf{X}_{p\times n} + \mathbf{U}_{p\times n}. \tag{2.9}$$

We can assume that the system is observed around the steady state and the linear form is obtained by the first order Taylor expansion of $f_i(.)$ and omitting the higher order terms.

The linear differential equation model can be inferred from both time-course gene expression data and steady state data. If steady state data are used, the derivative, dx_i/dt, equals zero. On the other hand, if the time-series data are used, the derivatives must be estimated. When computing the derivatives, the measurement errors in data may be amplified.

Because of the simple form of the linear differential equation, the number of parameters needed to describe the system is reduced and the amount of data required to infer the GRNs is much less than that required by a complex nonlinear differential equation model. Nevertheless, this linear model is based on the assumption that the influences of different regulators are additive and independent, which is a strong constraint. In fact, effects in a cell are usually non-additive, e.g., different TFs can cooperate and amplify the effects of each other or compete with each other [30]. This constraint may result in errors when modeling the GRNs. Despite the deficiencies of linear differential equation models, they can model the real systems to some extent and are usually employed to infer large-scale networks [55, 51, 56] due to the simplicity of their linear forms.

Biological systems are characterized with complex dynamic behaviors, such as oscillations and multi-stationary [8]. To capture these important properties of true biological network, nonlinear differential equation models are proposed. One extension of the linear differential equation model is the additive model [57, 58],

$$\frac{dx_i(t)}{dt} = s_i - \lambda_i x_i(t) + \sum_{j=1}^{p} \beta_{ij} f_j(x_j(t)), \quad i = 1, \ldots, p, \tag{2.10}$$

where s_i and λ_i are the basal synthesis and degradation rates of gene i, β_{ij} represents the regulation strength of gene j on gene i and $f_j(x_j)$ is the corresponding regulation function. Similar as the linear additive model, $\beta_{ij} > 0$ means x_j actives the expression of x_i, $\beta_{ij} < 0$ denotes the inhibition effect of x_j on x_i and $\beta_{ij} = 0$ indicates there is no regulation between x_j and x_i. The regulation functions $f_j(x_j)$'s are usually the Hill-functions,

$$f_j(x_j) = \frac{x_j^{h_j}}{x_j^{h_j} + \theta_j^{h_j}}, \tag{2.11}$$

where h_j is the Hill coefficient parameter and θ_j is a threshold parameter. The regulation functions are derived from chemical reaction kinetics. The additive model has the same deficiency as linear additive differential equations. It also assumes that the effects of regulators are independent with each other and additive.

S-system model [59, 60] derived from Biological System Theory is one of the most popular nonlinear

differential equation models. It is a type of power-law formalism described as follows:

$$\dot{x}_i = \alpha_i \prod_{j=1}^{p} x_j^{g_{ij}} - \beta_i \prod_{j=1}^{p} x_j^{h_{ij}}, \quad i = 1, \ldots, p, \tag{2.12}$$

where x_i is the concentration of component i and p is the number of genes in the network. The changes of x_i is represented by the difference of two terms, the first of which corresponds to the production while the second of which corresponds to the degradation. α_i's and β_i's are nonnegative rate constants. g_{ij}'s and h_{ij}'s are real-valued kinetic orders which reflect the regulation and interaction intensity of x_j to x_i. More specifically, if $g_{ij} > 0$, x_j activates the production of x_i, while if $g_{ij} < 0$, x_j represses the production of x_i. h_{ij} has the same effects as g_{ij} but on degradation. A zero-valued kinetic order indicates that x_j has no such effect on x_i. Therefore, the network structure and interactions between components are mapped onto the kinetic order parameters.

For a network of p genes, the number of parameters in S-systems is $2p(p + 1)$ which becomes quite large even for an intermediate-sized network. Generally, the estimation of S-system parameters is defined as a $2p(p + 1)$ dimensional nonlinear optimization problem. However, there is too many dimensions for optimization algorithms in cases that large-scale networks are studied. In addition, due to the high degree of nonlinearity, there is no guarantee for those algorithms to find the global optimal solutions. Therefore, many evolutionary algorithms have been proposed (e.g. Kikuchi et al.[61], Kimura et al. [62] and Ho et al. [63]). To circumvent expensive computational efforts in the inference of S-systems, some methods which avoid solving the system of nonlinear differential equations or take advantage of the special forms of S-systems are proposed. Voit et al. [59] replaces the left hand side of the model in equation (2.12) by slopes estimated from expression data, thereby decoupling the systems and transforming the S-systems to a set of nonlinear algebraic equations. Kimura et al. [62] proposed another strategy known as decomposition method which involves solving each differential equation in (2.12) one-by-one, in which during the integration the other state variables are treated as external inputs whose values are estimated from time-series data, thereby avoiding the integration of coupled differential equations. Liu et al. [64] proposed a method which utilizes the special form of S-systems model and decoupling strategy such that the search dimension is reduced. Vilela et al.[65] considered the form of S-systems in another way and proposed method which is based on eigenvector optimization of a matrix formed from multiple regression equations of the linearized decoupled S-system. Recently, Kimura et al. [66] developed a technique that reduces the search dimension by solving linear programming problems.

Although nonlinear differential equation models can describe the real biological network more accurately than linear models, it contains much more parameters than linear ones. Learning these parameters in nonlinear models demanding a large amount of data, which is hardly available in practice. Therefore, the practical application of nonlinear differential equation is still limited.

Gaussian Graphical Models

To construct a GRN graphical model, one simple and straightforward way is to calculate the Pearson correlations of each pair of genes from microarray gene expression data. An undirected edge appears between two genes if their correlation exceeds a threshold, which results in a relevance network [67]. However, the direct and indirect relationships between two genes cannot be distinguished in this graph.

The Gaussian Graphical Model (GGM) is proposed to model GRNs, in which the partial correlation is used to measure the conditional dependence of two genes. For a network of p genes, denote the expression levels of genes as $x_1, \ldots, x_p$, which are assumed jointly normal. The partial correlation is $\rho^{ij} = Corr(x_i, x_j | x_{-(i,j)})$, where $x_{-(i,j)} = \{x_k | 1 \leq k \neq i, j \leq p\}$. In a GGM, an edge exists between two genes if the corresponding ρ^{ij} is non-zero, which means that these two genes are conditionally dependent given all other genes. Therefore, edges in GGM only indicate the direct relations between genes. It is known that partial correlations have the following relationships with the inverse of covariance matrix,

$$\rho^{ij} = -\frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}, \tag{2.13}$$

where σ_{ij} is the corresponding element in the inverse of covariance matrix, i.e., $\Sigma^{-1} = (\sigma_{ij})$. Nevertheless, due to the large p, small n character of gene expression data, the inverse of covariance matrix always does not exist. Hence, directly inferring the GGM from the covariance matrix is not applicable. To solve this problem, some approaches infer the network structure based on low-order partial correlations [68, 69, 70], where the order of partial correlation is the number of variables conditioned on. To estimate the full-order partial correlations, the connection between partial correlation and ordinary least squares regression has been implemented [14, 13].

The edges in GGMs distinguish the directed relationships between genes from indirect relationships which may result from intermediate variables. Although these edges are undirected and do not infer causal relationships, they eliminate many possible connections among the nodes and provide good starting points for further analysis. One disadvantage of the GGMs is that they are static models.

Others

Models mentioned above are the most popular ones for the reverse engineering of GRNs from solely gene expression data, whose popularity come from the ease of implementation and good interpretation of biological systems. Except for these models, various other models have been proposed to model and infer the GRNs.

Some authors focus on interpreting the relationships between genes by the Granger causality. Gene X_i is defined as the Granger causal for gene X_j if the autoregressive model of X_j based on past values of both genes is significantly more accurate than that based on X_j alone. Specifically, let $X_i^{1:n}$ be the time-course

gene expressions of X_i up to time n and consider the following two regression models

$$X_j^{1:n} = AX_j^{1:n} + BX_i^{1:n} + \epsilon^n. \tag{2.14}$$

$$X_j^{1:n} = AX_j^{1:n} + \epsilon^n. \tag{2.15}$$

X_i is said to be Granger causal for X_j if and only if model (2.14) has significant improvement than model (2.15). For a gene network containing p variables, let $X^t = (X_1^t, \ldots, X_p^t)^T$ and consider graphical Granger model,

$$X^n = A^1 X^{n-1} + \ldots + A^{n-1} X^1 + \epsilon^n, \tag{2.16}$$

where X_j^t is said to be Granger-causal for X_i^n if the corresponding coefficient, $A_{i,j}^t$ is significant, in which case, there is an edge $X_j^t \to X_i^n$ in the graphical model with $n \times p$ nodes.

The Granger causalities between genes can be estimated by applying statistical methods to time-course gene expression observations. Both Opgen-Rhein and Strimmer [71] and Shimamura et al. [72] reconstructed the GRNs with Granger causality based on the first-order vector autoregressive (VAR) model. Shojaie and Michailidis [12] proposed a method called "truncating lasso" which can infer GRNs by using the VAR models whose order is not restricted to one and can be estimated by their algorithm. The Granger causality model can not only extract the interation relations between genes but also estimate the time lags of the regulations. Incorporated with the variable section techniques, such as lasso penalty, this model can deal with the large p small n problem. However, in reality, the regulation between genes show nonlinear behaviors which may not be captured by this linear model. The concept of Granger causality is quite intuitive which needs to be further justified in the study of biological systems.

Recently, state space models, one of the most powerful methods to describe a dynamic systems, have been developed for reverse engineering of GRNs. Much attention has been attracted due to its high computational efficiency and ability to handle the noise. The variables in state space models are divided into two groups: observed variables, Y_t, and latent variables, X_t, where $Y_t \in R^p$ is the vector of observed gene expressions at time t and $X_t \in R^k$, $k < p$, is the lower dimensional latent state vector. The assumption is that the dynamical behavior of observed Y_t is regulated by the time evolution of a few latent variables, X_t, which represents the factors that can not be measured by gene expressions, e.g., unobserved genes, activities of regulatory proteins, some biological entities present in the post-translational modifications [73]. For example, the input-driven state space model used by Rangel et al. [74] and Beal et al. [75] is

$$\begin{aligned} Y_t &= HX_t + AY_{t-1} + w_t, \\ X_t &= FX_{t-1} + BY_{t-1} + v_t. \end{aligned} \tag{2.17}$$

The matrix $A \in R^{p \times p}$ captures gene-to-gene expression level influences at consecutive time points and matrix $H \in R^{p \times k}$ captures the influences of the latent variables on gene expression level at each time point. Matrix

$B \in R^{k \times p}$ captures the influences of previous gene expressions to the current state of latent variables and F is the state dynamics matrix. Since the model (2.17) can be rewritten as

$$Y_t = (HB + A)Y_{t-1} + HFX_{t-1} + w_t + Hv_t, \tag{2.18}$$

the structure of gene regulations is obtained by estimating $HB + A$, which not only captures the direct gene-to-gene interactions but also the gene-to-gene interactions via the latent states over time [74]. Rangel et al. [74] applied the state space model (2.17) to represent the dynamics of T-cell activation and Hirose et al. [73] used the state space model to identify the transcriptional modules and module-based gene networks. Wu [76] and Wu et al. [77, 78] infer the GRNs with state space models with the consideration of time delays.

State space models can be regarded as a subclass of DBNs and have been extensively used in many areas of control and signal processing. The existence of latent variables, which denotes the factors that can not be measured, enables the modeling of noisy continuous observations. Linear relations between the variables makes this type of models computational efficient. However, for the modeling of gene network, the linear form may not always be suitable because of the existence of nonlinear regulatory behaviors observed in GRNs. Since the dimension and states of the latent variable are unknown, the identification and interpretation are another problem that has not been well solved when applying the state space models.

As an important method in soft computing, the recurrent neural networks (RNNs) have been applied to model and reconstruct the GRNs from gene expression data. In a RNN, neurons having relations are connected with each other to form a network, in which the output of each neuron goes back to its input after a unit delay. The recurrent structure of RNNs effectively reflect the feedback feature of GRNs and hence makes it suitable for the modeling of GRNs. In a RNN formulation of the GRN, each gene is considered as a neuron and the rate of change of its expression is

$$\tau_i \frac{dx_i}{dt} = f\left(\sum_{j=1}^{p} w_{ij}x_j + \sum_{k=1}^{K} v_{ik}u_k + \gamma_i\right) - \lambda_i x_i, \tag{2.19}$$

where x_i is the gene expression level of gene i ($1 \leq i \leq p$, p is the number of genes), $f(.)$ is a nonlinear function (typically sigmoid, $f(z) = 1/(1 + e^{-z})$), w_{ij} represents the effect of gene j on gene i, v_{ik} represents the effect of kth external variable u_k ($1 \leq k \leq K$, K is the number of external variables) on the gene i, τ_i is a time constant, γ is a bias term, and λ is the decay rate parameter. A positive value of w_{ij} indicates activation of gene j on gene i, while a negative value represents inhibition. No influence of gene j on gene i if w_{ij} is zero. Since gene expression data are measured at discrete time points, the discrete form of model (2.19) is

$$x_i(t + \Delta t) = \frac{\Delta t}{\tau_i} f\left(\sum_{j=1}^{p} w_{ij}x_j(t) + \sum_{k=1}^{K} v_{ik}u_k(t) + \gamma_i\right) + \left(1 - \frac{\lambda_i \Delta t}{\tau_i} x_i(t)\right). \tag{2.20}$$

The term $\sum_{k=1}^{K} v_{ik}u_k(t)$ is usually ignored, due to the unavailability of measurements of external variables.

In addition, the decay rate parameter is chosen as $\lambda = 1$, for simplicity. Vohradsky [79] studied the dynamic behaviors of a three-gene network in the framework of RNNs. Xu et al. [80] infer the GRN from time-series expression data by particle swarm optimization (PSO) approach based on the model (2.20) and their results show that RNNs can reveal potential regulatory interactions between genes.

Similar to DBNs and state space models, RNNs can be unfolded along the time points so that it becomes a layered forward network. The structure or regulatory relations between layers do not change when time evolves. Neural networks has been integrated with other soft computing methods, such as fuzzy set and evolutionary algorithms, to reversely engineering the GRNs, the readers are referred to Mitra et al. [22] which is a comprehensive review of soft computing methods for GRNs.

2.4.2 Models Based on Heterogeneous Data

Besides the dimensionality problem, the data from microarray experiments always contain many noises and measurement errors. Therefore, an accurate network can hardly be obtained due to the limited information in microarray data. With the development of technologies, a large amount of other diverse types of genomic data are collected. Many researchers are motivated to study gene networks by combining these data with microarray data. Because different types of the genomic data reflect different aspects of underlying networks, the inferences of GRNs based on the integration of different types of data are expected to provide more accurate and reliable results than based on microarray data alone. However, effective integration heterogeneous data is currently a hot research topic and a nontrivial task because they are generally collected along with much noise and related to each other in a complex way [81].

In general, using heterogeneous genomic data lead to two different levels of networks which we mentioned at the beginning of this section. One is the same level as using the microarray data alone, i.e., a GRN. The other level is the TRN, which depicts the interactions between sets of TFs and genes being regulated.

When one aims at reversely engineering a GRN from heterogeneous data, the models or mathematical templates used are the same as those mentioned above. As the Bayesian model can introduce prior information in a natural way by applying the Bayesian rule, it is widely used in this study. When Bayesian network is considered, other data are generally used to provide prior distribution for the network topology. Hartemink et al. [28] incorporated the TF binding location data into the network modeling by letting the prior probabilities of network topologies which failed to contain an edge that location data suggest one, be zero. By applying the simulated annealing search algorithm and posterior model averaging strategy, a discrete Bayesian network was learned from these data. Their work was extended from two aspects [82]. One is that a dynamic Bayesian network model which allows cyclic behavior is used in place of the Bayesian network. The other is a new informative prior over the network structures is proposed. The prior proposed by Bernard and Hartemink [82] has considered the noisy nature of location data and relaxed the constrains their previous work [28] by using a method to derive the prior probability from the p-values associated with the binding data. In the work of Imoto et al. [29], they proposed to model the network topology by a Gibbs distribution

which can be derived from various genomic data. In their paper, designs of the prior distributions from protein-protein interaction data, protein-DNA data and sequence data are discussed, respectively. A Bayesian network based on a nonparametric regression can be reversely engineered by their method. Based on this framework, Nariai et al. [26] inferred the gene network by integrating gene expression data and protein-protein interaction data from MIPS database. In the estimation procedure, if the relation between genes is found listed in protein-protein data, a protein complex and its activities are constructed from the expression levels of corresponding genes based on the principle component analysis. Werhli and Husmeier [83] extended the method of Imoto et al. [29] to allow the simultaneous inclusion of different sources of prior knowledge by designing a particular form of energy function such that the computation of the marginal posterior distribution over the hyperparameters becomes analytically tractable. Tamada et al. [27] proposed a method to infer a gene network and detect regulatory motifs simultaneously from the combination of gene expression data and DNA sequence information based on the framework of Imoto et al. [29]. The basic assumption in their work is that genes having the same parent share a consensus motif in the promoter regions of their DNA sequences. Tamada and Kanehisa [84] presented a statistical method which can estimate two GRNs of two distinct organisms from gene expression data simultaneously with a Bayesian network model utilizing the evolutionary information. The underlying idea in this paper is that orthologous genes with similar expression patterns have strongly related functions in each of the organisms' cells, so that gene expression data of one organism helps to estimate the gene network of the other.

Some researchers aim to reconstruct a TRN from these heterogeneous genomic data with a set of putative or identified TFs. In general, reverse engineering of a TRN is more difficult than that of a GRN, because both the network topology and the activities of TFs are needed to estimate. Bar-Joseph [85] developed the GRAM algorithm to discover regulatory networks in Saccharomyces cerevisiae by integrating the microarray expression data and the ChIP binding data. However, their method is not systematic modeling and depends on a heuristic parameter threshold. One of the commonly used models in literature for the study of this purpose is

$$\frac{x_i(t)}{x_0(t)} = \prod_{j=1}^{m} \left(\frac{a_j(t)}{a_j(0)} \right)^{J_{ij}}, \tag{2.21}$$

where $x_i(t)$ and $a_j(t)$ are the concentrations of mRNA i and TF j at experiment t, respectively. J_{ij} is the control strength of TF j on gene i. Sun et al. [81] derived this model based on a series of reversible biochemical reactions and assumed that all these reversible reactions reach equilibrium. By taking log-transformation, we can get the linear form of equation (2.21) as follows

$$\log(x_i(t)/x_i(0)) = \sum_{j=1}^{m} J_{ij} \log(a_j(t)/a_j(0)). \tag{2.22}$$

Considering the measurement noise and intrinsically stochastic nature of biological systems, the model can

be further expressed in matrix notations

$$X = JA + E, \tag{2.23}$$

where X is the relative gene expression data matrix with p rows (genes) and n columns (samples), A is a $m \times n$ matrix representing the relative activities of TFs, J is a $p \times m$ matrix whose elements denote the regulatory strengths, and E is the matrix of errors.

Although the regulation in gene networks is nonlinear, the linear model can reflect the properties of networks to some extent. Based on this linear model, Liao et al. [86], Kao et al. [87] and Boulesteix and Strimmer [88] have used a "network component analysis" approach to find activities of TFs and regulation strengths using gene expression data and ChIP binding data. However, in their studies, the connectivity matrix between genes and TFs, which is obtained from binding data, is assumed to be known without error. Based on equation (2.23), Sun et al. [81] developed a Bayesian hierarchical model to integrate the protein-DNA binding data and gene expression data to reconstruct TRNs. The measurement errors of both binding data and expression data are explicitly considered in the model. They use the measured relative protein-binding intensities of TFs to approximate the regulation strengths and employ the Markov Chain Monte Carlo (MCMC) method to get the network topology and TF activities from the posterior distributions. Sabatti and James [31] proposed a Bayesian hidden component model which is also based on the linear model to infer a TRN from gene expression with the prior distribution of network topology derived from promoter binding sites. Then, the network topology and regulation strengths as well as the activities of TFs are obtained from their posterior distributions. The uncertainty in the sequence information is also taken into account in their method.

The combinatorial regulation mechanism is common in the GRNs, e.g., TFs may cooperate or compete with each other when regulating a target gene. The synergistic and antagonistic relationships between genes have been investigated by Banerjee and Zhang [89]. Jensen et al. [30] developed a Bayesian hierarchical model that integrates gene expression data, ChIP binding data and promoter sequence data to reconstruct TRNs. In their method, the relationship between TFs and genes are modeled as follows

$$x_i(t) = \alpha_i + \sum_{j=1}^{m} J_j C_{ij} a_j(t) + \sum_{j \neq k} \gamma_{jk} C_{ij} C_{ik} a_j(t) a_k(t) + \epsilon_{it}, \tag{2.24}$$

where α_i is the baseline expression of gene i, J_i represents the linear effect of TF j on gene expression, C_{ij} is regulation indicator which equals 1 when TF j influences gene i otherwise equals 0, and γ_{jk} indicates the synergistic or antagonistic effect of TFs j and k. An informative prior distribution of C_{ij} is constructed from both the ChIP binding data and the promoter sequence data. The network topology as well as the relationships between TFs is estimated by using an MCMC algorithm. However, Jensen et al. [30] use the expressions of genes producing that TFs as the measurement of activities of TFs, which is not suitable as mentioned in literature [90, 86, 87, 88].

2.5 Inference Algorithms

After a model is chosen, the next critical step is to apply an inference algorithm to fit the model with data. This is a nontrivial step because the number of parameters is usually quite large but the information available is quite limited. The design of an inference algorithm depends not only on the complexity of the model framework but also on the quantity and quality of available data.

There are mainly two tasks: learning network topology and estimating values of parameters, where the network topology is of primary interest, which aims at finding the connectivity or regulation relations between the network components. Given the network topology, the values of parameters concerning the biological information, such as regulatory strengths, are estimated in the parameter estimation process. The identified network topology and estimated model parameters should satisfy existing biological constrains and be able to best explain observed biological data.

When inferring the network topology, identifying the set of regulators of each node is a challenging problem. For a network consisting of p nodes, the number of possible combination regulators for each node is 2^p, which is quite large even for a small-sized network. Therefore, the network topology inference is a combinatorial problem as well as a model selection problem since different topologies corresponds to different model structures. The parameter estimation problem is relatively easier compared with the topology inference problem. Additionally, these two problems are generally not independent with each other in the network inference. In some cases (c.f. [27, 26, 29]), the parameter estimation is embedded into the process of topology inference, while in other cases (c.f. [49, 13, 63]) the topology is simultaneously inferred in the parameter estimation process.

The network reconstruction problem is generally formulated as an optimization problem. Both the topology and parameters of a network is learned by optimizing a specific objective under some biological constrains through an optimization method. In the following, we will survey the inference algorithms from two perspectives: optimization objectives and methods.

2.5.1 Optimization Objectives

Gene networks are usually learned by solving optimization problems, the objectives in which should not only consider the fitting of observed data but also take into account or reflect some biological properties. The most common and important property of gene networks is sparseness [20] meaning that a gene is regulated only by a limited number of genes. As we mentioned above, the network topology inference is a model selection problem. In general, one of the important criterions for selecting the optimal model is the parsimony, i.e., the model should not be too complex, which coincides with the sparseness.

Under the Bayesian model framework, the optimal model is chosen by maximizing the network posterior

probability. Denoting the network by G, given data X, network G^* is inferred by

$$G^* = \text{argmax}_G P(G|X). \tag{2.25}$$

Using the Bayesian rule, the posterior probability is

$$P(G|X) = \frac{P(G)P(X|G)}{P(X)} \tag{2.26}$$

where

$$P(X|G) = \int P(X|\theta_G)P(\theta_G|\lambda)d\theta_G \quad \text{and} \quad P(X) = \sum_{G\in\Omega} P(X,G). \tag{2.27}$$

P(G) is the prior distribution of network topology, θ_G is the parameters related to network G, λ is the hyperparameter and Ω is the set of all possible networks. By taking logarithm and omitting the constant term (P(X)), we can define a Bayesian score for each network

$$BS(G) = \log(p(G)) + \log(p(X|G)). \tag{2.28}$$

Thus, the network can also be inferred by maximizing $BS(G)$ which takes into account both the prior probability and the likelihood. It is known that the marginal likelihood $P(X|G)$ has the effect of penalizing complex topologies, which is known as Occam's razor. Thus, the sparseness is reflected in $BS(G)$. If prior distributions of parameters are properly designed, the integral in equation (2.27) can be solved analytically. Kim et al. [46] used Laplace approximation to approximately compute the integral in equation (2.27) and proposed a criterion called Bayesian Nonparametric Regression Criterion (BNRC) which is widely used in their works [27, 26, 48, 46, 29].

For models based on differential equations, the topology is usually determined by values of parameters. To estimate the parameters, the objective in general is

$$\text{argmin}_\theta \|X - X^c\|^2 + r(\theta), \tag{2.29}$$

where X and X^c are the observed gene expression data and calculated expression data, respectively and $r(\theta)$ is a penalty function where θ is the parameter vector. The first term in formula (2.29) represents the fitting to the observed data and the second term is usually to make sure the network is sparse and avoid over-fitting. $r(\theta)$ usually takes the lasso or its adaptive form to penalize the parameters of topology, e.g., $r(\theta) = \sum_{i,j} |\theta_{ij}|$ where θ_{ij} is parameters related to the connectivity [91, 62, 61, 63]. Since the calculation of X^c requires to solve solve a set of differential equation which takes quite a lot of computation time, the following objective is sometimes considered [59],

$$\text{argmin}_\theta \|\dot{X} - \dot{X}^c\|^2 + r(\theta) \tag{2.30}$$

where $\dot{X}$ and $\dot{X}^c$ are observed and calculated derivatives, respectively. $\dot{X}$ is usually obtained from the available data by using numerical methods, which can avoid the numerical integration of differential equations. However, the estimation of slopes may introduce more errors that may affect the final results. When using the linear differential equations, the model can be discretized and transformed into a linear algebra equation. Then the objective function can be similar as formula (2.30) using a penalty term or the network structure can be optimized directly by other techniques, such as singular value decomposition (SVD)[54, 49]. Other model such as the GGMs [13] and Granger Causality [12] that can be represented as a linear algebra equation can also apply the lasso type penalty term to infer the topologies.

The objectives mentioned above can be considered to get a point solution, i.e., each method will result in only one inferred network. Other literature, such as Husmeier [92], aim to find the posterior probability distributions of the topology and parameters under the Bayesian framework. With the estimated posterior distribution, the confidence of the network inferred can be obtained, which, however, requires more computation cost.

2.5.2 Optimization Methods

Given the objective, solutions to some of them are obtained by using standard optimization methods, such as linear programming [54] and convex programming [91]. When the model is linear, the main problem in applying those methods relies in the limited information in data. The dimensionality problem prohibits the direct use of many existing methods. However, the sparseness property makes network inferences from limited data possible, e.g., Peng et al. [13] proposed an active-shooting algorithm to infer a GGM under the high-dimension and low-sample setting. Shojaie and Michailidis [12] developed a truncating lasso penalty based on the sparseness assumption and proposed an efficient algorithm to infer the Granger causality relationships in the network under the same setting.

For methods based on complex models, such as Bayesian models and nonlinear differential equation models, the objectives cannot be optimized analytically, e.g., the optimization problem (2.25) for Bayesian networks is known to be NP hard. Therefore, the researchers usually resort to heuristic search methods. For instance, Imoto et al. [29] and Naria et al. [26] employed the greedy hill-climbing algorithm to infer the network which optimizes the Bayesian score. The simulated annealing has been used to infer a Bayesian network [82, 28]. The genetic algorithm (GA) and its variants have been applied in the inference of an S-system [61, 62, 64]. A recurrent neural network model was learned by using the PSO algorithm [80]. The problem of using these heuristic optimization methods is that they are usually time consuming and no good way to avoid falling into the local optimum.

In the Bayesian framework, sometimes one intends to find the posterior distribution of the topology and parameters instead of a point solution. In general, calculating the posterior distribution analytically is prohibitive. Hence, researchers apply Markov chain Monte Carlo (MCMC) method to generate random samples from their posterior distribution [92, 30]. With these random samples, the posterior probability can

be approximately estimated.

Another approach to the directed acyclic graph (DAG) structure learning of a Bayesian network is the constrained-based method: each edge in network is learned if fulfilling a constraint by comparing the value of a statistical or information-theory-based test of conditional independence of each pair of nodes to a threshold. In the family of constrained-based learning algorithms, Path consistency (PC) algorithm [93] is a well-known structure learning algorithm which constructs a DAG in two consecutive stages. The first stage is learning associations between variables and resulting in an undirected graph (i.e., the skeleton of the graph). In this process, the number of conditional independence tests grows exponentially with the number of variables. A simple method reducing this complexity to polynomial complexity is to fix the maximal number of parents of each node. The second stage orients the undirected edges in the skeleton according an orientation rule such that the graph is a DAG. Saito and Horimoto [94] developed a simple method to assess co-expressed genes from expression profiles based on the PC algorithm. Tan et al. [95] proposed a new extended PC algorithm which can incorporate the prior knowledge from multiple sources such as ChIp-chip data and can work well for partially dense graphs. When using the PC algorithm with partial correlation as the conditional independence test, there are two problems the same as in GGM: the high-order (i.e., a large condition set) independence test and small sample size which are both avoided by identifying dense nodes and using multiple sources of data [95]. The expression data of genes are assumed to be normally distributed when utilizing the partial correlation independence test in PC algorithm. However, if this assumption does not hold, partial correlation can only detect the linear relation between pairs of genes, whereas nonlinear dependences are more common in biological processes. To cope with this problem, Zhang et al. [96] proposed to infer the GRNs from gene expression data by employing PC algorithm based on conditional mutual information. Algorithms of constraint-based approach are generally asymptotically correct and relatively efficient [93]. However, they depend on the threshold selected for the conditional independent test. Other efficient Bayesian network learning algorithms and their comparisons refer to Brown et al. [97].

The computation costs for both heuristic methods and MCMC methods are usually very high. The accuracies of results from these methods are not high enough, due to the limited data. Therefore, to increase the accuracy and efficiency, biological knowledge needs to be considered when designing and utilizing these methods. For instances, the prior probabilities of topology and parameters used by Jensen et al. [30] and Mazur et al. [57] take into account the sparseness property. Sheridan et al. [98] proposed a prior reflecting the scale-free property. In some literature [61, 62, 64], the parameter values are searched in the intervals having biological meanings.

2.6 Conclusions and Discussion

Various molecules in the cell and their interactions form a complex system called a GRN, the network identification of which has received considerable attention due to the significance of its potential applications.

Depending on whether identification aims at grouping the genes with similar expression profiles or finding the regulatory relations between genes, approaches can be classified as clustering methods or reverse engineering methods. The former methods group genes with similar expressions, due to the high probability that they are functionally directly or indirectly with each other. Clustering methods in data mining and soft computing, e.g., biclustering by fuzzy sets [22], and their extensions are always applied to group the genes. The methods reviewed in this paper falls under the latter class that aims at unveiling the regulation relations between genes. Due to the limited observations and high dimensionality of the biological data, most reverse engineering methods are unable to reconstruct large-scale networks. In this case, the data mining methods can be used as preliminary step to reduce the dimension by selecting the group of components that we are interested in.

Different mathematical and computational methods have been developed to reveal the regulation relations between the components in gene networks from biological data. Depending on the level of abstraction and types of available data, these methods may result in a GRN, a network consisting of genes, or a TRN, a network consisting of TFs and genes. Both the direct or indirect relations between genes can be captured in a GRN, while in a TRN, only the relations between TFs and genes are identified. In general, the identification of a TRN is more difficult than that of a GRN, because both the network structure and the activities of TFs are unknown. Thus, the identifications of TRNs often require the participation of other source of biological data in addition to gene expression data.

Due to the large p small n problem, most models currently can only reversely engineer a relatively small-scale network. The ability to infer a large network is very important since the real-life gene network is usually very large. GGM can reconstruct a large-scale network from the under-sampled data and lead to an undirected network whose edge means direct relations. Though the causal relations can not be described in this network, it is still a good start point which eliminates many possible edges. However, it is a static model that can not reflect the dynamic behaviors of the gene network.

Based on the static gene expression data, many models, such as Bayesian network, can build a static network. However, Bayesian network typically do not accommodate cycles and hence, can not handle feedbacks that are common in gene networks. With the availability of time course gene expression data, DBNs, RNNs, ODEs, and state space models are developed to model the network dynamics. DBNs, state space models and RNNs can be unfolded as multiple layered network along the time points, in which, the regulatory structure is assumed to be unchanged. Though Boolean networks can model dynamics, some dynamic behaviors may not be able to captured due to the discretization of gene expression data and its synchronous update. As an effective model for dynamical systems, ODE captures the causality and feedbacks of gene network in a natural way. The system structures are usually reflected by the values of parameters in these models.

Bayesain networks or DBNs model the expression of genes as random variables, which enables their abilities to deal with noises or missing data that are prevalent in biological data. The existence of latent variables in state space models also makes it be able to model the noisy measurements or missing data. ODEs and RNNs mentioned in this paper are deterministic models which can not handle the noisy data. Especially,

when the derivatives in the ODEs need to be estimated from the observed data, the numerical errors caused by noises may lead to significant errors in the final results.

To efficiently infer the gene network, many models are simplified to be the linear forms, such as ODEs and state space models. Although linear additive regulation can reveal certain linear relations in the regulatory systems, it lacks the capability to capture the nonlinear relations between the components in the network. To finely capture the nonlinear dynamics in the gene network, the model should be able to describe the mechanisms of the systems. Bayesian netowrks, DBNs and state space models are models based on some statistical assumptions which may not be able to reflect the mechanisms. The S-system of ODEs, which is derived from the Biochemical System Theory, is a good framework to depict the GRNs.

Based on a model, a network is inferred through a learning algorithm, in which there are two tasks: identifying network topology and estimating the parameters. Topology identification is a model selection problem, while parameter estimation is an estimation problem. These two problems are always blended with each other in the inference process. The network inference is usually formulated as an optimization problem whose objective is to find the parsimonious model that can best explain the observed data. The parsimony reflects the sparseness property of biological networks. Because of the complexity of biological systems and limited information in the data, the optimum solution is always difficult to find. Therefore, heuristic search and evolutionary algorithms are widely used. Moreover, some existing literature aims to find the posterior distributions of the topology and parameters, so that the confidence or variation of inferred network can be obtained. MCMC method is usually applied to get random samples from those posterior distributions. To increase the accuracy and reduce the computation cost, the biological properties and constraints should be taken into account when designing and applying these optimization methods.

The information contained in the gene expression data is quite limited, which makes the network inference difficult. One way to alleviate this problem is to aggregate multiple data sets. For one gene network, there may exist multiple gene expression data sets, each of which contain part of information of this system. To find an effective way to aggregate these multiple gene expression data is nontrivial, because there is no temporal connections between different data sets. Another way is to use heterogenous data including DNA sequence data, ChIP-chip data, and protein-protein interaction data to combine with gene expression data to reversely engineer gene networks. Bayesian networks and DBNs can naturally and conveniently make use of these data by transferring them into prior distributions of the network. The prior biological knowledge is also important to increase the accuracies and efficiency, e.g., the incorporation of the scale-free properties into reverse engineering methods is still in development.

Acknowledgements

This research is supported by Nature Science and Engineering Research Council of Canada (NSERC).

Chapter 3

Properties of Sparse Penalties on Inferring Gene Regulatory Networks from Time-course Gene Expression data

Published as: L.Z. Liu, F.X. Wu, and W.J. Zhang, "Properties of sparse penalties on inferring gene regulatory networks from time-course gene expression data," *IET Systems Biology*, April 2014, published [99].

In the previous chapter, we have reviewed different methods for reverse engineering the GRNs. It shows that linear ODEs are effective models to describe the GRNs and can capture the dynamic behaviors of the system to some extent. Based on the steady-states of the linear ODEs, sparse penalties, e.g. the adaptive LASSO, have been successfully applied to infer the network topology from static gene expression data [100], as it has been theoretically proved that some sparse penalties have the "oracle properties" under the linear regression setting with independent variables [16, 17]. However, when using the time-course gene expression data, the steady-states assumption for ODEs is invalid and properties of sparse penalties under this setting should be studied.

In this chapter, two sparse penalties, adaptive LASSO and SCAD, are proposed to infer GRNs from time-course gene expression data based on an autoregressive model which is derived from a linear ODE with mild assumptions. We analyze the statistical properties of these two penalties under this setting and show that they still enjoy the "Oracle properties". This chapter partially fulfills Objective 2 of this thesis.

abstract

Genes regulate each other and form a gene regulatory network (GRN) to realize biological functions. Elucidating GRN from experimental data remains a challenging problem in systems biology. Numerous techniques have been developed and sparse linear regression methods become a promising approach to infer accurate GRNs. However, most linear methods are either based on steady-state gene expression data or their statistical properties are not analyzed. Here, two sparse penalties, adaptive LASSO and SCAD, are proposed to infer GRNs from time-course gene expression data based on an auto-regressive model and their Oracle properties are proved under mild conditions. The effectiveness of those methods is demonstrated by applications to *in silico* and real biological data.

3.1 Introduction

Most genes make their functions through regulating among other genes and such regulatory interactions form a large-scale gene regulatory network (GRN). Analyses of GRNs are very important for identifying the disease mechanisms and finding therapeutic targets. Therefore, reconstruction or "reverse engineering" of GRNs from experimental data has significant biological meanings and is one of the fundamental challenges in Systems biology.

The reconstruction of GRNs is a non-trivial problem and many conventional methods cannot be applied directly. On the one hand, a typical gene expression data set consists of relatively few observations compared to a large number of genes. The lack of observations makes the inference task quite difficult and is known as dimensionality problem [13, 14]. On the other hand, the inherent noises in the gene expression data also makes the task challenging.

Two types of gene expression data are usually used for the inference: steady-state data and time-course data. The former measures the expression levels of genes in different samples while the latter measures the expression levels of genes at several successive time points. Steady-state data are assumed to be independent while time-course data exhibit autocorrelations between time points [7]. Compared with steady-state data, dynamic information and more complex regulatory relations could be captured by time-course data. This study considers the way of extracting regulatory relations from the time-course data.

Various approaches have been developed to infer GRNs from gene expression data, such as Boolean networks [38, 101], Bayesian networks [41, 92], neural networks [102], state-space models [74, 77] and differential equations [20, 64]. A wide variety of methods have been introduced and discussed in a review by Liu et al. [18]. A comparison of these methods requires their evaluation on benchmark data sets which have been provided by DREAM (Dialogue for Reverse Engineering Assessments and Methods) project [103]. The DREAM aims at understanding the strengths and the limitations of various methods to reconstruct the GRNs from high-throughput data.

It has been observed that biological networks exhibit low connectivity [104] and the sparseness of GRNs has been employed as an important prior knowledge to tackle with the dimensionality problem. In some methods, it has been used as explicit constraints on the connectivity of network components [20]. Some literature employ ℓ_1 norm approach to explore sparse networks. For example, Peng et al. [13] use the sparse regression which includes an ℓ_1 penalty to select the non-zero partial correlations from under-sampled data and obtain an undirected network whose edge means direct relations between genes. Zavlanos et al. [105, 91] employ a weighted ℓ_1 relaxation as the objective function, which together with additional linear constraints leads to a linear program to fit the data as well as satisfy the sparse structure. The data used in these two methods are steady-state data. Fujita et al. [106] use the ℓ_1 norm method based on the autoregressive model to infer GRNs.

A system is stable if its responses to any bounded input or stimuli are bounded. Generally, stability

is used as a criterion to illustrate the qualities of the inferred networks [107]. Since it is not imposed as explicit constraints, the resulting inferred network in many methods might not be stable. For example, the network inferred by Hoon et al. [108] is unstable while the one in Hu et al. [107] is almost stable. Recently, Zavlanos et al. [105, 91] have shown that the inference performance is significantly improved by explicitly imposing the stability condition on the network. The stability constraints could be linear or semidefinite constraints which are derived from Geršgorin's Theorem or Lyapunov inequality. In our previous work [109], the stability constraint for a linear model is derived from Geršgorin's Theorem and has been incorporated into an optimization framework to infer the GRNs from time-course gene expression data.

For a target gene, identifying its regulatory genes is essentially a variable selection problem. The least absolute shrinkage and selection operator (LASSO), developed by Tibshirani [15], is an effective variable selector. However, it has been shown that the LASSO produces biases [17] and may fail in variable selection if the "Irrepresentable Condition" [110] is violated. To remedy the issue of LASSO, two new penalties are proposed recently: nonconcave Smoothly Clipped Absolute Deviation (SCAD) penalty [17] and adaptive LASSO [16]. Both methods have been shown to enjoy the Oracle properties [17], i.e., asymptotically recovering the true model at the optimal rate, under the linear regression setting with independent observations.

Among numerous techniques, sparse linear regression methods become a promising approach e.g. [106, 100, 111] because of their simplicity and potential to be applied to large networks. Many methods can be considered as extensions to the sparse linear regression, e.g. [112, 113]. However, most studies are based on steady-state data and for those based on time-course data, they just employ the sparse penalties directly and do not analyze their statistical properties. This study employs the adaptive LASSO and SCAD penalties to explore sparse network topologies from time-course gene expression data. We show that, under some mild regularity conditions, the adaptive LASSO and SCAD penalties still enjoy the Oracle properties in the time series autoregressive setting, which means that the inference methods are theoretically guaranteed. The optimization problems involved are efficiently solved by coordinate descent method [114], the fastest for computing a variety of LASSO related loss functions. Instead of giving a determinant network topology, we adapt the recently proposed strategy "stability selection" [115] to our methods to produce a network topology with a probability assigned to each edge. In the strategy, Moving block bootstrap is used to generated the bootstrap samples and the stability condition is used to help computing the edge probability. Using the system stability condition to help calculating edge probability rather than as constraints like in other papers [105, 109] makes the network inference simpler and more accurate as discussed. The methods can also be applied to the high dimensional data sets where dimensionality problem always happen. The effectiveness of the methods is demonstrated by applying them to *in silico* data and real experimental data.

Briefly, the remainder of the paper is organized as follows. In Section 3.2, a model for GRN and the sparse penalties used in this paper are introduced. In Section 3.3, the Oracle properties of the sparse penalties in terms of inferring GRNs from time-course data are proved. Section 3.4 describes the inference algorithm and the stability selection procedure. In Section 3.5, the effectiveness of our methods is demonstrated by

applying them to several *in silico* networks as well as real gene networks, respectively. Section 3.6 concludes this study.

3.2 GRN Inference

3.2.1 Model

Similar to [109], the dynamic model we use for a GRN consisting of p genes is

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{C}\mathbf{x} + \mathbf{S}\mathbf{r} + \mathbf{n} + \boldsymbol{\xi}, \\ \mathbf{r} &= \mathbf{f}(\mathbf{x}), \end{aligned} \tag{3.1}$$

where $\mathbf{x} = [x_1, \ldots, x_p]^T \in \mathbb{R}^p$ is the vector of concentrations of mRNAs of p genes; $\mathbf{C} = \text{diag}[-c_1, \ldots, -c_p] \in \mathbb{R}^{p\times p}$ is a diagonal matrix, where $c_i > 0$ is the degradation rate of gene i; $\mathbf{n} = [n_1, \ldots, n_p]^T \in \mathbb{R}^p$ is the basal expression rate in the absence of regulators; $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_p]^T \in \mathbb{R}^p$ is noise; The vector $\mathbf{r} = [r_1, \ldots, r_m]^T \in \mathbb{R}^m$ represents the reaction rates, which is a function of mRNA concentrations and $\mathbf{S} \in \mathbb{R}^{p\times m}$ is the stoichiometric matrix of the network.

We assume that at the sampling points, $t_k = k\Delta t, k = 0, 1, \ldots, n$, $\mathbf{r}_k = \mathbf{r}(t_k)$ is a linear function of $\mathbf{x}_k = \mathbf{x}(t_k)$,

$$\mathbf{r}_k = \mathbf{F}\mathbf{x}_k, \tag{3.2}$$

where $\mathbf{F} \in \mathbb{R}^{m\times p}$. Thus, (3.1) becomes

$$\dot{\mathbf{x}}_k = \mathbf{C}\mathbf{x}_k + \mathbf{M}\mathbf{x}_k + \mathbf{n} + \boldsymbol{\xi}_k, \tag{3.3}$$

where $\mathbf{M} = \mathbf{SF} \in \mathbb{R}^{p\times p}$. The elements of $\mathbf{M} = (m_{ij})_{1\le i,j\le p}$ indicate the network topology or regulatory relationships between genes in the network. $m_{ij} > 0$ if gene j activates the expression of gene i; $m_{ij} < 0$ if gene j inhibits the expression of gene i; $m_{ij} = 0$ if gene j does not regulate the expression of gene i.

In order to infer the network topology, the signs of elements of matrix $\mathbf{M}$ must be determined. Since the gene expression levels are sampled at several time points, by using zeroth-order-hold discretization method and assuming $\mathbf{r}$ is piece-wise constant between the sampling time points, system (3.3) is discretized as

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{b} + \boldsymbol{\varepsilon}_k \tag{3.4}$$

where $\mathbf{A} = e^{\mathbf{C}\Delta t} + \mathbf{C}^{-1}(e^{\mathbf{C}\Delta t} - \mathbf{I})\mathbf{M}$, $\mathbf{b} = \mathbf{C}^{-1}(e^{\mathbf{C}\Delta \mathbf{t}} - \mathbf{I})\mathbf{n}$ and $\boldsymbol{\varepsilon}_k = \int_0^{\Delta t} e^{\mathbf{C}\tau}\boldsymbol{\xi}(k\Delta t - \tau)\, d\tau$ the noise at time t_k. Note that $e^{\mathbf{C}\Delta t}$ and $\mathbf{C}^{-1}(e^{\mathbf{C}\Delta t} - \mathbf{I})$ are both diagonal matrices and their diagonal elements are all positive. Thus, the off-diagonal elements of $\mathbf{A} = (a_{ij})_{1\le i,j\le p}$ have the same signs as those of $\mathbf{M}$. For the main diagonal elements, only negative signs can possibly be identified, i.e. $m_{ii} < 0$ if $a_{ii} \le 0$, but $\text{sgn}(m_{ii})$

is unknown if $a_{ii} > 0$. In this study, we are mainly interested in inferring the relationships between genes, i.e., the signs of off-diagonal elements of $\mathbf{M}$, and do not consider self-regulations. Therefore, the inference of network topology can be achieved by identifying the signs of non-zero off-diagonal elements in matrix $\mathbf{A}$.

Given time course gene expression data, $\{x_{it}\}$, $i = 1, \ldots, p$, $t = 0, \ldots, n$, we are seeking to infer the matrix $\mathbf{A}$ in the model (3.4). Let $\mathbf{x}_k$ represent expression levels of p genes observed at the kth time point. Denote by $\mathbf{A}^*$ the true value of $\mathbf{A}$. If $\widehat{\mathbf{A}}$ is an estimate of $\mathbf{A}^*$, $\mathbf{b}$ is estimated by $\widehat{\mathbf{b}} = \bar{\mathbf{x}}_t - \widehat{\mathbf{A}}\bar{\mathbf{x}}_{t-1}$, where $\bar{\mathbf{x}}_t$ and $\bar{\mathbf{x}}_{t-1}$ are the means of last and first n observations, respectively, i.e. $\bar{\mathbf{x}}_t = \sum_{k=1}^{n} \mathbf{x}_k/n$ and $\bar{\mathbf{x}}_{t-1} = \sum_{k=1}^{n} \mathbf{x}_{k-1}/n$. Without loss of generality, we assume that the data $\{\mathbf{x}_k\}$ are centered data, so the intercept $\mathbf{b}$ is not included in the model that we mainly use in this paper,

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \boldsymbol{\varepsilon}_k. \tag{3.5}$$

We define $\mathbf{Y} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$, $\mathbf{L} = [\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}]$ and $\mathbf{E} = [\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_n]$. With these notations, (3.5) can be written in the matrix form:

$$\mathbf{Y} = \mathbf{A}\mathbf{L} + \mathbf{E}. \tag{3.6}$$

Let $\mathbf{y} = vec(\mathbf{Y})$, $\boldsymbol{\beta} = vec(\mathbf{A})$ and $\mathbf{e} = vec(\mathbf{E})$, where $vec(\cdot)$ is the column stacking operator. The vector form of (3.6) is:

$$\mathbf{y} = \mathbf{Z}\boldsymbol{\beta} + \mathbf{e}, \tag{3.7}$$

with $\mathbf{Z} = \mathbf{L}^T \otimes \mathbf{I}$, where $\otimes$ represents the Kronecker product. The matrix form (3.6) is mainly used for computation as it can be separated by rows of $\mathbf{Y}$, $\mathbf{A}$ and $\mathbf{E}$ into a set of uncoupled linear regression problems. In this way, $\mathbf{A}$ can be estimated row by row, and the storage of very large matrix like $\mathbf{Z}$ is avoided. The vector form (3.7) is mainly used for technical proofs because existing results can be applied directly for the classical linear regression form of (3.7).

3.2.2 Sparse Penalties

GRNs are known to be sparse, i.e., the density of connections in the network is relatively low. Correspondingly, only a few entries of $\mathbf{A}$ are non-zero. Based on (3.6), least squares loss and sparseness penalty are used to identify the network topology,

$$\widehat{\mathbf{A}} = \underset{\mathbf{A} \in \mathbb{R}^{p \times p}}{\operatorname{argmin}} \|\mathbf{Y} - \mathbf{A}\mathbf{L}\|_F^2 + \sum_{i=1}^{p}\sum_{j=1}^{p} p_{\lambda_{ij}}(|a_{ij}|), \tag{3.8}$$

where $\|\cdot\|_F$ is the Frobenius norm. $p_{\lambda_{ij}}(\cdot)$ is the generic penalty function on each element and λ_{ij} is the corresponding tuning parameter. The first term of (3.8) fits the data to the model. The second term shrinks the elements of $\mathbf{A}$, removes the redundant connections and makes sure the sparseness of the resultant network. The tuning parameters control the extent of penalty so as to balance the fitting of the data and the sparseness

of the network.

For the LASSO penalty proposed by Tibshirani [15], $p_\lambda(|x|) = \lambda|x|$. LASSO penalty increases linearly with the magnitude of its argument, which leads to substantial biases in the estimate of large parameters [17]. To address this problem, Fan and Li [17] propose the SCAD penalty, which is a concave function defined by $p_{SCAD,\lambda}(0) = 0$ and for $|x| > 0$

$$p'_{SCAD,\lambda}(|x|) = \lambda \left\{ I(|x| \leq \lambda) + \frac{(a\lambda - |x|)_+}{(a-1)\lambda} I(|x| > \lambda) \right\}, \tag{3.9}$$

where $\lambda > 0$ and $a > 2$ are two tuning parameters. The notation θ_+ means the positive part of θ: $\theta_+ = \max(\theta, 0)$. (3.9) corresponds to the LASSO penalty when $a = \infty$. Often $a = 3.7$ is used [17]. Using the SCAD penalty, the network is identified by solving the following problem:

$$\underset{\mathbf{A} \in \mathbb{R}^{p \times p}}{\operatorname{argmin}} \|\mathbf{Y} - \mathbf{AL}\|_F^2 + n \sum_{i=1}^{p} \sum_{j=1}^{p} p_{SCAD,\lambda}(|a_{ij}|), \tag{3.10}$$

where we choose $\lambda_{ij} = \lambda$ for convenience. The corresponding vector form based on (3.7) is

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^{p^2}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{Z}\boldsymbol{\beta}\|^2 + n \sum_{i=1}^{p^2} p_{SCAD,\lambda}(|\beta_i|). \tag{3.11}$$

Zou [16] proposes another penalty called the adaptive LASSO to tackle the bias problem of LASSO. The adaptive LASSO penalty is in essence a weighted version of the LASSO penalty, where the weights are properly chosen. For our problem, the weights are defined to be $\tilde{w}_{ij} = |\tilde{a}_{ij}|^{-\gamma}$ for some $\gamma > 0$ and any root-n-consistent estimate $\tilde{\mathbf{A}} = (\tilde{a}_{ij})_{1 \leq i,j \leq p}$. Using the adaptive LASSO, we are seeking to solve the following problem:

$$\underset{\mathbf{A} \in \mathbb{R}^{p \times p}}{\operatorname{argmin}} \|\mathbf{Y} - \mathbf{AL}\|_F^2 + \lambda \sum_{i=1}^{p} \sum_{j=1}^{p} \tilde{w}_{ij} |a_{ij}|. \tag{3.12}$$

or

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^{p^2}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{Z}\boldsymbol{\beta}\|^2 + \lambda \sum_{i=1}^{p^2} \tilde{\omega}_i |\beta_i|, \tag{3.13}$$

where $\tilde{\omega}_i = |\tilde{\beta}_i|^{-\gamma}$ for some root-n-consistent initial estimate $\tilde{\boldsymbol{\beta}}$.

3.3 Properties

3.3.1 Oracle Properties of the Adaptive LASSO

Let $\boldsymbol{\beta}^* = vec(\mathbf{A}^*)$ be the true value of $\boldsymbol{\beta}$ in model (3.7) and denote by $\mathcal{B} = \{j : \beta_j^* \neq 0\}$ the true structure. Further assume that $|\mathcal{B}| = p_0 < p^2$, which means that the true model only depends on a subset of elements of $\mathbf{A}$. Using the terminology of Fan and Li [17], an estimation procedure has the oracle property if the estimate

from the procedure, denoted by $\widehat{\boldsymbol{\beta}}$, (i) can correctly identify the right subset, i.e. $\{j : \hat{\beta}_j \neq 0\} = \mathcal{B}$, and (ii) has the optimal estimation rate, $\sqrt{n}(\widehat{\boldsymbol{\beta}}_{\mathcal{B}} - \boldsymbol{\beta}^*_{\mathcal{B}}) \to_d \mathrm{N}(\mathbf{0}, \boldsymbol{\Sigma}^*)$, where $\boldsymbol{\Sigma}^*$ is the covariance matrix as knowing the true model structure.

The Oracle properties of adaptive LASSO and SCAD under the linear regression setting have been proved [17, 16, 116]. In the following, we show that based on our model (3.5), the Oracle properties of these two kinds of penalties are still reserved.

We first show that using the adaptive LASSO penalty, the inference of a gene regulatory network has the oracle properties. Given time course gene expression data, $\{x_{it}\}$, $i = 1, \ldots, p$, $t = 0, \ldots, n$, based on (3.13), the adaptive LASSO estimate is

$$\hat{\boldsymbol{\beta}}^{(n)} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{p^2}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{Z}\boldsymbol{\beta}\|^2 + \lambda_n \sum_{j=1}^{p^2} \tilde{\omega}_j |\beta_j|. \tag{3.14}$$

Similarly, define $\mathcal{B}_n = \{j : \hat{\beta}^{(n)}_j \neq 0\}$.

We show that under some mild conditions and with a proper choice of λ_n, the adaptive LASSO estimate based on the time series model (3.5) also enjoys the oracle properties.

Theorem 3.1. *Suppose that in* (3.5) *(A1)* $\mathbf{x}_k$ *are stationary for all* k*; (A2)* $\boldsymbol{\varepsilon}_k$ *are independent white noises with the nonsigular covariance matrix* $\mathrm{E}(\boldsymbol{\varepsilon}_k \boldsymbol{\varepsilon}_k^T) = \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}$ *and for some finite constant* c, $\mathrm{E}(|\varepsilon_{ik}\varepsilon_{jk}\varepsilon_{lk}\varepsilon_{mk}|) \leq c$ *for all* $1 \leq i, j, l, m \leq p$*. Further assume that (A3)* $\lambda_n/\sqrt{n} \to 0$ *and* $\lambda_n n^{(\gamma-1)/2} \to \infty$*. Then, the adaptive LASSO estimate satisfies:*

1. *Consistency in variable selection:* $\lim_{n\to\infty} \mathrm{P}(\mathcal{B}_n = \mathcal{B}) = 1$.

2. *Asymptotic normality:* $\sqrt{n}(\hat{\boldsymbol{\beta}}^{(n)}_{\mathcal{B}} - \boldsymbol{\beta}^*_{\mathcal{B}}) \to_d \mathrm{N}(\mathbf{0}, (\boldsymbol{\Gamma} \otimes \mathbf{I})^{-1}_{\mathcal{B},\mathcal{B}} (\boldsymbol{\Gamma} \otimes \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}})_{\mathcal{B},\mathcal{B}} (\boldsymbol{\Gamma} \otimes \mathbf{I})^{-1}_{\mathcal{B},\mathcal{B}})$,

where $\boldsymbol{\Gamma}$ *is the limit of* $\mathbf{L}\mathbf{L}^T/n$*, i.e.* $\frac{1}{n}\mathbf{L}\mathbf{L}^T \to_p \boldsymbol{\Gamma}$

Proof. The assumptions (A1) and (A2) imply that there exists nonsigular matrix $\boldsymbol{\Gamma}$ such that

$$\frac{1}{n}\mathbf{L}\mathbf{L}^T \to_p \boldsymbol{\Gamma}, \tag{3.15}$$

and

$$\frac{1}{\sqrt{n}}\mathbf{Z}^T\mathbf{e} \to_d \mathbf{W} = \mathrm{N}(\mathbf{0}, \boldsymbol{\Gamma} \otimes \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}), \tag{3.16}$$

as $n \to \infty$ [117, 118].

Similar to the proof of Zou [16], we first prove the asymptotic normality part. Let $\mathbf{u} = \sqrt{n}(\boldsymbol{\beta} - \boldsymbol{\beta}^*)$ and

$$\Psi_n(\mathbf{u}) = \left\| \mathbf{y} - \mathbf{Z}\left(\boldsymbol{\beta}^* + \frac{\mathbf{u}}{\sqrt{n}}\right) \right\|^2 + \lambda_n \sum_{j=1}^{p^2} \tilde{\omega}_j \left| \beta^*_j + \frac{u_j}{\sqrt{n}} \right|.$$

Let $\hat{\mathbf{u}}^{(n)} = \operatorname{argmin} \Psi_n(\mathbf{u})$. Then $\hat{\boldsymbol{\beta}}^{(n)} = \boldsymbol{\beta}^* + \frac{\hat{\mathbf{u}}^{(n)}}{\sqrt{n}}$. Note that $\Psi_n(\mathbf{u}) - \Psi_n(\mathbf{0}) = V^{(n)}(\mathbf{u})$, where

$$V^{(n)}(\mathbf{u}) = \mathbf{u}^T \left(\frac{1}{n}\mathbf{Z}^T\mathbf{Z}\right)\mathbf{u} - 2\mathbf{u}^T\frac{\mathbf{Z}^T\mathbf{e}}{\sqrt{n}} + \frac{\lambda_n}{\sqrt{n}}\sum_{j=1}^{p^2}\tilde{\omega}_j\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right).$$

From (3.15), we have

$$\frac{1}{n}\mathbf{Z}^T\mathbf{Z} = \frac{1}{n}\mathbf{L}\mathbf{L}^T \otimes \mathbf{I} \to_p \boldsymbol{\Gamma} \otimes \mathbf{I}.$$

The limit of the second term can be obtained by (3.16). For the limiting behavior of the third term, if $\beta_j^* \neq 0$, $\tilde{\omega}_j = |\tilde{\beta}_j|^{-\gamma} \to_p |\beta_j^*|^{-\gamma}$ and $\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) \to u_j \operatorname{sgn}(\beta_j^*)$. Since $\lambda_n/\sqrt{n} \to 0$, by Slutsky's theorem, we have $\frac{\lambda_n}{\sqrt{n}}\tilde{\omega}_j\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) \to_p 0$. If $\beta_j^* = 0$, $\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) = |u_j|$ and $\frac{\lambda_n}{\sqrt{n}}\tilde{\omega}_j = \frac{\lambda_n}{\sqrt{n}} n^{\gamma/2}(|\sqrt{n}\tilde{\beta}_j|)^{-\gamma}$ with $\sqrt{n}\tilde{\beta}_j = O_p(1)$. Therefore, by using Slutsky's theorem, we know that for every $\mathbf{u}$

$$V^{(n)}(\mathbf{u}) \to_d V(\mathbf{u}) = \begin{cases} \mathbf{u}_{\mathcal{B}}^T(\boldsymbol{\Gamma} \otimes \mathbf{I})_{\mathcal{B},\mathcal{B}}\mathbf{u}_{\mathcal{B}} - 2\mathbf{u}_{\mathcal{B}}^T\mathbf{W}_{\mathcal{B}} & \text{if } u_j = 0\ \forall j \notin \mathcal{B} \\ \infty & \text{otherwise.} \end{cases}$$

$V^{(n)}(\mathbf{u})$ and $V(\mathbf{u})$ are both convex and $V(\mathbf{u})$ has an unique minimum. Following the epi-convergence results of Geyer [119] and Knight and Fu [120], we have

$$\hat{\mathbf{u}}_{\mathcal{B}}^{(n)} \to_d (\boldsymbol{\Gamma} \otimes \mathbf{I})_{\mathcal{B},\mathcal{B}}^{-1}\mathbf{W}_{\mathcal{B}} \quad \text{and} \quad \hat{\mathbf{u}}_{\mathcal{B}^c}^{(n)} \to_d \mathbf{0}. \tag{3.17}$$

Note that $\mathbf{W}_{\mathcal{B}} = \mathrm{N}(\mathbf{0}, (\boldsymbol{\Gamma} \otimes \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}})_{\mathcal{B},\mathcal{B}})$, the asymptotic normality part is proved.

For the consistency part, for any $j \in \mathcal{B}$, the asymptotic normality results indicates that $\hat{\beta}_j^{(n)} \to_p \beta_j^*$. Therefore, $\mathrm{P}(j \in \mathcal{B}_n) \to 1$. Then we only need to show that $\forall j' \notin \mathcal{B}$, $\mathrm{P}(j' \in \mathcal{B}_n) \to 0$. For the event $j' \in \mathcal{B}_n$, by KKT conditions, we see that

$$2\mathbf{z}_{j'}^T(\mathbf{y} - \mathbf{Z}\hat{\boldsymbol{\beta}}^{(n)}) = -\lambda_n\tilde{\omega}_{j'}\operatorname{sgn}(\hat{\beta}_{j'}^{(n)}),$$

where $\mathbf{z}_{j'}$ is the j''th column of $\mathbf{Z}$. Note that $\lambda_n\tilde{\omega}_{j'}/\sqrt{n} = \frac{\lambda_n}{\sqrt{n}} n^{\gamma/2}\frac{1}{|\sqrt{n}\tilde{\beta}_{j'}|^{\gamma}} \to_p \infty$. However,

$$2\frac{\mathbf{z}_{j'}^T(\mathbf{y} - \mathbf{Z}\hat{\boldsymbol{\beta}}^{(n)})}{\sqrt{n}} = 2\frac{\mathbf{z}_{j'}^T\mathbf{Z}\sqrt{n}(\boldsymbol{\beta}^* - \hat{\boldsymbol{\beta}}^{(n)})}{n} + 2\frac{\mathbf{z}_{j'}^T\mathbf{e}}{\sqrt{n}}$$

By (3.17), we know that $2\frac{\mathbf{z}_{j'}^T\mathbf{Z}\sqrt{n}(\boldsymbol{\beta}^* - \hat{\boldsymbol{\beta}}^{(n)})}{n} \to_d$ some normal distribution random variable and $2\frac{\mathbf{z}_{j'}^T\mathbf{e}}{\sqrt{n}} \to_d$

$\mathrm{N}(\mathbf{0}, 4(\mathbf{\Gamma} \otimes \mathbf{\Sigma})_{j',j'})$. Thus,

$$\mathrm{P}\left(j' \in \mathcal{B}_n\right) \leq \mathrm{P}\left(2\mathbf{z}_{j'}^T(\mathbf{y} - \mathbf{Z}\hat{\boldsymbol{\beta}}^{(n)}) = \lambda_n \tilde{w}_{j'} \mathrm{sgn}(\hat{\beta}_j^{(n)})\right) \to 0.$$

□

Remarks: (1) In Theorem 3.1, $\boldsymbol{\varepsilon}_k$ are required to satisfy the assumption (A2). We can alternatively assume that $\boldsymbol{\xi}$ satisfy the conditions in (A2), which can imply $\boldsymbol{\varepsilon}_k$ fulfill those conditions. (2) Assumption (A1) requires that $\mathbf{x}_k$ is stationary. This can be guaranteed by the stability condition: roots of

$$\det(\mathbf{I} - \mathbf{A}z) = 0$$

lie outside the unit circle. Or equivalently, the modulus of eigenvalues of $\mathbf{A}$ are less than 1.

3.3.2 Oracle Properties of SCAD

Using the SCAD penalty to select the variables in the linear regression setting, Zou and Li [116] proposes a local linear approximation (LLA) method which is an improvement of the local quadratic approximation (LQA) method in Fan and Li [17]. The LLA locally approximates the SCAD penalty by a symmetric linear function. Considering (3.11), for any $\beta_j^{(0)}$, by the Taylor expansion, $p_{SCAD,\lambda}(|\beta_j|)$ can be approximated in a neighborhood of $|\beta_j^{(0)}|$ as follows:

$$p_{SCAD,\lambda}(|\beta_j|) \approx p_{SCAD,\lambda}(|\beta_j^{(0)}|) + p'_{SCAD,\lambda}(|\beta_j^{(0)}|)(|\beta_j| - |\beta_j^{(0)}|),$$

where $p'_{SCAD,\lambda}(\cdot)$ is defined as (3.9).

Under the linear regression setting, Zou and Li [116] studies the one-step LLA method, i.e. solving the SCAD penalized problem by replacing SCAD penalty with its LLA, and shows that the one-step estimate enjoys the oracle properties. In our problem of time series setting, similarly, using the LLA of SCAD penalty, we define the one-step estimate as

$$\hat{\boldsymbol{\beta}}_{ose}^{(n)} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{p^2}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{Z}\boldsymbol{\beta}\|^2 + n \sum_{j=1}^{p^2} p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|)|\beta_j|, \tag{3.18}$$

where $\boldsymbol{\beta}^{(0)}$ is any root-n-consistent initial estimate. In the following theorem, the initial value $\boldsymbol{\beta}^{(0)}$ can be taken as the ordinary least squares estimate. In the following, we show that the one-step estimate, $\hat{\boldsymbol{\beta}}_{ose}^{(n)}$, also satisfies the oracle properties under some mild conditions similar to those in Theorem 3.1. In the following, define $\mathcal{B}_n^{ose} = \{j : \hat{\beta}_{ose,j}^{(n)} \neq 0\}$.

Theorem 3.2. *Suppose conditions (A1) and (A2) in Theorem 3.1 also hold here. If (A3)* $\sqrt{n}\lambda_n \to \infty$ *and* $\lambda_n \to 0$*, then the one-step SCAD estimate* $\hat{\boldsymbol{\beta}}_{ose}^{(n)}$ *satisfy:*

1. *Consistency in variable selection:* $\lim_{n\to\infty} \mathrm{P}(\mathcal{B}_n^{ose} = \mathcal{B}) = 1$.

2. *Asymptotic normality:* $\sqrt{n}(\hat{\boldsymbol{\beta}}_{ose,\mathcal{B}}^{(n)} - \boldsymbol{\beta}_{\mathcal{B}}^*) \to_d \mathrm{N}(\mathbf{0}, (\boldsymbol{\Gamma}\otimes\mathbf{I})_{\mathcal{B},\mathcal{B}}^{-1}(\boldsymbol{\Gamma}\otimes\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}})_{\mathcal{B},\mathcal{B}}(\boldsymbol{\Gamma}\otimes\mathbf{I})_{\mathcal{B},\mathcal{B}}^{-1})$,

where $\boldsymbol{\Gamma}$ *is defined the same as in Theorem 3.1.*

Proof. The proof is similar to that in Theorem 3.1 and Zou's paper [116]. First, we prove asymptotic normality part.

Let $\mathbf{u} = \sqrt{n}(\hat{\boldsymbol{\beta}}_{ose}^{(n)} - \boldsymbol{\beta}^*)$ and

$$\begin{aligned}\Psi_n(\mathbf{u}) = &\left\|\mathbf{y} - \mathbf{Z}\left(\boldsymbol{\beta}^* + \frac{\mathbf{u}}{\sqrt{n}}\right)\right\|^2 \\ &+ n\sum_{j=1}^{p^2} p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|)\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right|.\end{aligned}$$

Let $\hat{\mathbf{u}}^{(n)} = \operatorname{argmin}\Psi_n(\mathbf{u})$. Then $\hat{\boldsymbol{\beta}}_{ose}^{(n)} = \boldsymbol{\beta}^* + \frac{\hat{\mathbf{u}}^{(n)}}{\sqrt{n}}$. Let $V^{(n)}(\mathbf{u}) = \Psi_n(\mathbf{u}) - \Psi_n(\mathbf{0})$, then

$$\begin{aligned}V^{(n)}(\mathbf{u}) = &\mathbf{u}^T\left(\frac{1}{n}\mathbf{Z}^T\mathbf{Z}\right)\mathbf{u} - 2\mathbf{u}^T\frac{\mathbf{Z}^T\mathbf{e}}{\sqrt{n}} \\ &+ \sum_{j=1}^{p^2}\sqrt{n}p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|)\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right).\end{aligned}$$

Because of (A1) and (A2), we know that $\mathbf{Z}^T\mathbf{Z}/n \to_p \boldsymbol{\Gamma}\otimes\mathbf{I}$ and $\mathbf{Z}^T\mathbf{e}/\sqrt{n} \to_d \mathbf{W} = \mathrm{N}(\mathbf{0}, \boldsymbol{\Gamma}\otimes\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}})$. Thus, the limit of the first two terms can be obtained. Now, we study the limit behavior of the third term. If $\beta_j^* \neq 0$, we see that $\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) \to u_j\mathrm{sgn}(\beta_j^*)$ and if $\beta_j^* = 0$, $\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) = |u_j|$. By conditions (A1) and (A2), the ordinary lest squares estimate $\boldsymbol{\beta}^{(0)}$ has the property [117, 118]

$$\sqrt{n}(\boldsymbol{\beta}^{(0)} - \boldsymbol{\beta}^*) \to_d \mathrm{N}(\mathbf{0}, \boldsymbol{\Gamma}^{-1}\otimes\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}). \tag{3.19}$$

Thus, if $\beta_j^* \neq 0$, we have $|\beta_j^{(0)}| \to_p |\beta_j^*| > 0$. Note that $p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|) = 0$ if $|\beta_j^{(0)}| > a\lambda_n$, $(a = 3.7)$ and $\lambda_n \to 0$. We have $\sqrt{n}p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|)\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) \to_p 0$. When $\beta_j^* = 0$, $\sqrt{n}p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|)\sqrt{n}$ $\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) = 0$ if $u_j = 0$. For $u_j \neq 0$, note that $p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|) = \lambda_n$ for all $|\beta_j^{(0)}| < \lambda_n$. By (3.19), $\beta_j^{(0)} = O_p(1/\sqrt{n})$, $\sqrt{n}\lambda_n \to \infty$ ensures that $\sqrt{n}p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|)\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) = |u_j|\sqrt{n}\lambda_n$ with probability tending to one. Therefore, when $\beta_j^* = 0$ and $u_j \neq 0$, $\sqrt{n}p'_{SCAD,\lambda_n}(|\beta_j^{(0)}|)\sqrt{n}\left(\left|\beta_j^* + \frac{u_j}{\sqrt{n}}\right| - |\beta_j^*|\right) \to_p$ ∞. Based on the discussions above, for every $\mathbf{u} = (\mathbf{u}_{\mathcal{B}}^T, \mathbf{u}_{\mathcal{B}^c}^T)^T$, we know that

$$\begin{aligned}V^{(n)}(\mathbf{u}) &\to_d V(\mathbf{u}) \\ &= \begin{cases}\mathbf{u}_{\mathcal{B}}^T(\boldsymbol{\Gamma}\otimes\mathbf{I})_{\mathcal{B},\mathcal{B}}\mathbf{u}_{\mathcal{B}} - 2\mathbf{u}_{\mathcal{B}}^T\mathbf{W}_{\mathcal{B}} & \text{if } \mathbf{u}_{\mathcal{B}^c} = 0 \\ \infty & \text{otherwise.}\end{cases}\end{aligned}$$

$V^{(n)}(\mathbf{u})$ and $V(\mathbf{u})$ are both convex and $V(\mathbf{u})$ has an unique minimum. Following the epi-convergence results of Geyer [119] and Knight and Fu [120], we have

$$\hat{\mathbf{u}}_{\mathcal{B}}^{(n)} \to_d (\mathbf{\Gamma} \otimes \mathbf{I})_{\mathcal{B},\mathcal{B}}^{-1} \mathbf{W}_{\mathcal{B}} \quad \text{and} \quad \hat{\mathbf{u}}_{\mathcal{B}^c}^{(n)} \to_d \mathbf{0}. \tag{3.20}$$

Note that $\mathbf{W}_{\mathcal{B}} = \mathrm{N}(\mathbf{0}, (\mathbf{\Gamma} \otimes \mathbf{\Sigma}_{\boldsymbol{\varepsilon}})_{\mathcal{B},\mathcal{B}})$, the asymptotic normality part is proved.

Next, we prove the consistency part. For any $j \in \mathcal{B}$, the asymptotic normality results indicates that $\hat{\beta}_{ose,j}^{(n)} \to_p \beta_j^*$, which implies $\mathrm{P}(j \in \mathcal{B}_n^{ose}) \to 1$. Then it suffices to prove that for any $j' \notin \mathcal{B}$, $\mathrm{P}(j' \in \mathcal{B}_n^{ose}) \to 0$. Assuming $j' \in \mathcal{B}_n^{ose}$, by KKT condition of (3.18), we must have

$$2\frac{\mathbf{z}_{j'}^T(\mathbf{y} - \mathbf{Z}\hat{\boldsymbol{\beta}}_{ose}^{(n)})}{\sqrt{n}} = -\sqrt{n}p'_{SCAD,\lambda_n}(|\beta_{j'}^{(0)}|)\mathrm{sgn}(\hat{\beta}_{ose,j'}^{(n)}), \tag{3.21}$$

where $\mathbf{z}_{j'}$ is the j''th column of $\mathbf{Z}$. In the aforementioned discussions, we have shown that when $\beta_{j'}^* = 0$, the right-hand side goes to ∞ in probability. However, the left-hand side can be written as

$$2\frac{\mathbf{z}_{j'}^T\mathbf{Z}\sqrt{n}(\boldsymbol{\beta}^* - \hat{\boldsymbol{\beta}}_{ose}^{(n)})}{n} + 2\frac{\mathbf{z}_{j'}^T\mathbf{e}}{\sqrt{n}}.$$

By (3.20), the first term converges in distribution to some normal, and so does the second term by (3.16). Therefore,

$$\mathrm{p}(j' \in \mathcal{B}_n^{ose}) \leq \mathrm{P}(\text{KKT condition (3.21) holds}) \to 0.$$

□

We have the same remarks for Theorem 3.2 as for Theorem 3.1.

3.4 Inference Algorithms

The inferred GRNs are obtained by solving the weighted LASSO optimization problems (3.14) and (3.18). The fastest approach to this kind of problem is coordinated descent method by Friedman et al. [114], which one-at-a time optimizes one variable by making use of soft-thresholding operator [114]. To avoid the storage of very large matrix in the case of high dimensional data, we use the corresponding matrix forms of (3.14) and (3.18) instead and solve the regulatory matrix $\mathbf{A}$ row by row.

For adaptive LASSO penalized problems, we fix $\gamma = 0.5$ and the solution path, i.e., solutions for several λ values, is computed. λ values are picked in the interval from $\lambda_{\max} = \max_{i,j}\{2|\mathbf{L}_j\mathbf{Y}_i^T|/\tilde{\omega}_{ij}\}$, the minimum λ resulting in zero solution where $\mathbf{L}_i$ and $\mathbf{Y}_i$ are the ith rows of matrix $\mathbf{L}$ and $\mathbf{Y}$ in (3.6), to $\lambda_{\min} = 0.01\lambda_{\max}$. We use the strategy of warm starts [114], i.e, the initial value for computing the solution of current λ is the solution of the previous λ. For SCAD penalized problems, we fix $a = 3.7$, and compute the solutions of several λ's values picked in the interval from $\lambda_{\min} = \min_j |\beta_j^{(0)}|/a$ to $\lambda_{\max} = \max\{\max_j |\beta_j^{(0)}|, \max_{i,j} 2|\mathbf{L}_j\mathbf{Y}_i|/n\}$.

λ in both methods is a tuning parameter, different values of which may lead to different solutions. Methods used to determine the value of λ include cross-validation and information criterion, e.g. AIC and BIC. In contrast to selecting a specific value for λ which corresponds to a determinant network topology, a method called "stability selection" recently proposed by Meinshausen and Buhlmann [115] finds a network with a probability for each edge. Stability selection performs the variable selection methods, .e.g. LASSO or sparse penalized regression, many times, resampling the data in each run and computing the frequencies with which each variable is selected across these runs. It has been used with linear regression method to infer GRNs from steady-state gene expression data in Haury et al. [100] and has shown perspective effectiveness. In this study, we adapt the stability selection method to finding probable GRNs from time-course gene expression data with sparse penalized methods. Given a time course gene expression data set $\mathbf{X} \in \mathbb{R}^{p\times n}$ with rows representing genes and columns representing time points, for each method and a specific $\lambda \in \Lambda$, the stability selection is as follows,

1. Use moving block bootstrap to draw N bootstrap samples $\mathbf{X}^{*(k)}$, $k = 1, \ldots, N$, from time-course gene expression data.

2. Use the sparse penalized method to infer a network $\hat{\mathbf{A}}_\lambda^{*(k)}$ from each bootstrap sample.

3. Compute the frequencies for each edge (i, j), i.e., from gene j to gene i, in the network

$$\Pi_\lambda(i,j) = \frac{\sharp\{k : \hat{\mathbf{A}}_\lambda^{*(k)} \text{ is stable and } \hat{\mathbf{A}}_\lambda^{*(k)}(i,j) \neq 0\}}{\sharp\{k : \hat{\mathbf{A}}_\lambda^{*(k)} \text{ is stable}\}}.$$

For a set of $\lambda \in \Lambda$, the probability of each edge in the inferred network is

$$\Pi(i,j) = \max_{\lambda\in\Lambda} \Pi_\lambda(i,j).$$

The network topology can be obtained by setting a threshold, edges with probabilities less than which are considered nonexistent. This study only focuses on giving a list of edges with probabilities. The selection of threshold is not discussed here. When multiple time-course gene expression data sets are available, we use the above mentioned method to infer a list of edges with probabilities from each single data set and then obtain the probability for each edge by simply taking average of those results across data sets.

Since we use the time series data, the moving block bootstrap method is employed in the first step. In the moving block bootstrap, with block length b, the data is split into $n - b + 1$ blocks: block j consists of observations j to $j + b - 1$, $j = 1, \ldots, n - b + 1$. $\lceil n/b \rceil$ blocks are randomly draw from $n - b + 1$ with replacement and are aligned in the order they are picked to form a bootstrap sample.

Note that in the third step, we only use the stable estimated networks to calculate the frequencies of edges. From the remarks of our theorems, we know that the system stability requirement can help improve the inference accuracy. To see the existence of stable estimated network, note that in our previous work

Table 3.1: AUROC and AUPR of each method for DREAM3 size-10 networks.

Size 10	Ecoli 1		Ecoli 2		Yeast 1		Yeast 2		Yeast 3	
	AUROC	AUPR	AUROC	RUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR
LASSO	0.5178	0.1276	0.5151	0.1637	0.5288	0.1058	0.6314	0.4084	0.5130	0.3129
Adaptive LASSO	0.6283	0.1546	0.5431	0.1741	0.6013	0.1416	0.6815	0.4992	0.5625	0.3829
SCAD	0.7100	0.2266	0.5422	0.2345	0.5750	0.2262	0.7108	0.5085	0.5348	0.3161

[109], a sufficient condition has been proposed for the stability: the ℓ_1 norm of each row of $\mathbf{A}$ is less than 1. It is observed that for both adaptive LASSO and SCAD penalized methods, the ℓ_1 norm of each row of $\mathbf{A}$ tends to 0 as λ becomes large enough. Therefore, there exists λ values giving stable networks. Unlike other papers [105, 109] using a sufficient condition as explicit constraints to ensure the stability, this paper uses the stability condition to help finding the edge probability. The Oracle properties of these penalties indicate that the unconstrained optimization can correctly infer the network with a proper regularization parameter. The unconstrained optimization is simpler than constrained ones and for our unconstrained problem, there exists a very efficient algorithm. The existing stability constraints [105, 109] are very strong and only sufficient conditions, which the true stable network may not be satisfied with.

3.5 Applications

Since the adaptive LASSO and SCAD penalized methods enjoy the Oracle properties, they are expected to have better performances than LASSO. To demonstrate their effectiveness, they are applied to inferring GRNs from *in silico* time-course data and real experimental time-course data, respectively. The *in silico* data are time-course data in DREAM3 challenges for networks of size 10 and 100. The real data are the time-course data sets of *E. coli* SOS DNA repair network and a cell cycle regulatory subnetwork in *S. cerevisiae*.

3.5.1 DREAM3 Networks

The performances of the methods are evaluated on a number of GRNs inferred from DREAM3 *in silico* data sets. We consider the challenges for networks of size 10 and 100. For the size-10 network, there are five networks (E. coli 1, E. coli 2, Yeast 1, Yeast 2 and Yeast 3), each of which has 4 time series consisting of 21 time points. For the size-100 network, we consider 2 networks (E. coli 1 and Yeast 1), each of which has 46 time series consisting of 21 time points. These data were generated by simulating a thermodynamic model for gene expression to which the noises were added. The multiple time series were obtained by assigning different initial values to the thermodynamic model. The network typologies are extracted from currently accepted GRNs and have varying patterns of sparsity and topology structures.

For each network challenge, the adaptive LASSO and SCAD penalized methods are applied to those multiple data sets and the above mentioned stability selection procedure is used to infer a network in which each edge has a probability. Varying the threshold for the edge and comparing the resulted network with

Table 3.2: AUROC and AUPR of each method for Ecoli 1 and Yeast 1 of DREAM3 size-100 networks.

Size 100	Ecoli 1		Yeast 1	
	AUROC	AUPR	AUROC	AUPR
LASSO	0.6463	0.0278	0.5174	0.0184
Adaptive LASSO	0.6997	0.0308	0.5310	0.0194
SCAD	0.7081	0.0366	0.5775	0.0206

the gold standard network topology, the areas under the ROC curve (AUROC) and precision-recall curve (AUPR) are computed. As the number of time points is larger than the number of genes in the size-10 network, the initial estimates for the adaptive LASSO and SCAD methods are least squares estimates. For the size-100 network, there are less time points compared with the number of genes and the pseudoinverse method is used to solve the least squares problem and provide the initial estimates for the adaptive LASSO and SCAD methods. In the implementation, we set the number of bootstrap samples $N = 30$ with block length $b = 15$ for the size-10 networks and set the number of bootstrap samples $N = 10$ with block length $b = 15$ for the size-100 networks. We choose smaller bootstrap sample size for the size-100 networks to save computation time as the network size is relatively large and there are 46 time-series data sets.

The AUROCs and AUPRs of adaptive LASSO and SCAD methods and those from LASSO for DREAM3 size-10 and size-100 networks are reported in the Table 3.1 and 3.2, respectively. Table 3.1 shows that all three methods have better performances than random guess, as all AUROCs are larger than 0.5. A comparison among these three methods in Table 3.1 indicates that adaptive LASSO and SCAD penalized methods outperform the LASSO methods in both AUROC and AUPR. More specifically, for size-10 Ecoli 1 and Yeast 2, SCAD penalized method outperforms LASSO significantly. For size-10 Yeast 1 and Yeast 3, adaptive LASSO has the largest AUROC among the three and it has higher AUPRs than LASSO. The results for the size-100 Ecoli 1 and Yeast 1 networks are shown in Table 3.2 and give the similar conclusions, i.e., all methods are better than random guess and adaptive LASSO and SCAD penalized methods outperform LASSO in both AUROC and AUPR. It is noticeable that AUROCs of these methods for size-100 networks are comparable to those for size-10 networks and SCAD outperforms LASSO significantly in AUROC for size-100 Ecoli 1. The relatively low values of AUPR is because of the low density of connectivity in the size-100 networks. All the results demonstrate the effectiveness of adaptive LASSO and SCAD penalties which have the Oracle properties as we have proven in the theorems.

3.5.2 *E. coli* SOS Network

In this example, we apply the proposed methods to identify the real gene regulatory network, *E. coli* SOS DNA repair system as shown in Figure 3.1. This network is in charge of repairing the DNA after some damage happens. In the normal state, LexA acts as the master repressor of many genes. When a damage occurs, RecA acts as a sensor and binds to single-stranded DNA to sense the damage and mediates LexA

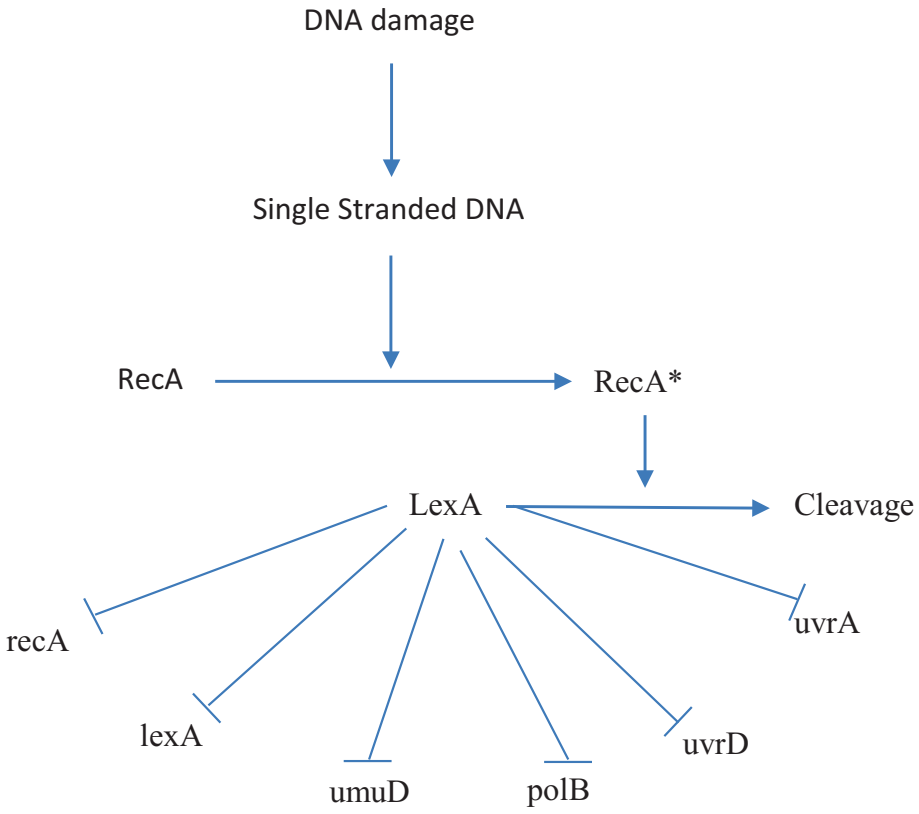


Figure 3.1: SOS DNA repair pathway in *E. coli*.

autocleavage. The repressions of the SOS genes are halted by the drop in LexA levels. The SOS genes are activated and start to repair the damages. When the repair is done, the dropping of RecA level stops mediating LexA autocleavage. Then, LexA accumalates and represses the SOS genes and the cell goes back to the normal state.

Four gene expression data sets of SOS DNA network are downloaded from the Uri Alon lab,[1] which are taken from four experiments for various UV light intensities (Experiment 1 and 2: $5Jm^{-2}$, Experiment 3 and 4: 20 Jm^{-2}). Each data set contain 8 genes and their measurements at 50 time points. As other literature did, e.g. [1, 121, 122, 123], only 6 genes, i.e., uvrD, lexA, umuD, recA, uvrA and polB are considered because they are well studied and the gold standard network of these genes are illustrated in Table 3.3. Details of the gold standard can be found in [1]. In this study, we do not consider the signs and the self-regulations.

We apply the adaptive LASSO and SCAD penalized methods to infer GRNs from experiment 1 and 2 data sets and experiment 3 and 4 data sets, respectively and compare with the results from LASSO. The initial estimates for adaptive LASSO and SCAD are least squares estimates. The number of bootstrap samples is $N = 50$ with block length $b = 25$. The AUROC and AUPR of each method for the SOS network are shown in Table 3.4, from which we can see that all methods are better than random guess and adaptive LASSO

[1] http://www.weizmann.ac.li/mcb/UriAlon.

Table 3.3: Gold standard of SOS network, collected from literature [1].

	uvrD	lexA	umuD	recA	uvrA	polB
uvrD	0	-1	-1	1	1	0
lexA	0	-1	-1	1	0	0
umuD	0	-1	-1	1	0	0
recA	0	-1	-1	1	0	0
uvrA	1	-1	-1	1	0	0
polB	0	-1	-1	1	0	0

Table 3.4: AUROC and AUPR of each method for E. coli SOS network.

	Exp. 1 and 2		Exp. 3 and 4	
	AUROC	AUPR	AUROC	AUPR
LASSO	0.6199	0.7634	0.5249	0.6777
Adaptive LASSO	0.6493	0.7762	0.6176	0.7970
SCAD	0.7489	0.8200	0.7172	0.8008

and SCAD penalized methods outperform the LASSO. SCAD penalized method has the best performance among these three. These results are consistent with those of *in silico* data and demonstrate the effectiveness of the methods.

3.5.3 *S. cerevisae* Cell Cycle Subnetwork

A cell cycle regulatory subnetwork in *S. cerevisae* is inferred by proposed methods from experimental microarray data. As in [124], the subnetwork considered consists of 27 genes including 10 genes for producing transcription factors (ace2, fkh1, swi4, swi5, mbp1, swi6, mcm1, fkh2, ndd1, yox1) and 17 genes for producing cyclin and cyclin/CDK regulatory proteins (cln1, cln2, cln3, cdc20, clb1, clb2, clb4, clb5, clb6, sic1, far1, spo12, apc1, tem1, gin4, swe1 and whi5). The corresponding microarray we use are from [125], collected by alpha factor arrest method. There are 27 genes and 18 time points in the data.

In order to demonstrate the effectiveness of proposed methods, the inferred results are compared with the interaction network of the chosen 27 genes, drawn from BioGRID database [126]. The network in the database has 112 interactions, not including the self-regulations, and we take it as the gold standard regulatory network. Each method is applied to the data with the number of bootstrap samples $N = 30$ and block size $b = 10$. Since the number of genes is greater than the number of time points, the initial estimates for adaptive LASSO and SCAD methods are obtained by solving the least squares problem with pseudoinverse method. The performances of these methods are shown in Table 3.5. It can be seen that all methods are better than random guess and adaptive LASSO and SCAD penalized methods outperform the LASSO. For this data, adaptive LASSO penalized method has the largest values in both AUROC and AUPR. Although the gold standard from the database may be incomplete or contain some errors, it still can demonstrate the

Table 3.5: AUROC and AUPR of each method for S. cerevisae cell cycle subnetwork.

	AUROC	AUPR
LASSO	0.5083	0.1710
Adaptive LASSO	0.5418	0.2040
SCAD	0.5384	0.1841

effectiveness of the adaptive LASSO and SCAD penalized methods to some degree.

3.6 Conclusions

In this study, after discretizing an ODE model of the GRN, an auto-regressive model is obtained, base on which the network topology inference task is achieved by identifying the non-zero elements in the regulatory coefficient matrix.

This paper aims at uncovering the network topology from time course gene expression data. To solve the task and improve the accuracy, two important properties of GRNs, sparseness and stability, are proposed to be used. The adaptive LASSO and SCAD penalties are employed to result in sparseness: making many irrelevant elements in the regulatory matrix become zero. These penalties have been proved in literature to be effective and have the Oracle properties in the ordinary linear regression setting. With some mild regularity conditions, we have proved that the Oracle properties are preserved when they are used in our time series model. Therefore, the proposed methods are theoretically guaranteed. An adapted stability selection strategy has been used in this study to infer a GRN in which each edge is assigned a probability. In this strategy, the moving block bootstrap method is used to generate the bootstrap samples and the system stability conditions are used to help improving the accuracy of calculating probabilities of the edges. Unlike other literature that use a stability sufficient condition as an optimization constraint, we only make use of the stability property as a criterion to help finding the network. As a result, we only need to solve an unconstrained optimization problem with the advantage of the existing efficient coordinate descent algorithm.

The effectiveness of the proposed methods are demonstrated by applications to the DREAM3 *in silico* data of size 10 and 100 network challenges and the real experimental data of *E. coli* SOS network and *S. cerevisae* cell cycle subnetwork. In *in silico* examples, the results show that all the methods are better than random guess and adaptive LASSO and SCAD penalized methods outperform the LASSO. The results of real data applications have the similar conclusions which confirm the effectiveness of these penalties that enjoy the Oracle properties.

Although some results of the proposed methods are not good enough, it may be because of the nonlinearities and inherent noises of GRNs and limited information of the time-course data. Ways to improve their performances include using kernel methods to tackle with the nonlinearity, incorporating with other source of biological data and developing robust methods to deal with the large noises. The methods studied here

provide a good foundation, as all these ways either take them as components of powerful integrated systems or based on them, make further extensions.

Acknowledgement

This study is supported by Natural Science and Engineering Research Council of Canada (NSERC).

Chapter 4

A Group LASSO-Based Method for Robustly Inferring Gene Regulatory Networks from Multiple Time-Course Datasets

Published as: L.Z. Liu, F.X. Wu, and W.J. Zhang, "A group lasso-based method for robustly inferring gene regulatory networks from multiple time-course datasets," *BMC Systems Biology*, March 2014, in press [127]. This work is an extension to our conference paper: L.Z. Liu, F.X. Wu, and W.J. Zhang, "Robust inference of gene regulatory networks from multiple microarray datasets," in *Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on*, pp. 544–547, Dec 2013 [128].

In the previous chapter, based on an autoregressive model, the statistical properties of sparse penalties, adaptive LASSO and SCAD, for the inference of GRNs from time-course gene expression data are analyzed. Although, the network topology can be asymptotically reconstructed, the limited number of observations and the inherent noises make the inference accuracy relatively low. As more and more gene expression data are collected, integrating multiple data sets should be able to improve the accuracy. However, the time-course gene expression data cannot be simply combined as there is no temporal relationships between different datasets.

In this chapter, a novel method, Huber group LASSO, is proposed to infer the GRNs from multiple time-course gene expression datasets as well as to take the robustness to large errors and outliers into account. To solve the optimization problem involved in the proposed method, an efficient algorithm is developed and its convergence is also analyzed. Simulation and real data examples show the effectiveness of the proposed method. This chapter and previous chapter accomplish Objective 2 of this thesis.

Abstract

As an abstract mapping of the gene regulations in the cell, gene regulatory network is important to both biological research study and practical applications. The reverse engineering of gene regulatory networks from microarray gene expression data is a challenging research problem in systems biology. With the development of biological technologies, multiple time-course gene expression datasets might be collected for a specific gene network under different circumstances. The inference of a gene regulatory network can be improved by

integrating these multiple datasets. It is also known that gene expression data may be contaminated with large errors or outliers, which may affect the inference results.

A novel method, Huber group LASSO, is proposed to infer the same underlying network topology from multiple time-course gene expression datasets as well as to take the robustness to large error or outliers into account. To solve the optimization problem involved in the proposed method, an efficient algorithm which combines the ideas of auxiliary function minimization and block descent is developed. A stability selection method is adapted to our method to find a network topology consisting of edges with scores. The proposed method is applied to both simulation datasets and real experimental datasets. It shows that Huber group LASSO outperforms the group LASSO in terms of both areas under receiver operating characteristic curves and areas under the precision-recall curves.

The convergence analysis of the algorithm theoretically shows that the sequence generated from the algorithm converges to the optimal solution of the problem. The simulation and real data examples demonstrate the effectiveness of the Huber group LASSO in integrating multiple time-course gene expression datasets and improving the resistance to large errors or outliers.

4.1 Background

A Gene regulatory network (GRN) consists of a set of genes and regulatory relationships among them. Tremendous amount of microarray data that measure expression levels of genes under specific conditions are obtained from experiments. It is a challenging problem in systems biology to reconstruct or "reverse engineer" GRNs by aiming at retrieving the underlying interaction relationships between genes from microarray data. Various approaches have been developed to infer GRNs from microarray data. Most of them can be classified into two categories: parametric or model-based methods and nonparametric or dependency-measure-based methods. Commonly used models include ordinary differential equations [64], Gaussian graphical models [13] and Bayesian networks [92]. Dependency measures include partial correlation coefficient [129], mutual information [130], and z-score [131].

The reconstruction of GRN is a non-trivial problem. On the one hand, the number of possible network topologies grows exponentially as the number of genes increases. On the other hand, the information in the microarray data is quite limited. The data contain a lot of inherent noises generated from the devices or the experiment processes. For large-scale networks, the number of observations is usually much less than that of genes, also known as "dimensionality problem" [13, 14]. The lack of observations and the high dimensionality of the data prohibit the direct application of traditional methods and make the inference task extremely challenging.

As more and more microarray datasets on the same species are produced from different laboratories, their integration leads to more robust and more reliable results. The methods that integrate multiple datasets could synergize the strength of each dataset and either infer a more accurate network if all the integrated

datasets are in high qualities or infer a robust network which is better than the worse one that is from a single dataset. However, multiple time-course datasets can not be simply combined as one dataset as there is no temporal dependencies between the datasets. Wang et al. [54] proposes a linear programming framework to integrate multiple time-course gene expression data to infer a network topology that is most consistent to all datasets. In their method, the regulatory strengths between genes is assumed to be the same across all datasets. However, different datasets may be produced under different circumstances, which may result in different regulatory strength between genes. Another problem is that the value of the tuning parameter in their method, which controls the degree of sparsity of the inferred network, is only determined intuitively. Chen et al. [132] infer one GRN from each time-course data separately, and combine edges of inferred GRNs using a strategy similar to majority vote. For this method, using each dataset separately in the inference process may miss the opportunity of taking advantage of information in other datasets and the tuning parameter is also determined intuitively.

This study focuses on inferring the topologies of GRNs from multiple time-course gene expression datasets based on an autoregressive model. We assume that one GRN corresponds to one dataset and these GRNs share the same topology across all datasets. By assigning the parameters representing the regulatory strengths of the same edge into the one group, the group LASSO [133] can be applied to find the sparse network topology. Microarray data typically contain noises and outliers, which could severely affect the quality of inferred results. Rosset and Zhu [134] proposes a robust version of LASSO by replacing the squared error loss of LASSO with Huber loss. We propose to use the Huber loss to extend the group LASSO such that the new method, Huber group LASSO, is more resistant to the large noises and outliers.

To solve the Huber group LASSO, a new algorithm is developed in our previous work [128], which combines the idea of auxiliary function minimization [135] and the block coordinates descent method [136]. The proposed algorithm is efficient and can also be adapted for solving the group LASSO problem without the orthogonality restriction. In this study, we analyze the convergence of our proposed algorithm and show that the sequence the algorithm generated indeed converges to the optimal solution of the Huber group LASSO problem. Instead of picking a specific value for the tuning parameter which corresponds to a determinant network topology as in our previous work [128], in this study, we adapt the "stability selection" [115] strategy to our method to find a network consisting of edges with probabilities or scores. The Huber group LASSO is applied to both simulation data and real experimental data and its performances are compared with those of the group LASSO in terms of areas under the receiver operating characteristic (AUROC) and areas under the precision-recall (AUPR). Results show that the Huber group LASSO outperforms the group LASSO and therefore demonstrate the effectiveness of our proposed method.

Briefly, the remainder of the paper is organized as follows. In Model Section, we introduced the model for the GRN, based on which the network topology is inferred. In Result Section, our proposed method is applied to the both simulation data and real experimental data. The results demonstrate the effectiveness of our method. Then, we conclude this study and point out the future work along this research in Conclusion

Section. Details of the method and its theoretical analysis can be found in Method Section.

4.2 Model

A model for GRN consisting of p genes is used in this study [109]:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{C}\mathbf{x} + \mathbf{S}\mathbf{r}, \\ \mathbf{r} &= \mathbf{f}(\mathbf{x}), \end{aligned} \tag{4.1}$$

where $\mathbf{x} = [x_1, \ldots, x_p]^T \in \mathcal{R}^p$ is the vector of mRNA concentrations; $\mathbf{C} = \text{diag}[-c_1, \ldots, -c_p] \in \mathcal{R}^{p\times p}$ is a diagonal matrix with $c_i > 0$ the degradation rate of gene i; the vector $\mathbf{r} = [r_1, \ldots, r_m]^T \in \mathcal{R}^m$ represents the reaction rates, which is a function of mRNA concentrations and $\mathbf{S} \in \mathcal{R}^{p\times m}$ is the stoichiometric matrix of the network. We assume that reaction rate $\mathbf{r}$ is a linear combination of mRNA concentrations,

$$\mathbf{r} = \mathbf{F}\mathbf{x}, \tag{4.2}$$

where $\mathbf{F} \in \mathcal{R}^{m\times p}$. Then, (4.1) becomes

$$\dot{\mathbf{x}} = \mathbf{C}\mathbf{x} + \mathbf{M}\mathbf{x}, \tag{4.3}$$

where $\mathbf{M} = \mathbf{S}\mathbf{F} \in \mathcal{R}^{p\times p}$. The elements of $\mathbf{M} = (m_{ij})_{1\le i,j\le p}$ indicate the network topology or regulatory relationships between genes. $m_{ij} \neq 0$ if gene j regulates the expression of gene i. Otherwise, $m_{ij} = 0$, gene j does not regulate gene i.

Since the gene expression levels are sampled at several time points, by using zero order hold discretization method, system (4.3) is discretized as

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} \tag{4.4}$$

where $\mathbf{A} = e^{\mathbf{C}\Delta t} + \mathbf{C}^{-1}(e^{\mathbf{C}\Delta t} - \mathbf{I})\mathbf{M}$. Note that $e^{\mathbf{C}\Delta t}$ and $\mathbf{C}^{-1}(e^{\mathbf{C}\Delta t} - \mathbf{I})$ are both diagonal matrices and their diagonal elements are all positive. Thus, the off-diagonal elements of $\mathbf{A} = (a_{ij})_{1\le i,j\le p}$ have the same zero and nonzero pattern as those of $\mathbf{M}$. In this study, we focus on inferring relationships between genes and do not consider self-regulations. As mentioned above, this can be achieved by identifying the nonzero off-diagonal elements in matrix $\mathbf{A}$, which can be interpreted as regulatory strengths.

Multiple time-course gene expression datasets for a GRN may be collected under different circumstances. One dataset is assumed to correspond to one inferred GRN topology, and all inferred GRNs should share the same network topology as their corresponding datasets are generated from the same underlying gene network. Our purpose is to reverse engineer the underlying network topology from these multiple datasets. More specifically, suppose we have m time-course gene expression datasets for a gene network: $\tilde{\mathbf{X}}(1), \ldots, \tilde{\mathbf{X}}(m)$, each of which is measured at $n_k + 1$ time points, i.e., $\tilde{\mathbf{X}}(k) \in \mathcal{R}^{p\times(n_k+1)}$. According to the model (4.4), these

datasets should satisfy

$$\mathbf{Y}(k) = \mathbf{A}(k)\mathbf{X}(k) + \mathbf{E}(k), \quad k = 1, \ldots, m, \tag{4.5}$$

where $\mathbf{Y}(k) = [\tilde{\mathbf{x}}_2(k), \ldots, \tilde{\mathbf{x}}_{n_k+1}(k)]$, the last n_k observations; $\mathbf{X}(k) = [\tilde{\mathbf{x}}_1(k), \ldots, \tilde{\mathbf{x}}_{n_k}(k)]$, the first n_k observations, $\mathbf{A}(k) \in \mathcal{R}^{p\times p}$, the regulatory matrix for the kth dataset and $\mathbf{E}(k)$, the errors or noises. All $\mathbf{A}(k)$'s are required to have the same structure. i.e., zero and nonzero pattern, but do not need to have the same value for every nonzero position because gene network is dynamic and regulatory strength may be different under different circumstances. In this study, we propose to use group LASSO penalty to implement this requirement and to use Huber loss function to take into account the robustness. Details of the proposed method are shown in the Method Section.

4.3 Results

To study the effectiveness of the proposed method, the Huber group LASSO is applied to inferring GRNs from both simulation datasets and real experimental datasets and the results of Huber group LASSO are compared with those from group LASSO in both area under receiver operating characteristic (AUROC) curve and area under the precision and recall (AUPR) curve.

4.3.1 Simulation example

A small-GRN consisting of 5 genes is considered in this example. The corresponding true network topology matrix is

$$\mathbf{A}_0 = \begin{bmatrix} + & - & + & 0 & 0 \\ - & + & 0 & 0 & + \\ 0 & + & + & 0 & 0 \\ + & - & 0 & + & 0 \\ 0 & 0 & 0 & + & + \end{bmatrix},$$

where $+$ and $-$ indicate the existence and regulation types of the edge. We randomly generate m stable regulatory matrices $\mathbf{A}(k)$, $k = 1, \ldots, m$, according to the template $\mathbf{A}_0$, such that $\text{sign}(\mathbf{A}(k)) = \text{sign}(\mathbf{A}_0)$. Then, m simulated time-course gene expression datasets, each with the number of time points, n_k, are generated from (4.5) with randomly chosen expression values at the first time point. The simulated error follows a mixed Gaussian distribution: with probability of 0.8, it has the distribution $N(0, 1)$ and with probability of 0.2, it has the distribution $N(0, 10^2)$. In this way, the simulated data contain large errors and outliers. To investigate the performances of our methods in different situations, we vary the values of m and n_k and apply the group LASSO and Huber group LASSO respectively to these generated datasets and compare the results from these two methods.

Data are generated under three situations ($m = 8, n_k = 15$), ($m = 4, n_k = 15$) and ($m = 4, n_k = 8$). Using the stability selection procedure that is introduced in the Method Section, network typologies consisting of

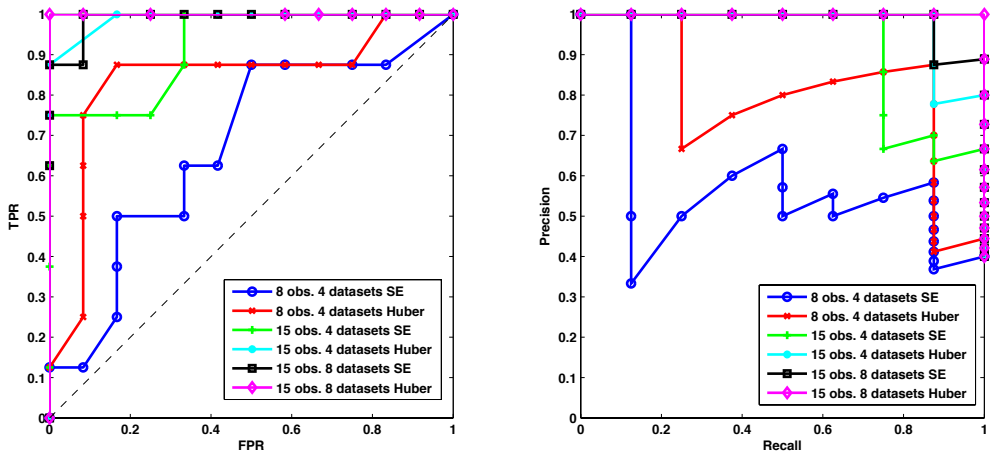


Figure 4.1: ROC plots and precision-recall plots of the Huber group LASSO and group LASSO for the simulation data under different situations. Left: the ROC plots of the Huber group LASSO and the group LASSO. Right: precision-recall plots of the Huber group LASSO and the group LASSO. TPR: true positive ratel. FPR: false positive rate. Huber group LASSO has better performance than group LASSO. The larger the number of observations or datasets, the better the performances of the methods.

Table 4.1: The areas under ROC (AUROC) and precision-recall (AUPR) of the Huber group LASSO and the group LASSO for the simulation datasets under different situations. SE: group LASSO. Huber: Huber group LASSO.

Situation	Method	AUROC	AUPR
15 observations 8 datasets	SE	0.9896	0.9852
	Huber	1.0000	1.0000
15 observations 4 datasets	SE	0.9219	0.9169
	Huber	0.9896	0.9736
8 observations 4 datasets	SE	0.6719	0.5749
	Huber	0.8385	0.8049

edges with scores or probabilities are inferred by Huber group LASSO and group LASSO. For the first two situations, we set the number of bootstrap samples as 30 and the moving block length as 10. For the third situation, we set the number of bootstrap samples as 30 and the moving block length as 5. Varying the threshold, the ROC plots and precision-recall plots of each method for different situations are obtained and are illustrated in Figure 4.1. The areas under the ROCs (AUROCs) and precision-recall curves (AUPRs) are calculated and reported in Table 4.1. From Figure 4.1 and Table 4.1, we can see that for each situation, the Huber group LASSO outperforms the group LASSO, i.e. the AUROC and AUPR of Huber group LASSO are larger than those of group LASSO. ROC plots in Figure 4.1 also show that both methods have better performances than the random guess. For the case of $m = 8$ and $n_k = 15$, the Huber group LASSO even achieves the maximum value of AUROC and AUPR. It can also be seen that for each method, the more the observations or the more the datasets, the larger AUROC and AUPR can be obtained. This is in accord with the intuition because, in this example, more observations or datasets indicate more information as these simulated data are generated under quite similar circumstances. All the simulation results have demonstrated

the effectiveness of our proposed method.

4.3.2 *In vivo* reverse engineering and modeling assessment (IRMA) data

The data used in this example come from the *In vivo* Reverse Engineering and Modeling Assessment (IRMA) experiment [137], where a network composed of five genes (GAL80, GAL4, CBF1, ASH1 and SWI5) was synthesized in yeast *Saccharomyces cerevisiae*, in which genes regulate each other through a variety of regulatory interactions. The network is negligibly affected by endogenous genes and it is responsive to small molecules. Galactose and glucose are respectively used to switch on and off the network. In this study, we use the IRMA time-course data consisting of four switch off datasets (with the number of time points varying from 19 to 21) and five switch on datasets (with the number of time points varying from 11 to 16).

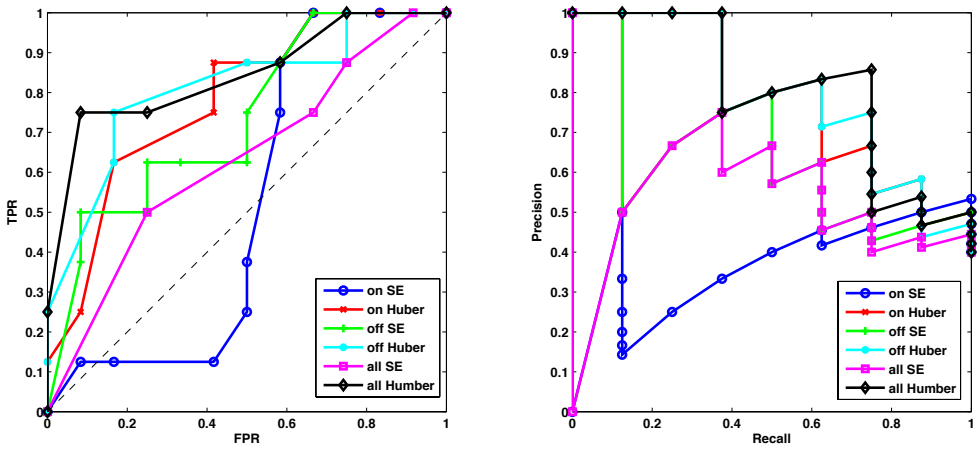


Figure 4.2: ROC plots and precision-recall plots of the Huber group LASSO and group LASSO for the IRMA datasets . Left: the ROC plots of the Huber group LASSO and the group LASSO. Right: precision-recall plots of the Huber group LASSO and the group LASSO. TPR: true positive ratel. FPR: false positive rate. Huber group LASSO has better performance than group LASSO.

Table 4.2: The areas under ROC (AUROC) and precision-recall (AUPR) of the Huber group LASSO and the group LASSO for the IRMA datasets. SE: group LASSO. Huber: Huber group LASSO.

Case	Method	AUROC	AUPR
Switch on datasets	SE	0.5208	0.3711
	Huber	0.7812	0.6971
Switch off datasets	SE	0.7344	0.6341
	Huber	0.8125	0.7928
All datasets	SE	0.6302	0.5122
	Huber	0.8438	0.8049

The Huber group LASSO and the group LASSO are applied to these data in three cases: (1) switch on datasets, (2) switch off datasets and (3) all datasets, i.e., combining switch on and switch off datasets. In the stability selection procedure, the number of bootstrap samples is 30 for all cases and the moving block length is 14 for the second case and 8 for the other cases. The ROC plots and precision-recall plots for the Huber group

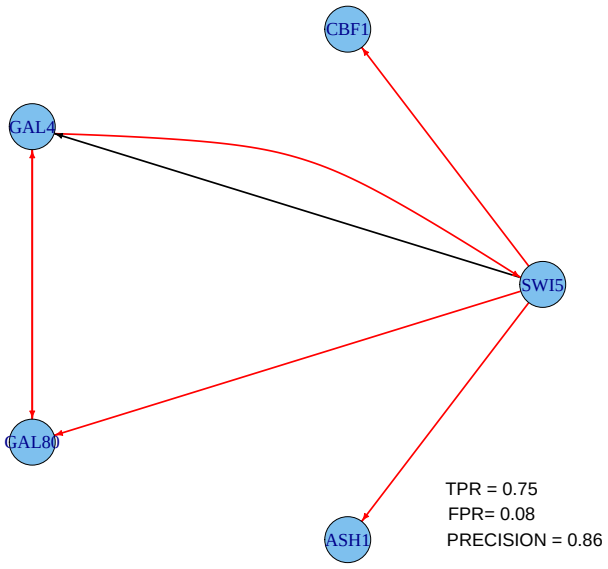


Figure 4.3: One network topology from Huber group LASSO using all IRMA datasets. TPR: true positive ratel. FPR: false positive rate.

LASSO and the group LASSO for each case are illustrated in Figure 4.2 and the corresponding AUROCs and AUPRs are summarized in Table 4.2. It can be seen that except the group LASSO for the switch on datasets, the performances of the methods are better than random guesses. The Huber group LASSO outperforms the group LASSO in both AUROCs and AUPRs. All methods for the switch off datasets perform better than for the switch on datasets. The group LASSO for all datasets has better performance than for the switch on datasets but is not as good as for the switch off datasets. The Huber group LASSO for all datasets has the best performance among all cases. This indicates that combining multiple datasets may lead to either the best result or a robust result which is better than the worst case. The network topology with false positive rate (FPR) 0.08 of the Huber group LASSO for all datasets is shown in Figure 4.3 and the corresponding true positive rate (TPR) is 0.75 with precision 0.86, in which the red edges represent true positives while black edges are false positives. The results show the effectiveness of our method for the IRMA data.

4.3.3 *E. coli* SOS network

In this example, we apply the proposed method to identify the real GRN, *E. coli* SOS DNA repair system as shown in Figure 4.4. This network is responsible for repairing the DNA after some damage happens. LexA acts as the master repressor of many genes in the normal states. When a damage occurs, RecA acts as a sensor and binds to single-stranded DNA to sense the damage and mediates the autocleavage of LexA. The repressions of the SOS genes are halted by the drop in LexA levels. The SOS genes are activated and start to repair the damages. When the repair is done, RecA level drops and stops mediating the autocleavage of LexA. Then, LexA accumulates and represses the SOS genes to make the cell go back to the normal state.

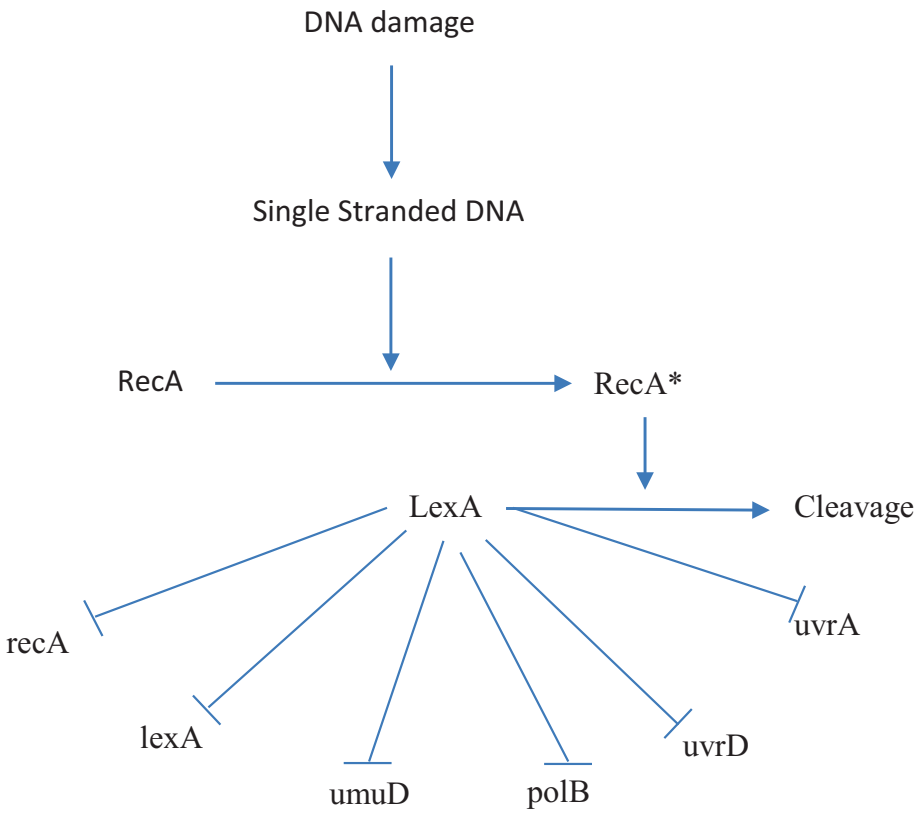


Figure 4.4: SOS DNA repair pathway in *E. coli*. The arrow represent activation while the flat arrow represents inhibition. Genes are in lower cases, proteins in capital letters.

Four time-course gene expression datasets of SOS DNA network are downloaded from the Uri Alon lab [1], which are produced from four experiments for various UV light intensities (Experiment 1 and 2: 5 Jm^{-2}, Experiment 3 and 4: 20 Jm^{-2}). Each dataset contain 8 genes and their measurements at 50 time points. As other literature did, e.g. [1, 121, 122, 123], only 6 genes, i.e., uvrD, lexA, umuD, recA, uvrA and polB are considered because they are well studied and the gold standard network of these genes are illustrated in Table 4.3. Details of the gold standard can be found in [1]. In this study, we do not consider the signs and the self-regulations.

As the conditions for the first two experiments are different for the last two experiments, we consider applying the method to three cases: (1) datasets of experiment 1 and 2, (2) datasets of experiment 3 and 4 and (3) all experiment datasets. In the stability selection procedure, the number of bootstrap samples is 30 and the moving block length is 25 for all cases. The ROC plots and precision-recall plots for the Huber group LASSO and the group LASSO for each case are illustrated in Figure 4.5 and the corresponding AUROCs and AUPRs are illustrated in Table 4.4. From the ROC plots and AUROCs, it can be seen that the Huber group LASSO performs significantly better than random guess while the group LASSO method is only a

[1] http://www.weizmann.ac.li/mcb/UriAlon.

Table 4.3: Gold standard of SOS network, collected from literature [1].

	uvrD	lexA	umuD	recA	uvrA	polB
uvrD	0	-1	-1	1	1	0
lexA	0	-1	-1	1	0	0
umuD	0	-1	-1	1	0	0
recA	0	-1	-1	1	0	0
uvrA	1	-1	-1	1	0	0
polB	0	-1	-1	1	0	0

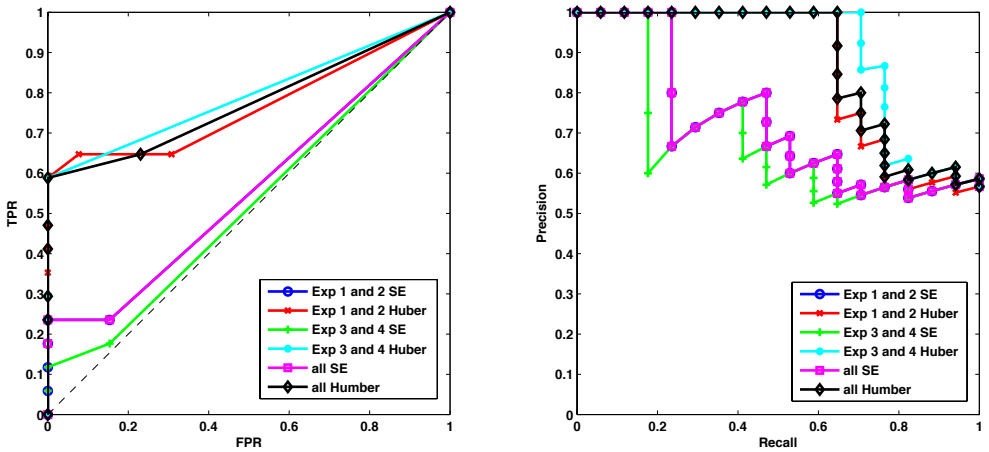


Figure 4.5: ROC plots and precision-recall plots of the Huber group LASSO and group LASSO for *E. coli* SOS datasets. Left: the ROC plots of the Huber group LASSO and the group LASSO. Right: precision-recall plots of the Huber group LASSO and the group LASSO. TPR: true positive ratel. FPR: false positive rate. Huber group LASSO has better performance than group LASSO.

little bit better than random guess. Obviously, the Huber group LASSO outperforms the group LASSO both in AUROCs and AUPR for all cases. The Huber group LASSO using experiment 3 and 4 datasets has the best performance. Performance of the Huber group LASSO using all datasets is between that in the first case and that in the second case. It can be considered as a robust result because of the using of multiple datasets. The network topology with FPR 0 and TPR 0.59 of the Huber group LASSO for all datasets is shown in Figure 4.6, in which all inferred edges are correct. These results demonstrate the effectiveness of our method for the *E. coli* SOS data.

4.3.4 *S. cerevisae cell cycle subnetwork*

A cell cycle regulatory subnetwork in S. cerevisae is inferred by the proposed method from 5 experimental microarray datasets. As in [124], the subnetwork consists of 27 genes including 10 genes for producing transcription factors (ace2, fkh1, swi4, swi5, mbp1, swi6, mcm1, fkh2, ndd1, yox1) and 17 genes for producing cyclin and cyclin/CDK regulatory proteins (cln1, cln2, cln3, cdc20, clb1, clb2, clb4, clb5, clb6, sic1, far1, spo12, apc1, tem1, gin4, swe1 and whi5). The time-course datasets we use include cell-cycle alpha factor release, cdc15, alpha factor fkh1 fkh2, fkh1,2 alpha factor and Elutriation, which are all downloaded from

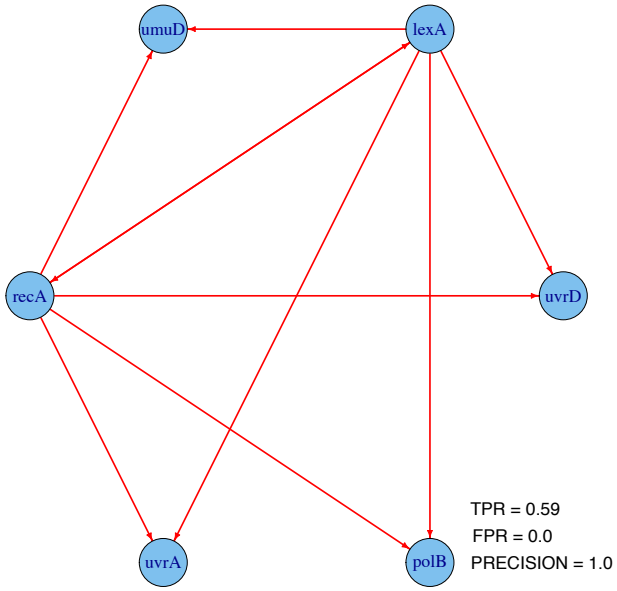


Figure 4.6: One network topology from Huber group LASSO using all *E. coli* SOS datasets. TPR: true positive ratel. FPR: false positive rate.

Table 4.4: The areas under ROC (AUROC) and precision-recall (AUPR) of the Huber group LASSO and the group LASSO for the *E. coli* SOS datasets. SE: group LASSO. Huber: Huber group LASSO.

Case	Method	AUROC	AUPR
Experiment 1 and 2	SE	0.5588	0.7225
	Huber	0.7670	0.8649
Experiment 3 and 4	SE	0.5204	0.6801
	Huber	0.7941	0.8981
All experiment data	SE	0.5588	0.7225
	Huber	0.7760	0.8756

Stanford Microarray Database (SMD). We apply the Huber group LASSO and the group LASSO respectively to infer the network from the datasets.

Table 4.5: The areas under ROC (AUROC) and precision-recall (AUPR) of the Huber group LASSO and the group LASSO for the cell cycle datasets. SE: group LASSO. Huber: Huber group LASSO.

Method	AUROC	AUPR
SE	0.5466	0.1844
Huber	0.5753	0.1941

In order to demonstrate the effectiveness of the proposed method, the inferred results are compared with the interaction network of the chosen 27 genes, drawn from BioGRID database [126]. The network in the database has 112 interactions, not including the self-regulations, and we take it as the gold standard regulatory network. In the stability selection procedure, the number of bootstrap samples is 30 and the moving block length is 9. The ROC plots and precision-recall plots are illustrated in Figure 4.7 and the AUROCs and AUPRs are shown in Table 4.5. We can see that both methods have better performances than

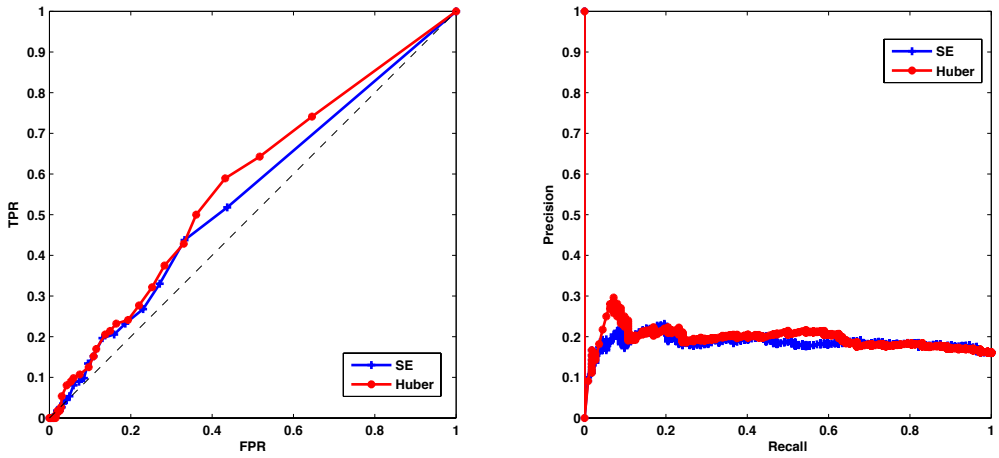


Figure 4.7: ROC plots and precision-recall plots of the Huber group LASSO and group LASSO for the cell cycle datasets. Left: the ROC plots of the Huber group LASSO and the group LASSO. Right: precision-recall plots of the Huber group LASSO and the group LASSO. TPR: true positive ratel. FPR: false positive rate. Huber group LASSO has better performance than group LASSO.

random guess and the Huber group LASSO outperforms the group LASSO. One network from Huber group LASSO with FPR 0.43 and TPR 0.59 is shown in Figure 4.8, in which red edges are those inferred edges having been identified in the database and grey edges might be either false positives or novel discovered regulatory relations. Although the gold standard network extracted from the database may contain false edges or not be complete, this shows the effectiveness of our method to some extent.

4.4 Conclusions

A novel method, Huber group LASSO, has been proposed to integrate multiple time-course gene expression datasets to infer the underlying GRN topology. As an extension to the group LASSO, it is robust to large noises and outliers. An efficient algorithm which combines the ideas of auxiliary function minimization and block descent is developed to slove the involved optimization problem. The convergence analysis of the algorithm shows that the sequence generated from the algorithm indeed converges to the optimal solution of the problem. Instead of selecting a specific tuning parameter corresponding to a determinant network topology, an adapted stability selection procedure is used to lead to a network consisting of edges with scores. The applications of the proposed method to the simulation datasets and real experimental datasets show that Huber group LASSO outperforms the group LASSO in both AUROC and AUPR. It also shows that the integration of multiple time-course gene expression datasets by the proposed method lead to reliable inferred network typologies.

The information in the gene expression data is quite limited. One direction of the future work along this study is to extend the method to be able to integrate other types of data with the gene expression data.

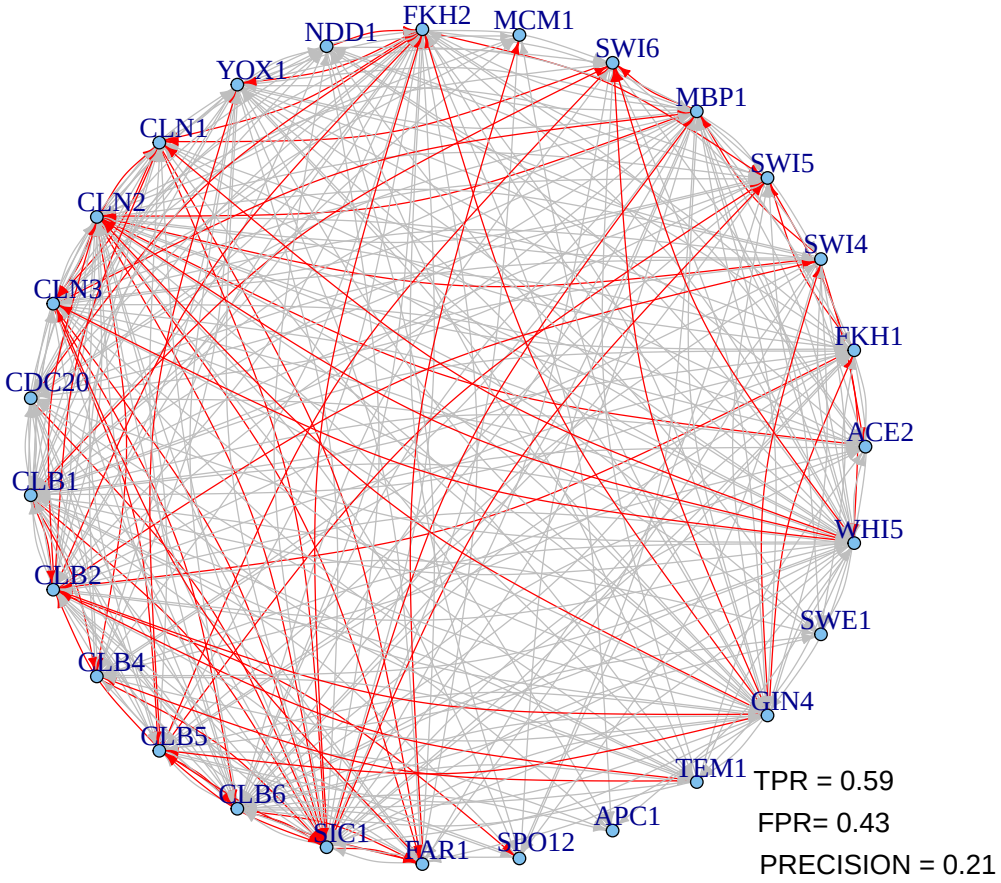


Figure 4.8: One network topology from Huber group LASSO for cell cycle datasets. TPR: true positive ratel. FPR: false positive rate.

4.5 Method

4.5.1 Huber group LASSO

Given m datasets, $\tilde{\mathbf{X}}(1), \ldots, \tilde{\mathbf{X}}(m)$, satisfying model (4.5), to ensure that all $\mathbf{A}(k)$'s have the same structure, elements of $\mathbf{A}(k)$'s on the same position are grouped together and can be inferred by the group LASSO,

$$\min_{\mathbf{A}(k)} \sum_{i=1}^{p} \sum_{k=1}^{m} w_k \sum_{j=1}^{n_k} (y_{ij}(k) - \mathbf{A}_i(k)^T \mathbf{x}_j(k))^2 + \lambda \sum_{i=1}^{p} \sum_{\ell=1}^{p} \sqrt{a_{i\ell}(1)^2 + \ldots + a_{i\ell}(m)^2}, \tag{4.6}$$

where $\mathbf{A}_i(k)^T$ is the ith row of the matrix $\mathbf{A}(k)$ and $\mathbf{x}_j(k)$ is the jth column of the matrix $\mathbf{X}(k)$. w_k is the weight for the kth dataset, which can be assigned by experience. In this study, we choose $w_k = n_k / \sum n_k$ i.e., the more observations the dataset has, the higher weight it is assigned with. The penalty term in (4.6) takes advantage of the sparse nature of GRNs and has the effect making the grouped parameters to be estimated either all zeros or all non-zeros [133], i.e., $a_{i\ell}(k)$'s, $k = 1, \ldots, m$, become either all zeros or all non-zeros.

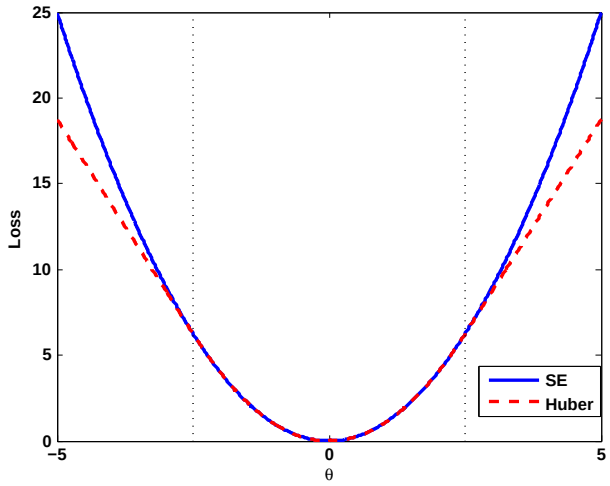


Figure 4.9: Squared error and Huber loss functions. For small error, θ, squared error loss and Huber loss are the same. For large error, squared error penalizes quadratically while Huber loss penalizes linearly.

Therefore, a consistent network topology can be obtained from the group LASSO method. λ is a tuning parameter which controls the degree of sparseness of the inferred network. The larger the value of λ, the more grouped parameters become zeros.

To introduce robustness, we consider using the Huber loss function instead of the squared error loss function and propose the following Huber group LASSO method

$$\min_{\mathbf{A}(k)} \sum_{i=1}^{p} \sum_{k=1}^{m} w_k \sum_{j=1}^{n_k} H_\delta(y_{ij}(k) - \mathbf{A}_i(k)^T \mathbf{x}_j(k)) + \lambda \sum_{i=1}^{p} \sum_{\ell=1}^{p} \sqrt{a_{i\ell}(1)^2 + \ldots + a_{i\ell}(m)^2}, \tag{4.7}$$

where the Huber loss function is defined as

$$H_\delta(\theta) = \begin{cases} \theta^2 & \text{if } |\theta| \leq \delta \\ 2\delta|\theta| - \delta^2 & \text{otherwise.} \end{cases} \tag{4.8}$$

The squared error and Huber loss function are illustrated in Figure 4.9. It can be seen that for small errors, these two loss functions are exactly the same while for large errors, Huber loss which increases linearly is less than the squared error loss which increases quadratically. Because Huber loss penalizes much less than the squared error loss for large errors, the Huber group LASSO is more robust than group LASSO when there exists large noise or outliers in the data. It is also known that the Huber loss is nearly as efficient as squared error loss for Gaussian errors [138].

For convenience, we define some notations and rewrite the problems (4.6) and (4.7) in more compact forms. Let $\mathbf{Y}_i = [\mathbf{Y}_i(1)^T, \ldots, \mathbf{Y}_i(m)^T]^T$, the vector stacking observations of the ith target gene across all datasets, where $\mathbf{Y}_i(k)^T$ is the ith row of $\mathbf{Y}(k)$. Let $\mathbf{b}_{i\ell} = [a_{i\ell}(1), \ldots, a_{i\ell}(m)]^T$, the vector containing the grouped parameters. Denote by $\mathbf{b}_i = [\mathbf{b}_{i1}^T, \ldots, \mathbf{b}_{ip}^T]^T$ the vector containing all parameters related to the

regulation of the ith target gene. According to the orders of the parameters in $\mathbf{b}_i$, re-arrange the rows of $\mathbf{X}(k)$ and piece them together to have $\mathbf{X} = [\mathbf{X}_1^T, \ldots, \mathbf{X}_p^T]^T$, where $\mathbf{X}_i = \text{diag}(\mathbf{X}_i(1)^T, \ldots, \mathbf{X}_i(m)^T)$ with $\mathbf{X}_i(k)^T$ being the ith row of $\mathbf{X}(k)$. Then (4.7) can be rewritten as

$$\sum_{i=1}^{p}\sum_{j=1}^{n} \omega_j H_\delta(y_{ij} - \mathbf{x}_j^T \mathbf{b}_i) + \lambda \sum_{i=1}^{p}\sum_{\ell=1}^{p} \|\mathbf{b}_{i\ell}\|_2, \tag{4.9}$$

where $\mathbf{x}_j$ is the jth column of $\mathbf{X}$, y_{ij} is the jth element of $\mathbf{Y}_i$, $n = \sum_{k=1}^{m} n_k$ and $\omega_i = w_1 I(i \leq n_1) + \sum_{k=2}^{m} w_k I(\sum_{l=1}^{k-1} n_l < i \leq \sum_{l=1}^{k} n_l)$. (4.6) can be rewritten similarly.

4.5.2 Optimization algorithm

The minimization of problem (4.9) is not easy as the penalty term is not differentiable at zero and the Huber loss does not have the second order derivatives at the transition points, $\pm\delta$. Observed that fixing i, the problem (4.9) can be decomposed into p sub-optimization problems. For each, we get $\mathbf{b}_i$ by minimizing

$$J(\mathbf{b}) = \sum_{j=1}^{n} \omega_j H_\delta(y_j - \mathbf{x}_j^T \mathbf{b}) + \lambda \sum_{\ell=1}^{p} \|\mathbf{b}_\ell\|_2, \tag{4.10}$$

where for notational convenience, we omit the subscript i here and $\mathbf{b}_\ell$ is a block of parameters of $\mathbf{b}$, i.e. $\mathbf{b} = [\mathbf{b}_1^T, \ldots, \mathbf{b}_p^T]^T$.

To optimize (4.10), an iterative method is developed by constructing an auxiliary function, the optimization of which keeps $J(\mathbf{b})$ decreasing. As in [135], given any current estimate $\mathbf{b}^{(k)}$, a function $Q(\mathbf{b}|\mathbf{b}^{(k)})$ is an auxiliary function for $J(\mathbf{b})$ if conditions

$$J(\mathbf{b}^{(k)}) = Q(\mathbf{b}^{(k)}|\mathbf{b}^{(k)}) \quad \text{and} \quad J(\mathbf{b}) \leq Q(\mathbf{b}|\mathbf{b}^{(k)}) \quad \text{for all } \mathbf{b}, \tag{4.11}$$

are satisfied. In this study, we construct the auxiliary function as

$$Q(\mathbf{b}|\mathbf{b}^{(k)}) = \sum_{j=1}^{n} \omega_j H_\delta(y_j - \mathbf{x}_j^T \mathbf{b}^{(k)}) - \sum_{j=1}^{n} \omega_j H'_\delta(y_j - \mathbf{x}_j^T \mathbf{b}^{(k)}) \mathbf{x}_j^T (\mathbf{b} - \mathbf{b}^{(k)}) + 2\gamma \|\mathbf{b} - \mathbf{b}^{(k)}\|_2^2 + \lambda \sum_{\ell=1}^{p} \|\mathbf{b}_\ell\|_2, \tag{4.12}$$

where γ is the largest eigenvalue of $\sum_{j=1}^{n} \omega_j \mathbf{x}_j \mathbf{x}_j^T$. It can be easily shown that this auxiliary function satisfies (4.11).

Considering the block structure of $\mathbf{b}$, we apply a block-wise descent strategy [136], i.e., cyclically optimize one block of parameters, $\mathbf{b}_j$, at a time. Denote by $\mathbf{b}^{(k)}(\ell) = [{\mathbf{b}_1^{(k+1)}}^T, \ldots, {\mathbf{b}_\ell^{(k+1)}}^T, {\mathbf{b}_{\ell+1}^{(k)}}^T, \ldots, {\mathbf{b}_p^{(k)}}^T]^T$ the vector after updating the ℓth block. Given $\mathbf{b}^{(k)}(\ell - 1)$, update it to $\mathbf{b}^{(k)}(\ell)$ by computing

$$\begin{aligned}\mathbf{b}_\ell^{(k+1)} &= \arg\min_{\mathbf{b}_\ell} Q\left([{\mathbf{b}_1^{(k+1)}}^T, \ldots, {\mathbf{b}_{\ell-1}^{(k+1)}}^T, \mathbf{b}_\ell^T, {\mathbf{b}_{\ell+1}^{(k)}}^T, \ldots, {\mathbf{b}_p^{(k)}}^T]^T | \mathbf{b}^{(k)}(\ell-1)\right) \\ &= \left(\frac{1}{4\gamma} - \frac{\lambda}{4\gamma \| \sum_{j=1}^n \omega_j H'_\delta(y_j - \mathbf{x}_j^T \mathbf{b}^{(k)}(\ell-1))\mathbf{x}_{j,\ell} + 4\gamma \mathbf{b}_\ell^{(k)} \|_2}\right)_+ \\ &\times \left(\sum_{j=1}^n \omega_j H'_\delta(y_j - \mathbf{x}_j^T \mathbf{b}^{(k)}(\ell-1))\mathbf{x}_{j,\ell} + 4\gamma \mathbf{b}_\ell^{(k)}\right).\end{aligned} \tag{4.13}$$

where $\mathbf{x}_{j,\ell}$ is the block of elements in $\mathbf{x}_j$ corresponding to $\mathbf{b}_\ell$ and $(\cdot)_+ = \max(\cdot, 0)$. We repeat to update every block using (4.13) until it converges. For a specific value of λ, the whole procedure is described as follows:

1. Initialize $\mathbf{b}^{(0)}$. Set iteration number $k = 0$.
2. Cycle through (4.13) one at a time to update the ℓth block, $\ell = 1, \ldots, p$
3. If $\{\mathbf{b}^{(k)}\}$ converges to $\mathbf{b}^*$, go to the next step. Otherwise, set $k := k + 1$ and go to Step 2.
4. Return the solution $\mathbf{b}^*$.

Note that the algorithm can be adapted to solve (4.6) with quite similar derivations. In the following section, we show that the sequence $\{\mathbf{b}^{(k)}\}$ generated from the algorithm guarantees the objective function $J(\mathbf{b})$ keep decreasing. We also show that the limit point of the sequence generated is indeed the minimum point of $J(\mathbf{b})$.

4.5.3 Convergence analysis

The convergence of the optimization algorithm for the minimization of (4.10) is analyzed in the way similar to [139]. We first show the descent property of the algorithm.

Lemma 1. *The sequence $\{\mathbf{b}^{(k)}\}$ generated from the optimization algorithm keeps the objective function $J(\mathbf{b})$ decreasing, i.e., $J(\mathbf{b}^{(k)}) \geq J(\mathbf{b}^{(k+1)})$.*

Proof. By (4.11) and (4.13), we have

$$\begin{aligned}J(\mathbf{b}^{(k)}) &= Q(\mathbf{b}^{(k)}|\mathbf{b}^{(k)}) \geq Q(\mathbf{b}^{(k)}(1)|\mathbf{b}^{(k)}) \\ &\geq Q(\mathbf{b}^{(k)}(2)|\mathbf{b}^{(k)}(1)) \geq \cdots \geq Q(\mathbf{b}^{(k)}(p)|\mathbf{b}^{(k)}(p-1)) \geq J(\mathbf{b}^{(k)}(p)) \\ &= J(\mathbf{b}^{(k+1)}).\end{aligned}$$

□

Next, we show that if the generated sequence satisfies some conditions, it converges to the optimal solution.

Lemma 2. *Assume the data $(\mathbf{y}, \mathbf{X})$ lies on a compact set and the following conditions are satisfied:*

1. *The sequence* $\{\mathbf{b}^{(k)}\}$ *is bounded.*

2. *For every convergent subsequence* $\{\mathbf{b}^{(n_k)}\} \subset \{\mathbf{b}^{(k)}\}$*, the successive differences converge to zeros,* $\mathbf{b}^{(n_k)} - \mathbf{b}^{(n_k-1)} \to \mathbf{0}$.

Then, every limit point $\mathbf{b}^\infty$ *of the sequence* $\{\mathbf{b}^{(k)}\}$ *is a minimum for the function* $J(\mathbf{b})$*, i.e., for any* $\boldsymbol{\delta} = (\boldsymbol{\delta}_1^T, \ldots, \boldsymbol{\delta}_p^T)^T \in \mathbb{R}^{mp}$*,*

Proof. For any $\mathbf{b} = (\mathbf{b}_1^T, \ldots, \mathbf{b}_p^T)^T \in \mathbb{R}^{mp}$ and $\boldsymbol{\delta}(j) = (\mathbf{0}^T, \ldots, \boldsymbol{\delta}_j^T, \ldots, \mathbf{0}^T)^T \in \mathbb{R}^{mp}$

$$\liminf_{\alpha \downarrow 0+} \left\{ \frac{J(\mathbf{b} + \alpha\boldsymbol{\delta}(j)) - J(\mathbf{b})}{\alpha} \right\} = \nabla_j f(\mathbf{b})^T \boldsymbol{\delta}_j + \liminf_{\alpha \downarrow 0+} \left\{ \frac{\lambda(\|\mathbf{b}_j + \alpha\boldsymbol{\delta}_j\|_2 - \|\mathbf{b}_j\|_2)}{\alpha} \right\},$$

where $f(\mathbf{b}) = \sum_{i=1}^n \omega_i H_\delta(y_i - \mathbf{x}_i^T\mathbf{b})$ and ∇_j represents the partial derivatives with respect to the jth block of parameters. Denote the second term by $\partial P(\mathbf{b}_j; \boldsymbol{\delta}_j)$ and it has

$$\partial P(\mathbf{b}_j; \boldsymbol{\delta}_j) = \begin{cases} \lambda \frac{\mathbf{b}_j^T \boldsymbol{\delta}_j}{\|\mathbf{b}_j\|_2} & \text{if } \mathbf{b}_j \neq \mathbf{0}, \\ \lambda \|\boldsymbol{\delta}_j\|_2 & \text{otherwise.} \end{cases} \tag{4.14}$$

We assume the subsequence $\{\mathbf{b}^{(n_k)}\}$ converges to $\mathbf{b}^\infty = (\mathbf{b}_1^{\infty T}, \ldots, \mathbf{b}_p^{\infty T})^T \in \mathbb{R}^{mp}$. From condition 2. and (4.14), we have

$$\mathbf{b}^{(n_k-1)}(j) = ({\mathbf{b}_1^{(n_k)}}^T, \ldots, {\mathbf{b}_j^{(n_k)}}^T, {\mathbf{b}_{j+1}^{(n_k-1)}}^T, \ldots, {\mathbf{b}_p^{(n_k-1)}}^T)^T \to \mathbf{b}^\infty, \quad \text{as } k \to \infty,$$

and

$$\text{if } \mathbf{b}_j^\infty \neq \mathbf{0}, \quad \partial P(\mathbf{b}_j^{(n_k)}; \boldsymbol{\delta}_j) \to \partial P(\mathbf{b}_j^\infty; \boldsymbol{\delta}_j); \quad \text{if } \mathbf{b}_\mathbf{j}^\infty = \mathbf{0}, \quad \partial P(\mathbf{b}_j^\infty; \boldsymbol{\delta}_j) \geq \liminf_{k \to \infty} \partial P(\mathbf{b}_j^{(n_k)}; \boldsymbol{\delta}_j), \tag{4.15}$$

since $\mathbf{b}_j^T \boldsymbol{\delta}_j \leq \|\mathbf{b}_j\|_2 \|\boldsymbol{\delta}_j\|_2$.

As $\mathbf{b}_j^{(n_k)}$ minimizes $Q(({\mathbf{b}_1^{(n_k)}}^T, \ldots, {\mathbf{b}_{j-1}^{(n_k)}}^T, \mathbf{b}_j^T, {\mathbf{b}_{j+1}^{(n_k-1)}}^T, \ldots, {\mathbf{b}_p^{(n_k-1)}}^T)^T | \mathbf{b}^{(n_k)}(j-1))$ with respect to the jth block of parameters, using (4.14), we have

$$\nabla_j q(\mathbf{b}^{(n_k)}(j) | \mathbf{b}^{(n_k)}(j-1))^T \boldsymbol{\delta}_j + \partial P(\mathbf{b}_j^{(n_k)}; \boldsymbol{\delta}_j) \geq 0, \quad \text{for all } k, \tag{4.16}$$

with

$$\begin{aligned} q(\mathbf{b}^{(n_k)}(j) | \mathbf{b}^{(n_k)}(j-1)) = & \sum_{i=1}^n \omega_i H_\delta(y_i - \mathbf{x}_i^T \mathbf{b}^{(n_k)}(j-1)) \\ & - \sum_{i=1}^n \omega_i H'_\delta(y_i - \mathbf{x}_i^T \mathbf{b}^{(n_k)}(j-1)) \mathbf{x}_i^T (\mathbf{b}^{(n_k)}(j) - \mathbf{b}^{(n_k)}(j-1)) \\ & + 2\gamma \|\mathbf{b}^{(n_k)}(j) - \mathbf{b}^{(n_k)}(j-1)\|_2^2 \end{aligned}$$

Due to condition 2.,

$$\nabla_j q(\mathbf{b}^{(n_k)}(j)|\mathbf{b}^{(n_k)}(j-1)) \to \nabla_j f(\mathbf{b}^\infty) \quad \text{as } k \to \infty. \tag{4.17}$$

Therefore, (4.15), (4.16) and (4.17) yield

$$\nabla_j f(\mathbf{b}^\infty)^T \boldsymbol{\delta}_j + \partial P(\mathbf{b}^\infty; \boldsymbol{\delta}_j) \geq \liminf_{k\to\infty} \left\{ \nabla_j q(\mathbf{b}^{(n_k)}(j)|\mathbf{b}^{(n_k)}(j-1))^T \boldsymbol{\delta}_j + \partial P(\mathbf{b}_j^{(n_k)}; \boldsymbol{\delta}_j) \right\} \geq 0, \tag{4.18}$$

for any $1 \leq j \leq p$.

For $\boldsymbol{\delta} = (\boldsymbol{\delta}_1^T, \ldots, \boldsymbol{\delta}_p^T)^T \in \mathbb{R}^{mp}$, due to the differentiability of $f(\mathbf{b})$,

$$\begin{aligned}
\liminf_{\alpha\downarrow 0+} \left\{ \frac{J(\mathbf{b}^\infty + \alpha\boldsymbol{\delta}) - J(\mathbf{b}^\infty)}{\alpha} \right\} &= \sum_{j=1}^p \nabla_j f(\mathbf{b}^\infty)^T \boldsymbol{\delta}_j + \sum_{j=1}^p \liminf_{\alpha\downarrow 0+} \left\{ \frac{\lambda(\|\mathbf{b}_j^\infty + \alpha\boldsymbol{\delta}_j\|_2 - \|\mathbf{b}_j^\infty\|_2)}{\alpha} \right\} \\
&= \sum_{j=1}^p \left\{ \nabla_j f(\mathbf{b}^\infty)^T \boldsymbol{\delta}_j + \partial P(\mathbf{b}^\infty; \boldsymbol{\delta}_j) \right\} \geq 0.
\end{aligned}$$

□

Finally, we show that the sequence generated from the proposed algorithm satisfies these two conditions.

Theorem 4.1. *Assuming the data* $(\mathbf{y}, \mathbf{X})$ *lies on a compact set and no column of* $\mathbf{X}$ *is identically* $\mathbf{0}$*, the sequence* $\{\mathbf{b}^{(k)}\}$ *generated from the algorithm converges to the minimum point of the objective function* $J(\mathbf{b})$*.*

Proof. We only need to show that the generated sequence meets the conditions in Lemma 2.

For the sake of notational convenience, for fixed j and $(\mathbf{b}_1^T, \ldots, \mathbf{b}_{j-1}^T, \mathbf{b}_{j+1}^T, \ldots, \mathbf{b}_p^T)^T$, define

$$\chi(\cdot) : \mathbf{u} \mapsto J((\mathbf{b}_1^T, \ldots, \mathbf{b}_{j-1}^T, \mathbf{u}^T, \mathbf{b}_{j+1}^T, \ldots, \mathbf{b}_p^T)^T).$$

Let $\mathbf{b}(\mathbf{u})$ be the vector containing $\mathbf{u}$ as its jth block of parameters with other blocks being the fixed values.

Assume $\mathbf{u}+\boldsymbol{\delta}$ and $\mathbf{u}$ represent the values of the jth block of parameters before and after the block update, respectively. Hence, as defined in (4.12), $\mathbf{u}$ is obtained by minimizing the following function with respect to the jth block in the algorithm:

$$\begin{aligned}
&Q(\mathbf{b}(\mathbf{u})|\mathbf{b}(\mathbf{u}+\boldsymbol{\delta})) \\
=&f(\mathbf{b}(\mathbf{u}+\boldsymbol{\delta})) + \nabla_j f(\mathbf{b}(\mathbf{u}+\boldsymbol{\delta}))^T(\mathbf{u} - (\mathbf{u}+\boldsymbol{\delta})) \\
&+ 2\gamma\|\mathbf{u} - (\mathbf{u}+\boldsymbol{\delta})\|_2^2 + \lambda\|\mathbf{u}\|_2 + \lambda\sum_{\ell\neq j}\|\mathbf{b}_\ell\|_2,
\end{aligned} \tag{4.19}$$

where $f(\mathbf{b}) = \sum_{i=1}^n \omega_i H_\delta(y_i - \mathbf{x}_i^T\mathbf{b})$ and $\nabla_j f(\mathbf{b}) = -\sum_{i=1}^n \omega_i H'_\delta(y_i - \mathbf{x}_i^T\mathbf{b})\mathbf{x}_{i,j}$. Thus, $\mathbf{u}$ should satisfy

$$\nabla_j f(\mathbf{b}(\mathbf{u}+\boldsymbol{\delta})) - 4\gamma\boldsymbol{\delta} + \lambda\mathbf{s} = 0, \tag{4.20}$$

where $\mathbf{s} = \mathbf{u}/\|\mathbf{u}\|_2$ if $\mathbf{u} \neq 0$; $\|\mathbf{s}\|_2 \leq 1$ if $\mathbf{u} = 0$. Then, we have

$$\begin{aligned}
&\chi(\mathbf{u}+\boldsymbol{\delta}) - \chi(\mathbf{u}) \\
=&f(\mathbf{b}(\mathbf{u}+\boldsymbol{\delta})) - f(\mathbf{b}(\mathbf{u})) + \lambda(\|\mathbf{u}+\boldsymbol{\delta}\|_2 - \|\mathbf{u}\|_2) \\
=&\nabla_j f(\mathbf{b}(\mathbf{u}+\tau\boldsymbol{\delta}))^T\boldsymbol{\delta} - \nabla_j f(\mathbf{b}(\mathbf{u}+\boldsymbol{\delta}))^T\boldsymbol{\delta} + [\nabla_j f(\mathbf{b}(\mathbf{u}+\boldsymbol{\delta}))^T\boldsymbol{\delta} - 4\gamma\boldsymbol{\delta}^T\boldsymbol{\delta} + \lambda\mathbf{s}^T\boldsymbol{\delta}] \\
&+ 4\gamma\boldsymbol{\delta}^T\boldsymbol{\delta} + \lambda(\|\mathbf{u}+\boldsymbol{\delta}\|_2 - \|\mathbf{u}\|_2 - \mathbf{s}^T\boldsymbol{\delta}) \\
=&-\left[\sum_{i=1}^n \omega_i H'_\delta(y_i - \mathbf{x}_i^T\mathbf{b}(\mathbf{u}+\tau\boldsymbol{\delta}))\mathbf{x}_{i,j}^T\boldsymbol{\delta} - \sum_{i=1}^n \omega_i H'_\delta(y_i - \mathbf{x}_i^T\mathbf{b}(\mathbf{u}+\boldsymbol{\delta}))\mathbf{x}_{i,j}^T\boldsymbol{\delta}\right] \\
&+ 4\gamma\boldsymbol{\delta}^T\boldsymbol{\delta} + \lambda(\|\mathbf{u}+\boldsymbol{\delta}\|_2 - \|\mathbf{u}\|_2 - \mathbf{s}^T\boldsymbol{\delta}) \\
\geq&-2\gamma(1-\tau)\|\boldsymbol{\delta}\|_2^2 + 4\gamma\|\boldsymbol{\delta}\|_2^2 \geq 2\gamma\|\boldsymbol{\delta}\|_2^2.
\end{aligned} \tag{4.21}$$

The second and third equalities are obtained using mean value theorem with $\tau \in (0,1)$ and (4.20). For the first inequality, the following property of the Huber loss function and the property of subgradient are used.

$$(H'_\delta(\theta_1) - H'_\delta(\theta_2))(\theta_1 - \theta_2) \leq 2(\theta_1 - \theta_2)^2.$$

The result from (4.21) gives that

$$J(\mathbf{b}^{(k)}(j-1)) - J(\mathbf{b}^{(k)}(j)) \geq 2\gamma\|\mathbf{b}_j^{(k)} - \mathbf{b}_j^{(k+1)}\|_2^2 = 2\gamma\|\mathbf{b}^{(k)}(j-1) - \mathbf{b}^{(k)}(j)\|_2^2, \tag{4.22}$$

where $\mathbf{b}^{(k)}(j) = [{\mathbf{b}_1^{(k+1)}}^T, \ldots, {\mathbf{b}_j^{(k+1)}}^T, {\mathbf{b}_{j+1}^{(k)}}^T, \ldots, {\mathbf{b}_p^{(k)}}^T]^T$.

Using (4.22) repeatedly across every block, for any k, we have

$$J(\mathbf{b}^{(k)}) - J(\mathbf{b}^{(k+1)}) \geq 2\gamma\|\mathbf{b}^{(k)} - \mathbf{b}^{(k+1)}\|_2^2. \tag{4.23}$$

Note that by Lemma 1, $\{J(\mathbf{b}^{(k)})\}$ converges as it keeps decreasing and is bounded from below. The convergence of $\{J(\mathbf{b}^{(k)})\}$ yield the convergence of $\{\mathbf{b}^{(k)}\}$. Hence, conditions of Lemma 2 hold which imply that the limit of $\{\mathbf{b}^{(k)}\}$ is the minimum point of $J(\mathbf{b})$. □

4.5.4 Implementation

The tuning parameter λ controls the sparseness of the resulted network. A network solution path can be obtained by computing networks on a grid of λ values from $\lambda_{\max} = \max_{i,\ell} \left\|\sum_{j=1}^n \omega_j H'_\delta(y_{ij})\mathbf{x}_{j,\ell}\right\|_2$, which is the smallest value that gives the empty network, to a small value, e.g. $\lambda_{\min} = 0.01\lambda_{\max}$. In our previous work [128], BIC criterion is used to pick a specific λ value which corresponds to a determinant network topology. A method called "stability selection" recently proposed by Meinshausen and Buhlmann [115] finds a network with probabilities for edges. Stability selection performs the network inference method, e.g. group LASSO, many times, resampling the data in each run and computing the frequencies with which each edge is selected

across these runs. It has been used with the linear regression method to infer GRNs from steady-state gene expression data in Haury et al. [100] and has shown prospective effectiveness. In this study, we adapt the stability selection method to finding GRN topology from multiple time-course gene expression datasets. Given a family of m time-course gene expression datasets $\tilde{\mathbf{X}}(k) \in \mathcal{R}^{p\times n_k}$, $k = 1, \ldots, m$, for a specific $\lambda \in \Lambda$, the stability selection procedure is as follows

1. Use moving block bootstrap to draw N bootstrap samples for every dataset to form N bootstrap families of multiple time-course datasets, i.e. $\{\tilde{\mathbf{X}}(k)^{*(b)}\}_{k=1}^{m}$, $b = 1, \ldots, N$.

2. Use the proposed Huber group LASSO to infer $\{\mathbf{A}(k)_\lambda^{*(b)}\}_{k=1}^{m}$ from the bth bootstrap family of datasets. Denote by $\mathbf{A}_\lambda^{*(b)}$ the network topology shared by $\{\mathbf{A}(k)_\lambda^{*(b)}\}_{k=1}^{m}$.

3. Compute the frequencies for each edge (i, j), i.e., from the gene j to gene i, in the network

$$\Pi_\lambda(i,j) = \frac{\sharp\{b : \mathbf{A}_\lambda^{*(b)}(i,j) \neq 0\}}{N}, \tag{4.24}$$

   where $\mathbf{A}_\lambda^{*(b)}(i,j)$ is the (i,j)'s entry of $\mathbf{A}_\lambda^{*(b)}$ and $\sharp\{\cdot\}$ is the number of elements in that set.

For a set of $\lambda \in \Lambda$, the probability of each edge in the inferred network is

$$\Pi(i,j) = \max_{\lambda\in\Lambda} \Pi_\lambda(i,j) \tag{4.25}$$

The final network topology can be obtained by setting a threshold, edges with probabilities or scores less than which are considered nonexistent. This study only focus on giving a list of edges with scores. The selection of threshold is not discussed here. The stability selection procedure can also be applied with the group LASSO method (4.6).

Since the data used are time series data, the moving block bootstrap method is employed in the first step to draw bootstrap samples from each dataset. For a dataset with n observations, in the moving block bootstrap with block length l, the data is split into $n - l + 1$ blocks: block j consists of observations j to $j + l - 1$, $j = 1, \ldots, n - l + 1$. $\lceil n/b \rceil$ blocks are randomly drawn from $n - l + 1$ blocks with replacement and are aligned in the order they are picked to form a bootstrap sample.

Another tuning parameter δ controls the degree of robustness. Generally, it picks $\delta = 1.345\hat{\sigma}$ where $\hat{\sigma}$ is the estimated standard deviation of the error and $\hat{\sigma} = MAD/0.6745$, where MAD is the median absolute deviation of the residuals. In this study, we use the least absolute deviations (LAD) regression to obtain the residuals. To avoid the overfitting of LAD which leads to a very small δ, we choose by $\delta = \max(1.345\hat{\sigma}, 1)$.

Declarations

Publication of this article was funded by Natural Science and Engineering Research Council of Canada (NSERC).

Chapter 5

Alternating Weighted Least Squares Parameter Estimation for Biological S-Systems

Published as: L.Z. Liu, F.X. Wu, and W.J. Zhang, "Alternating weighted least squares parameter estimation for biological s-systems," in *Systems Biology (ISB), 2012 IEEE 6th International Conference on*, pp. 6–11, 2012 [140].

In Chapters 2 and 3, we infer the GRNs based on linear models. Although owning to the simplicity of linear models, efficient algorithms often exist and they can also be applied to large networks, the real regulatory relationships between genes are in essence nonlinear. Linear models are only approximations to nonlinear ones and therefore can only reveal the biological system to some extent.

From this chapter on, we focus on the nonlinear GRN models, specifically, S-systems which is a set of nonlinear ODEs. As the nonlinarity and complexity of the S-system, the GRN inference from time-course gene expression data becomes extremely difficult. We first consider the parameter estimation of the S-system with the known system structure, as the parameter estimation is usually very important for the structure identification.

In this chapter, a novel method, alternating weighted least squares (AWLS), is developed to estimate the parameters in S-systems. Compared with the existing method, alternating regression (AR) [141], whose objective function is not clear, AWLS is derived from an clear objective function and it aims to decrease the objective function in each iteration. Simulation results show that AWLS outperforms the AR significantly. This chapter partially fulfills Objective 3 of this thesis.

Abstract

The S-system, which is a set of nonlinear ordinary differential equations and derived from the generalized mass action law, is a consistent model to describe various biological systems. Parameters in S-systems contain important biological information and yet can not be obtained directly from experiments. Therefore, the parameter estimation methods are a choice to estimate parameters in S-systems. However, the parameter estimation for this model turns out to be a complex nonlinear optimization problem. A novel method, alternating weighted least squares (AWLS), is proposed in this paper to estimate the parameters in S-systems. The fast deterministic AWLS method takes advantage of the special structure of the S-system model and

reduces solving the nonlinear optimization problem into alternately solving weighed least squares problems which have analytical solutions. The effectiveness of AWLS is demonstrated by the simulation studies and the results show that the AWLS outperforms the existing alternating regression method.

5.1 Introduction

Biological systems, such as metabolic pathways and genetic regulatory networks, consist of many components and the interactions between them. One task of systems biology is to reveal the interactions and the biological functions those interactions may result in [142]. Instead of focusing on individual components, systems biology applies system engineering methods and principles to study all components and their interactions as parts of a biological system. Such a systematic view provides an insight into the control and optimization of parts of the system while considering the effects those may have on the whole system. It may lead to the discovery of new properties of a biological system, which helps understand the mechanisms of biological systems, and valuable clues and new ideas in practical areas such as disease treatment and drug design [20].

Many mathematical models have been proposed to describe the molecular biological systems based on biochemical principles. Most models are nonlinear in both parameters and system state variables [142, 60]. Estimation of parameters in those models are thus formulated as nonlinear optimization problems which generally have no analytical solutions. One popular model is the S-system, which is nonlinear and derived from the generalized mass action law [60].

An S-system with N components is a type of power-law formalism and typically a group of nonlinear ordinary differential equations in the following format:

$$\dot{X}_i = \alpha_i \prod_{j=1}^{N} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_j^{h_{ij}}, \quad i = 1, \ldots, N, \tag{5.1}$$

where X_i represents the concentration of metabolite i, whose changes are the difference between production and degradation, α_i and β_i are non-negative rate constants, and g_{ij} and h_{ij} are real-valued kinetic orders. It is an effective mathematical framework to characterize and analyze the molecular biological systems and their system dynamics. The representation of this model maps the dynamical and topological information of the system onto its parameters.

Parameter estimation and structure identification of S-system models are extremely difficult and challenging tasks, where the parameter estimation usually occurs after or in the process of structure identification. As the estimation of parameters in S-systems is a nonlinear problem, in principle, all algorithms for nonlinear optimization problems can be used, for example, Gauss-Newton iteration method, and its variants such as Box-Kanemasu interpolation method, Levenberg damped least squares method, and Marquardt's method [143]. However, these methods are initial-sensitive and most of them need to calculate the inverse of the Hessian which costs computation effort.

Several numerical methods have been proposed in the literature to estimate the parameters in S-systems, most of which are based on heuristics. For example, Kikuchi et al. [61] employed a genetic algorithm to infer the S-systems. Gonzalez et al. [144] showed the effectiveness of the simulated annealing technique. Voit and Almeida [59] developed an ANN-based method to identify the structure and estimate the parameters of S-systems. Ho et al. [63] and Wang et al. [145] respectively proposed an intelligent two-stage evolutionary algorithm and an unified approach to estimate the parameters in S-systems. Those methods are computationally expensive and do not sufficiently take the special model structure of the S-system into account.

Wu and Mu [146] introduced a separable parameter estimation method which takes advantage of the structure of the S-system model, i.e., one group of parameters is linear in model while the other group is nonlinear. This method has been extended to the case when system topology is unknown with a genetic algorithm by Liu et al. [64, 147]. One observation of the S-system is that if the parameters in one term on the right hand side of (5.1) is known, this term can be moved to the left side and a linear model is obtained by taking logarithm of both sides. Based on this observation, an alternating regression (AR) method was proposed by Chou et al. [141], which reduces the nonlinear estimation problem into the iterative procedures of linear regression. However, the objective of the iterations is vague and the necessary and sufficient criteria for convergence are not known. Inspired by the idea of AR, Vilela et al. [65] proposed a novel method based on eigenvector optimization of a matrix formed from multiple regression equations of the linearized decoupled S-systems, which, however, involves an nonlinear optimization problem.

In this paper, an alternating weighted least squares (AWLS) method is proposed. AWLS is a fast deterministic method and aims at reducing the nonlinear optimization problem into a series of easily solved problems, the idea of which is similar as AR's [141]. AWLS starts from the nonlinear least squares objective which can be approximated by a quadratic function with the assumption that part of the parameters are known. The approximated function turns out to be a weighted least squares problem which has an analytical solution. With the solution of the approximated problem, the other part of the parameters can also be estimated or further updated by forming another weighted least squares problem. AWLS takes advantage of the special form of S-systems and has a more clear objective than AR.

Briefly, the paper is organized as follows. In Section 5.2, the AWLS method is introduced and derived. In Section 5.3, the AWLS approach is applied to estimate the parameters of S-systems. The performance of AWLS is also compared with that of AR. Finally, in Section 5.4, conclusions are drawn and some future works along this research are pointed out.

5.2 Alternating Weighted Least Squares

Consider a biological system with N components described by an S-system in (5.1). For each component X_i, time series data consisting of n time points, $x_{i1}, x_{i2}, \ldots, x_{in}$, are assumed to be observed. The purpose is to estimate the parameters in (5.1) from these observed data. We substitute the derivative of X_i at each time

t with the estimated slope, S_{it}, so that the original coupled differential equations are decoupled into $n \times N$ uncoupled algebraic equations [65, 59]:

$$S_{it} = \alpha_i \prod_{j=1}^{N} x_{jt}^{g_{ij}} - \beta_i \prod_{j=1}^{N} x_{jt}^{h_{ij}}, \tag{5.2}$$

where $i = 1, \ldots, N$ and $t = 1, \ldots, n$. The estimation of slopes is a crucial step and may have effects on the final results. To increase the accuracy, the five-point numerical derivative method is employed in this study, i.e.,

$$S_{it} = \frac{-x_{i,t+2} + 8x_{i,t+1} - 8x_{i,t-1} + x_{i,t-2}}{12\Delta t}, \tag{5.3}$$

where Δt is the length of sampling step.

Generally, the sum of least squares is used as a criterion to determine the values of parameters, i.e., parameters in each equation i of (5.1) are estimated by minimizing the following objective:

$$J_i(\alpha_i, \beta_i, g_i, h_i) = \sum_{t=1}^{n} \left[S_{it} - \alpha_i \prod_{j=1}^{N} x_{jt}^{g_{ij}} + \beta_i \prod_{j=1}^{N} x_{jt}^{h_{ij}} \right]^2, \tag{5.4}$$

where $g_i = [g_{i1}, \ldots, g_{iN}]^T$ and $h_i = [h_{i1}, \ldots, h_{iN}]^T$. Suppose values of β_i and h_i are given and let

$$D_{it} = S_{it} + \beta_i \prod_{j=1}^{N} x_{jt}^{h_{ij}} \text{ and } P_{it} = \alpha_i \prod_{j=1}^{N} x_{jt}^{g_{ij}}.$$

Then, we have

$$\begin{aligned}
J_i(\alpha_i, \beta_i, g_i, h_i) &= \sum_{t=1}^{n} (D_{it} - P_{it})^2 \\
&= \sum_{t=1}^{n} \left[e^{\log D_{it}} \left(1 - e^{\log P_{it} - \log D_{it}} \right) \right]^2 \\
&= \sum_{t=1}^{n} \left[D_{it} \left(\log D_{it} - \log P_{it} + o(\log D_{it} - \log P_{it}) \right) \right]^2 \\
&= \sum_{t=1}^{n} D_{it}^2 \left(\log D_{it} - \log P_{it} \right)^2 + o\left(\sum_{t=1}^{n} D_{it}^2 \left(\log D_{it} - \log P_{it} \right)^2 \right).
\end{aligned}$$

From (5.2), D_{it} and P_{it} should be close and in the third equality above, the first order Taylor approximation is applied. The last equality shows that $J_i(\alpha_i, \beta_i, g_i, h_i)$ can be minimized if the first term is small enough. Hence, the last term can be omitted.

This study assumes the structure of the system, i.e., the positions of nonzero kinetic orders, is available. From this information, some entries in g_i and h_i are known to be zeros. Let $\tilde{g}_i = [g_{ij_1}, \ldots, g_{ij_p}]^T$ and $\tilde{h}_i = [h_{i\ell_1}, \ldots, h_{i\ell_q}]^T$ denote the vectors of nonzero kinetic orders in g_i and h_i, respectively. α_i and $\tilde{g}_i$ can be

estimated by solving the optimization problem

$$\underset{\alpha_i,\tilde{g}_i}{\text{minimize}} \quad \sum_{t=1}^{n}\left(S_{it}+\beta_i\prod_{j=1}^{N}x_{jt}^{h_{ij}}\right)^2\left[\log\left(S_{it}+\beta_i\prod_{j=1}^{N}x_{jt}^{h_{ij}}\right)-\log\alpha_i-\sum_{k=1}^{p}g_{ij_k}\log x_{j_kt}\right]^2, \tag{5.5}$$

which is a weighted least squares problem. Let

$$\mathbf{W}(\beta_i,h_i)=\text{diag}(D_{i1}^2,\ldots,D_{in}^2),\quad \mathbf{Y}(\beta_i,h_i)=\left[\log D_{i1},\ldots,\log D_{in}\right]^T,\quad \mathbf{\Gamma}_i=[\zeta_i,\tilde{g}_i^T]^T,\ \text{where } \zeta_i=\log\alpha_i,$$

and

$$\mathbf{X}_{g,i}=\begin{bmatrix}1 & \log x_{j_11} & \ldots & \log x_{j_p1}\\ 1 & \log x_{j_12} & \ldots & \log x_{j_p2}\\ \vdots & \vdots & \vdots & \vdots\\ 1 & \log x_{j_1n} & \ldots & \log x_{j_pn}\end{bmatrix}.$$

Then, (5.5) becomes

$$\underset{\mathbf{\Gamma}_i}{\text{minimize}} \quad \|\mathbf{W}(\beta_i,h_i)^{\frac{1}{2}}[\mathbf{Y}(\beta_i,h_i)-\mathbf{X}_{g,i}\mathbf{\Gamma}_i]\|_2^2, \tag{5.6}$$

and the analytical solution is

$$\widehat{\mathbf{\Gamma}}_i=(\mathbf{X}_{g,i}^T\mathbf{W}(\beta_i,h_i)\mathbf{X}_{g,i})^{-1}\mathbf{X}_{g,i}^T\mathbf{W}(\beta_i,h_i)\mathbf{Y}(\beta_i,h_i),\quad \hat{\alpha}_i=\exp(\hat{\zeta}_i). \tag{5.7}$$

Similarly, when α_i and g_i are given, β_i and $\tilde{h}_i$ can be estimated from

$$\underset{\mathbf{\Theta}_i}{\text{minimize}} \quad \|\mathbf{W}(\alpha_i,g_i)^{\frac{1}{2}}[\mathbf{Y}(\alpha_i,g_i)-\mathbf{X}_{h,i}\mathbf{\Theta}_i]\|_2^2, \tag{5.8}$$

where $E_{it}=\alpha_i\prod_{j=1}^{N}x_{jt}^{g_{ij}}-S_{it}$,

$$\mathbf{X}_{h,i}=\begin{bmatrix}1 & \log x_{\ell_11} & \ldots & \log x_{\ell_q1}\\ 1 & \log x_{\ell_12} & \ldots & \log x_{\ell_q2}\\ \vdots & \vdots & \vdots & \vdots\\ 1 & \log x_{\ell_1n} & \ldots & \log x_{\ell_qn}\end{bmatrix},$$

and

$$\mathbf{W}(\alpha_i,g_i)=\text{diag}(E_{i1}^2,\ldots,E_{in}^2),\quad \mathbf{Y}(\alpha_i,g_i)=[\log E_{i1},\ldots,\log E_{in}]^T,\quad \mathbf{\Theta}_i=[\eta_i,\tilde{h}_i^T]^T,\ \text{where } \eta_i=\log\beta_i.$$

The corresponding analytical solution is

$$\widehat{\mathbf{\Theta}}_i=(\mathbf{X}_{h,i}^T\mathbf{W}(\alpha_i,g_i)\mathbf{X}_{h,i})^{-1}\mathbf{X}_{h,i}^T\mathbf{W}(\alpha_i,g_i)\mathbf{Y}(\alpha_i,g_i),\quad \hat{\beta}_i=\exp(\hat{\eta}_i). \tag{5.9}$$

Based on the derivations above, we can see that when part of the parameters in an S-system are known, the rest can be estimated by solving a weighted least squares problem. Thus, given the initial values of one part of the parameters, all parameters in the S-system can be iteratively estimated by alternately solving weighted least squares problems. The objective value J_i is reduced in each iteration and the estimated parameters are obtained when the iterations converge. The proposed AWLS method for each equation i is:

Require: The structure of the system and initial values of β_i and $\tilde{h}_i$,

1: **repeat**

2: Estimate α_i and $\tilde{g}_i$ by (5.7) with known β_i and $\tilde{h}_i$,

3: Estimate β_i and $\tilde{h}_i$ by (5.9) with known α_i and $\tilde{g}_i$,

4: **until** a stopping criteria is met.

In this paper, the stopping criteria is

$$\frac{\|\gamma^{(k)} - \gamma^{(k-1)}\|_2}{\|\gamma^{(k)}\|_2} < \theta, \tag{5.10}$$

or the number of iterations is greater than 10,000, i.e., not convergent. Here, θ is a preset threshold and $\gamma^{(k)} = [\hat{\alpha}_i^{(k)}, \hat{\beta}_i^{(k)}, \hat{g}_i^{(k)T}, \hat{h}_i^{(k)T}]^T$, i.e., the parameter estimations in the kth iteration.

5.3 Simulation and Comparison

5.3.1 Parameter Estimation

4-dimensional model

Consider the following S-system of 4 metabolites [60]:

$$\begin{aligned}
\dot{X}_1 &= 12X_3^{-0.8} - 10X_1^{0.5}, \\
\dot{X}_2 &= 8X_1^{0.5} - 3X_2^{0.75}, \\
\dot{X}_3 &= 3X_2^{0.75} - 5X_3^{0.5}X_4^{0.2}, \\
\dot{X}_4 &= 2X_1^{0.5} - 6X_4^{0.8}.
\end{aligned} \tag{5.11}$$

The noise-free time series data are obtained by numerically solving the S-system with an initial condition $X(0) = [x_{10}, x_{20}, x_{30}, x_{40}]^T$. The data are sampled at time points in the interval $[0, 5]$ with $\Delta t = 0.1$.

In this example, the data are generated with $X(0) = [2.7255, 1.8601, 4.7343, 3.7162]^T$ whose elements are randomly chosen in $[0, 5]$. The time series data are shown in Figure 5.1, from which we can see all states of X_i's are eventually in the steady states. The AWLS method is applied to estimate the parameters from these data with the initial values for β_i and h_i chosen by

$$(\beta_i^{\text{init}}; h_i^{\text{init}}) = (\beta_i^{\text{true}}; h_i^{\text{true}})(1 + \sigma\varepsilon), \tag{5.12}$$

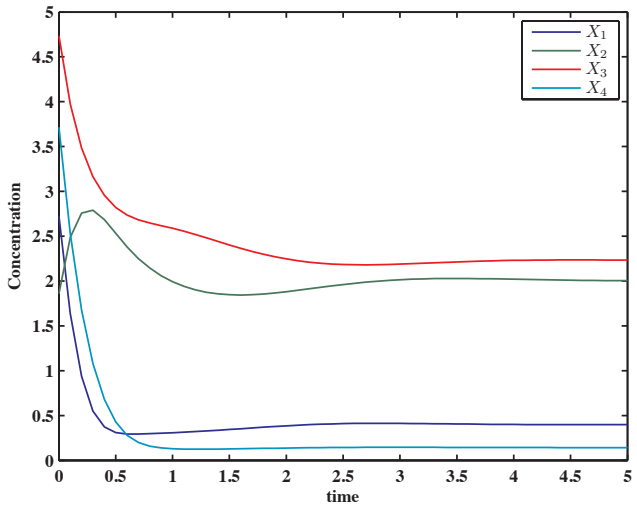


Figure 5.1: Time series data of the 4-dimensional model.

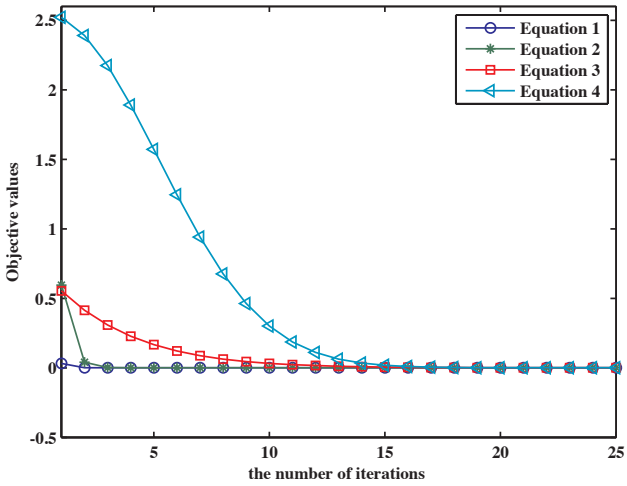


Figure 5.2: Objective values over various numbers of iterations in AWLS.

where ε is a standard Gaussian random variable and σ is a positive constant. Since (5.4) is a nonlinear optimization problem, to avoid falling into the local optimum, we apply AWLS 100 times initiated with different values and choose the best one as the final solution. Here, we set $\sigma = 90\%$, the objective values $J_i^{(k)}$ for each equation i in each iteration k are illustrated in Figure 5.2. It can be seen that the objective values decrease with the increase of iteration steps. Table 5.1 shows the estimated results, from which we can see that the estimated values are quite close to their true values and the optimal objective values are all very small.

Table 5.1: Estimated results of the 4-dimensional model.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	12	12.0922	0.77%	1.4754×10^{-4}
β_1	10	10.1040	1.04%	
g_{13}	-0.8	-0.7893	1.34%	
h_{11}	0.5	0.4936	1.29%	
α_2	8	7.9851	0.19%	2.0490×10^{-4}
β_2	3	2.9830	0.57%	
g_{21}	0.5	0.5025	0.50%	
h_{22}	0.75	0.7522	0.29%	
α_3	3	3.0205	0.68%	1.8454×10^{-4}
β_3	5	5.0550	1.10%	
g_{32}	0.75	0.7531	0.41%	
h_{33}	0.5	0.4963	0.73%	
h_{34}	0.2	0.1996	0.22%	
α_4	2	2.0081	0.41%	1.0699×10^{-4}
β_4	6	5.9883	0.19%	
g_{41}	0.5	0.5157	3.13%	
h_{44}	0.8	0.8045	0.57%	

5-dimensional model

A benchmark 5-dimensional model [1, 65] is considered,

$$\begin{aligned}
\dot{X}_1 &= 5X_3X_5^{-1} - 10X_1^2, \\
\dot{X}_2 &= 10X_1^2 - 10X_2^2, \\
\dot{X}_3 &= 10X_2^{-1} - 10X_2^{-1}X_3^2, \\
\dot{X}_4 &= 8X_3^2X_5^{-1} - 10X_4^2, \\
\dot{X}_5 &= 10X_4^2 - 10X_5^2.
\end{aligned} \tag{5.13}$$

The data used in this example are generated with the initial condition $X(0) = [0.1, 0.7, 0.7, 0.16, 0.18]^T$, the same in Yang et al. [1]. Figure 5.3 shows the time series data that are sampled in the interval $[0, 0.5]$ with $\Delta t = 0.01$. Note that the states of all variables quickly converge to the steady state. Hence, only limited information on the dynamics of the system is contained in the data. We run AWLS 100 times with different initial values obtained from (5.12) with $\sigma = 80\%$ and select the best one as the solution. The results in Table 5.2 show the effectiveness of AWLS: estimated values of parameters are close to the true values and the optimal objective values are all very small.

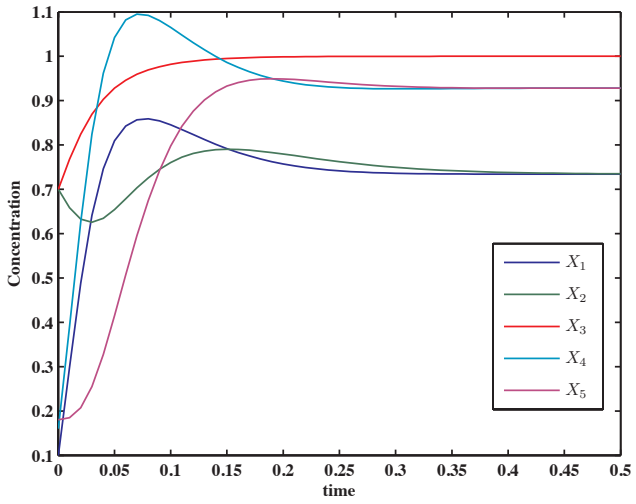


Figure 5.3: Time series data of the 5-dimensional model.

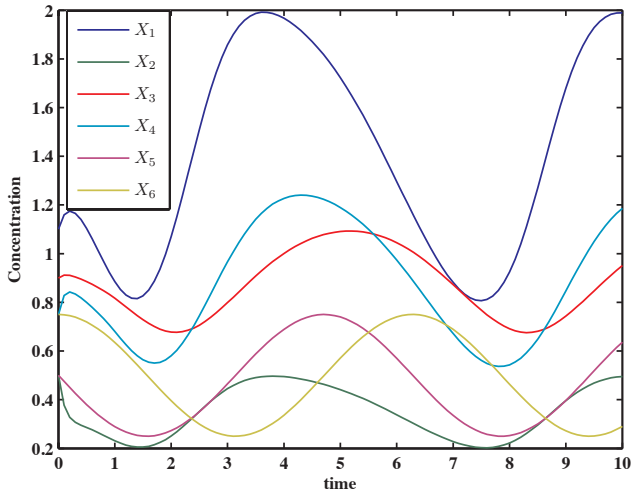


Figure 5.4: Time series data of the 6-dimensional model.

6-dimensional model

In this example, the AWLS method is applied to estimate the parameters in the following 6-dimensional S-system [1]:

$$\begin{aligned}
\dot{X}_1 &= 10X_3^{-2}X_5 - 5X_1^{0.5},\\
\dot{X}_2 &= 5X_1^{0.5} - 10X_2^{0.5},\\
\dot{X}_3 &= 2X_2^{0.5} - 1.25X_3^{0.5},\\
\dot{X}_4 &= 8X_2^{0.5} - 5X_4^{0.5},\\
\dot{X}_5 &= 0.5 - X_6,\\
\dot{X}_6 &= X_5 - 0.5.
\end{aligned} \tag{5.14}$$

Table 5.2: Estimated results of the 5-dimensional model.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	5	4.8473	3.05%	6.0840×10^{-4}
β_1	10	9.8529	1.47%	
g_{13}	1	1.1337	13.37%	
g_{15}	-1	-1.0316	3.16%	
h_{11}	2	2.0443	2.21%	
α_2	10	10.0039	0.04%	8.0113×10^{-4}
β_2	10	9.9626	0.37%	
g_{21}	2	2.0049	0.25%	
h_{22}	2	1.9898	0.51%	
α_3	10	9.9699	0.30%	1.5504×10^{-4}
β_3	10	9.9711	0.29%	
g_{32}	-1	-0.9686	3.14%	
h_{32}	-1	-0.9684	3.16%	
h_{33}	2	2.0411	2.05%	
α_4	8	7.5653	5.43%	1.1×10^{-3}
β_4	10	9.5512	4.49%	
g_{43}	2	2.1869	9.35%	
g_{45}	-1	-1.0490	4.90%	
h_{44}	2	2.0824	4.12%	
α_5	10	9.9910	0.09%	1.4×10^{-3}
β_5	10	9.9770	0.23%	
g_{54}	2	2.0121	0.60%	
h_{55}	2	1.9917	0.42%	

The noise-free time series data are generated with the initial condition $X(0) = [1.1, 0.5, 0.9, 0.75, 0.5, 0.75]^T$ which matches that in Yang et al. [1]. Figure 5.4 illustrates the time series data which are sampled in the interval $[0, 10]$ with $\Delta t = 0.1$. Figure 5.4 also shows the periodic oscillating behavior of the data. The initial values for β_i and h_i are also chosen by (5.12) with $\sigma = 50\%$. The solution shown in Table 5.3 is the best one among 100 runs of AWLS with different initial values. The results in Table 5.3 indicate that the estimated parameters are close to their true values and the optimal objective values are all very small.

5.3.2 Comparison

The performances of AWLS and AR [141] are compared based on the previous 4-dimensional model (5.11) in the following procedure: (i) Fix a value of σ and randomly generate an initial condition $X(0)$ in $[0, 5]$. (ii) Obtain the noise-free data in the interval $[0, 5]$ with $\Delta t = 0.1$. (iii) Generate 100 initial values of $(\beta_i; h_i)$ by (5.12). (iv) Apply AWLS and AR to each equation i with each initial value, respectively. Therefore, each method has 100 results for each equation. (v) Remove those results which do not satisfy $\alpha_i, \beta_i \in [0.1, 12]$

Table 5.3: Estimated results of the 6-dimensional model.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	10	10.4440	4.44%	2.0×10^{-3}
β_1	5	5.5333	10.67%	
g_{13}	-2	-1.8332	8.43%	
g_{15}	1	0.9163	8.37%	
h_{11}	0.5	0.4580	8.40%	
α_2	5	5.3370	6.74%	7.3143×10^{-4}
β_2	10	10.3732	3.73%	
g_{21}	0.5	0.4793	4.14%	
h_{22}	0.5	0.4794	4.12%	
α_3	2	2.0485	2.42%	1.8097×10^{-4}
β_3	1.25	1.3000	4.00%	
g_{32}	0.5	0.4835	3.29%	
h_{33}	0.5	0.4840	3.20%	
α_4	8	7.9549	0.56%	1.2×10^{-3}
β_4	5	5.0024	0.05%	
g_{42}	0.5	0.4934	1.33%	
h_{44}	0.5	0.4932	1.35%	
α_5	0.5	0.4928	1.43%	1.3908×10^{-5}
β_5	1	0.9957	0.43%	
h_{56}	1	1.0160	1.60%	
α_6	1	0.9955	0.45%	1.5696×10^{-5}
β_6	0.5	0.4925	1.51%	
g_{65}	1	1.0170	1.70%	

or $g_{ij}, h_{ij} \in [-2, 3]$ (cf. [65, 60]) or not converge. Choose the result which has the the minimum objective value as the best one. (vi) For each method, put the best results of each equation together to form the final solution of the S-system. We run the aforementioned procedure (i)–(vi) with different $X(0)$'s and different σ's. In this experiment, we have 30 different $X(0)$'s and for each $X(0)$, σ's vary from 0 to 1 with step 0.1. Thus, each method has 30×11 results. We denote the results for the lth data, which are generated by $X^l(0)$, and $\sigma = \tau$ from AWLS and AR by $\gamma_{AWLS}^{(l,\tau)}$ and $\gamma_{AR}^{(l,\tau)}$, respectively, where $l = 1, \ldots, 30$ and $\tau = 0, 0.1, \ldots, 1.0$.

AWLS and AR are compared from two perspectives: the estimation error (EstErr) and the objective value (ObjVal).

$$\text{EstErr}^{(l,\tau)} = \frac{\|\gamma^{(l,\tau)} - \gamma_{true}\|_2^2}{\|\gamma_{true}\|_2^2}, \quad \text{ObjVal}^{(l,\tau)} = \sum_{i=1}^{N} J_i(\gamma^{(l,\tau)})$$

Figure 5.5 shows the mean estimation errors of AWLS and AR with respect to each value of σ, respectively. Figure 5.6 describes the mean objective values with respect to each value of σ. We can see that both the mean estimation error and the mean objective value grow with the increase of σ. In addition, both the mean

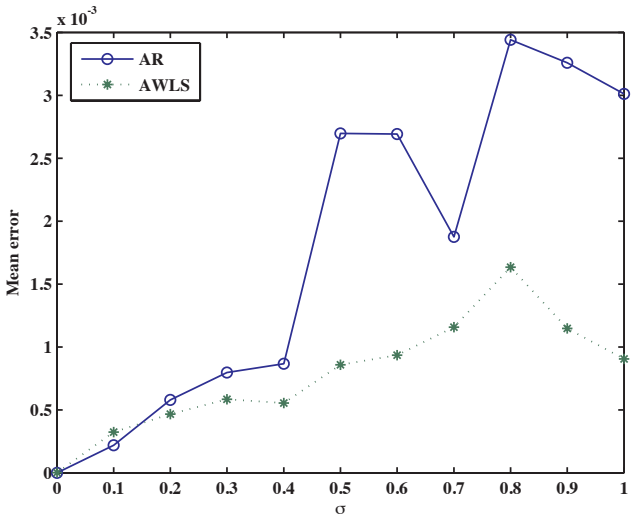


Figure 5.5: Comparison of AWLS and AR w.r.t. mean estimation errors.

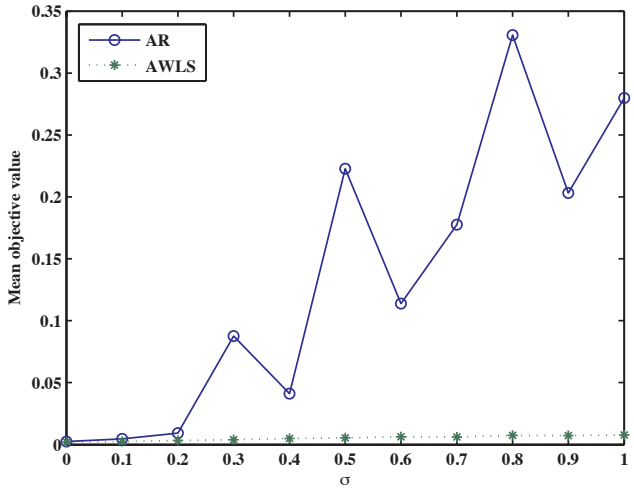


Figure 5.6: Comparison of AWLS and AR w.r.t. mean objective values.

estimation error and mean objective value of AWLS are in general less than those of AR, especially for large σ's.

To further confirm the conclusion, note that there are totally 330 cases (30 $X^k(0)$'s and 11 σ's) and in each case, the initial values for AWLS and AR are the same. Therefore, the 330 $\text{EstErr}^{(l,\tau)}$'s and 330 $\text{ObjVal}^{(l,\tau)}$'s of AWLS can be compared with those of AR, respectively, by paired hypothesis tests. The null hypotheses for the estimation error and the objective value are:

$$H_0 : \text{EstErr of AWLS} \geq \text{EstErr of AR};$$
$$H_0 : \text{ObjVal of AWLS} \geq \text{ObjVal of AR}.$$

We perform the paired t-test and paired Wilcoxon signed-rank test for the estimation error and objective

Table 5.4: Paired t-test (t) and paired Wilcoxon signed-rank test (V).

	t	df	p-value	V	p-value
EstErr	3.9859	329	4.143e-05	34151	3.985e-05
ObjVal	5.1832	329	1.907e-07	36958	1.321e-08

value, respectively. The null hypothesis of paired Wilcoxon test is similar as that of paired t-test but on medians and no normality assumption is required. The hypothesis test results are shown in Table 5.4, in which the p-values indicate that the estimation error and objective value of AWLS are significantly less than those of AR. Therefore, AWLS outperforms AR.

5.4 Conclusions

This paper has proposed an AWLS method to estimate the parameters in biological S-systems. AWLS takes advantage of the special structure of S-systems and is a fast deterministic method with low computational cost. The superb efficiency comes from the reduction of the complex nonlinear optimization problem into alternating weighted least squares problems. There is no need to compute the inverse of the Hessian matrix and only part of the parameters require initial values. The dimension of search space of parameters are hence reduced. The simulation results show that AWLS can find the values of parameters in S-systems and AWLS outperform AR, i.e., it has less estimation error and objective value than those of AR.

In this study, the structure of the system is assumed to be known. One direction of the future work is to extend AWLS with Lasso approach [148, 15] to infer the S-system without knowing the system structure.

Acknowledgment

This study is supported by Natural Science and Engineering Research Council of Canada (NSERC).

Chapter 6

Estimating Parameters of S-systems by an Auxiliary Function Guided Coordinate Descent Method

Published as: L.Z. Liu, F.X. Wu, and W.J. Zhang, "Estimating parameters of S-systems by an auxiliary function guided coordinate descent method," *Systems Science & Control Engineering*, vol. 2, no. 1, pp. 125–134, 2014 [149].

In previous chapter, we have developed an AWLS method to estimate the parameters of S-systems from time-course data. The iteration formula is derived from an objective function by assuming part of the parameters are known. Although it shows that AWLS outperforms the AR, its convergence cannot be guaranteed. If the method does not converge, we need rerun the algorithm with another initial values, which is inefficient in practice.

In this chapter, a novel method, auxiliary function guided coordinate descent (AFGCD), is developed to estimate the parameters in S-systems from time-course data. The proposed method designs an auxiliary function, by optimizing which the objective function keep decreasing. We cyclically optimize the auxiliary function and therefore obtain simple and efficient iteration formulas. This chapter partially fulfills Objective 3 of this thesis.

Abstract

The S-system, a set of nonlinear ordinary differential equations and derived from the generalized mass action law, is an effective model to describe various biological systems. Parameters in S-systems have significant biological meanings, yet difficult to be estimated because of the nonlinearity and complexity of the model. Given time series biological data, its parameter estimation turns out to be a nonlinear optimization problem. A novel method, auxiliary function guided coordinate descent, is proposed in this paper to solve the optimization problem by cyclically optimizing every parameter. In each iteration, only one parameter value is updated and it proves that the objective function keeps nonincreasing during the iterations. The updating rules in each iteration is simple and efficient. Based on this idea, two algorithms are developed to estimate the S-systems for two different constraints situations. The performances of algorithms are studied in several simulation examples. The results demonstrate the effectiveness of the proposed method.

6.1 Introduction

There are many various molecules and productions in a cell. Some of them can regulate others via some mechanisms to achieve specific cellular functions, based on which the cells adapt to the changing environments. These components and their interactions constitute biological systems, such as metabolic pathways and genetic regulatory networks. One task of systems biology is to reveal the interactions and the biological functions those interactions may result in [142]. Instead of focusing on individual components, systems biology applies system engineering methods and principles to studying all components and their interactions as parts of a biological system. Such a systematic view provides an insight into the control and optimization of parts of the systems while taking the effects those may have on the whole system into account. It is of help to the discovery of new properties of biological systems and may provide valuable clues and new ideas in practical areas such as disease treatment and drug design [20].

One effective way to study the biological system is using mathematical or computational methods. Many mathematical models have been developed to describe the molecular biological systems based on biochemical principles. Most of them are nonlinear in both parameters and state variables [142, 60]. Nonlinear optimization problems are usually formulated for estimating the parameters in those models. However, analytical solutions to those problems are hardly available. One of the most popular models is the S-system which is highly nonlinear and derived from the generalized mass action law [60].

The S-system is an effective mathematical framework to characterize and analyze the molecular biological systems. An S-system consisting of N components is type of power-law formalism and typically a group of nonlinear ordinary differential equations,

$$\dot{X}_i = \alpha_i \prod_{j=1}^{N} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_j^{h_{ij}}, \quad i = 1, \ldots, N, \tag{6.1}$$

where X_i represents the concentration of molecular species i measured at time t, whose changes are the difference between production and degradation. α_i's and β_i's are non-negative rate constants, and g_{ij}'s and h_{ij}'s are real-valued kinetic orders that reflect the interaction intensity of X_j to X_i. If $g_{ij} > 0$, X_j activates the production of X_i; if $g_{ij} < 0$, X_j inhibits the production of X_i. h_{ij} has the same effects but on the degradation. A zero-valued kinetic order indicates that X_j has no such effect on X_i. The representation of this model maps the dynamical and topological information of the biological system onto its parameters.

The parameter estimation and structure identification of the S-systems are extremely difficult tasks. The parameter estimation usually occurs after or in the process of structure identification. As the parameter estimation of the S-systems is essentially a nonlinear problem, in principle, all nonlinear optimization algorithms can be used, e.g. Gauss-Newton method and its variants, such as Box-Kanemasu interpolation method, Levenberg damped least squares method, and Marquardt's method [143]. However, most of these methods are initial-sensitive and need to calculate the inverse of the Hessian which is computational expensive.

Several numerical methods have been proposed to estimate the parameters in S-systems. Most of them are based on heuristic methods. For example, [61] employ a genetic algorithm to infer the S-systems. The effectiveness of the simulated annealing is studied in [144]. [59] develop an ANN-based method to identify and estimate the parameters of S-systems. [63] develop an intelligent two-stage evolutionary algorithm that combines the genetic algorithm and the simulated annealing. An unified approach has been proposed in [145]. Most of these methods are computational expensive and do not sufficiently take advantage of special model structure of the S-systems.

Several methods taking the model structure of S-systems into account have been proposed. [146] introduce a separable parameter estimation method which divides the parameters into two groups: one group is linear in model while the other nonlinear. This method has been extended with a genetic algorithm to the case when system topology is unavailable in [64]. One can observe that if parameters in one term on the right hand side of equation (6.1) is known, moving it to the left and taking logarithm of both sides, a linear model is obtained. Using this observation, an alternating regression method is proposed by [141], which reduces the nonlinear optimization into iterative procedures of linear regression. However, the convergence of the iterations can not be guaranteed. Similarly, an alternating weighted least squares method is proposed by [140], in which the objective function to be optimized is approximated by a weighted linear regression in each iteration. However, this approximation may fail in some cases.

This study focus on the parameter estimation of S-systems when system topology is known. A novel method, auxiliary function guided coordinate descent (AFGCD), is proposed. After decoupling the S-systems by replacing the derivatives with numerical slopes, the parameter estimation is formulated as a nonlinear optimization problem. AFGCD iteratively solves this problem and in each iteration only one parameter is optimized with other parameters fixed. Instead of directly optimizing the objective function, AFGCD takes advantage of a property of the exponential function and constructs an auxiliary function for each kinetic order parameter to optimize. It shows that optimizing the auxiliary function makes the objective function keep nonincreasing. Because the auxiliary function is simple and its optimization has analytical solution, the parameter updating in each iteration is simple and efficient. Based on this idea, two algorithms are developed to estimate the parameters in S-systems under two different situations. One algorithm is developed for the case when the range of each parameter is known and the other algorithm for the case that the only constraint is the non-negativity of the rate constants.

The remaining of this paper is organized as follows. In Section 6.2, two algorithms are developed based on the idea of AFGCD and their descent properties are proven. In Section 6.3, the effectiveness of the proposed algorithms are studied by several simulation examples. Finally, Section 6.4 concludes this study and points out some future works along this research.

6.2 Auxiliary Function Guided Coordinate Descent

6.2.1 Problem Statement

Consider a biological system of N molecular species, described by an S-system as equation (6.1), and assume that for each molecular species X_i, a time series concentration data measured at n equally-spaced time points, $x_{i1}, \ldots, x_{in}$ are obtained. This study assumes that the topology of the system is available, i.e., the zero-values kinetic orders are known. The purpose is to estimate the parameters of nonzero-valued kinetic orders and rate constants. We substitute the derivative of X_i at time t with the estimated slope, S_{it}, so that the original coupled ordinary differential equations are decoupled into $n \times N$ uncoupled algebraic equations [59, 65]:

$$S_{it} = \alpha_i \prod_{j=1}^{N} x_{jt}^{g_{ij}} - \beta_i \prod_{j=1}^{N} x_{jt}^{h_{ij}} + \epsilon_{it}, \tag{6.2}$$

where $i = 1, \ldots, N$, $t = 1, \ldots, n$ and ϵ_{it}'s are errors or noises. The estimation of slopes is a crucial step and may affect the final results. To increase the accuracy, the five-point numerical derivative method is used in this study, i.e.,

$$S_{it} = \frac{-x_{i,t+2} + 8x_{i,t+1} - 8x_{i,t-1} + x_{i,t-2}}{12\Delta t} \tag{6.3}$$

where Δt is the length of the sampling step.

To determine the parameter values, the sum of least squares is usually employed as an objective function to be minimized, i.e., the parameters of each ODE equation i in (6.1) is obtained by solving

$$\begin{aligned} &\underset{\alpha_i, \beta_i, g_i, h_i}{\text{minimize}} \quad J_i = \frac{1}{2} \sum_{t=1}^{n} \left[S_{it} - \alpha_i \prod_{j \in G_i} x_{jt}^{g_{ij}} + \beta_i \prod_{j \in H_i} x_{jt}^{h_{ij}} \right]^2, \\ &\text{subject to} \quad \alpha_i \geq \delta, \quad \beta_i \geq \delta \end{aligned} \tag{6.4}$$

where δ is predefined small positive number; G_i and H_i are the sets of indexes of molecular species which have effects on the production and degradation of molecular species i, respectively; $g_i = \{g_{ij}, j \in G_i\}$ and $h_i = \{h_{ij}, j \in H_i\}$. The minimization of equation (6.4) is non-trivial, as it is highly nonlinear and contains a lot of parameters.

6.2.2 Proposed Method

The optimization problem (6.4) has no analytical solutions and therefore, iterative methods are considered. A coordinate descent strategy is applied in this study, i.e., we cyclically optimize one parameter in one iteration with other parameters fixed such that the objective function keeps nonincreasing in every iteration. More specifically, denote by $\boldsymbol{\theta}^{(\ell)} = (\theta_1^{(\ell)}, \ldots, \theta_p^{(\ell)})$ the parameters of an optimization problem (e.g. for (6.4), $\boldsymbol{\theta} = (\alpha_i, g_i, \beta_i, h_i)$) at iteration ℓ. After one iteration, the first parameter is updated, $\boldsymbol{\theta}_1^{(\ell+)} = (\theta_1^{(\ell+1)}, \theta_2^{(\ell)}, \ldots, \theta_p^{(\ell)})$

and after k iterations, the first k parameters are updated, $\boldsymbol{\theta}_k^{(\ell+)} = (\theta_1^{(\ell+1)}, \ldots, \theta_k^{(\ell+1)}, \theta_{k+1}^{(\ell)}, \ldots, \theta_p^{(\ell)})$.

Consider the problem (6.4) and suppose the parameter h_{ik}, for some $k \in H_i$, needs to be updated at the current iteration, one simple way to make sure the objective function keeps nonincreasing is to use the classical gradient descent method:

$$h_{ik}^{(\ell+1)} = h_{ik}^{(\ell)} - d_\ell \frac{\partial J_i(h_{ik}^{(\ell)})}{\partial h_{ik}}, \tag{6.5}$$

where values of other parameters are fixed and d_ℓ is the stepsize taken along the negative gradient direction. Some methods can be used to search and select the stepsize, such as minimization rule and Armijo rule [150, 151]. However, these methods may either depend on a line search algorithm or introduce an extra iteration loop.

In this paper, a smart stepsize is derived. The introduced stepsize is computational efficient and the nonincrease of the objective function in each iteration is guaranteed, which is proved in the following subsection. The algorithm for solving the problem (6.4) is illustrated in Algorithm 1, in which the updates for each parameter are shown in equations (6.6)–(6.9). To check the convergence, the following stopping criteria is used

$$\frac{\|\boldsymbol{\theta}^{(\ell+1)} - \boldsymbol{\theta}^{(\ell)}\|_2}{\|\boldsymbol{\theta}^{(\ell)}\|_2} < \eta,$$

here η is a preset threshold. In this paper, we set $\eta = 10^{-5}$.

In some situations, from experiments, the range of each parameter is known. Then, more constrains can be added to the problem (6.4) as follows,

$$\begin{aligned} \underset{\alpha_i,\beta_i,g_i,h_i}{\text{minimize}} \quad & J_i = \frac{1}{2}\sum_{t=1}^{n}\left[S_{it} - \alpha_i \prod_{j\in G_i} x_{jt}^{g_{ij}} + \beta_i \prod_{j\in H_i} x_{jt}^{h_{ij}}\right]^2, \\ \text{subject to} \quad & r_{\min} \le \alpha_i \le r_{\max}, \quad r_{\min} \le \beta_i \le r_{\max}, \\ & k_{\min} \le g_{ij} \le k_{\max}, j \in G_i, \\ & k_{\min} \le h_{ij} \le k_{\max}, j \in H_i, \end{aligned} \tag{6.10}$$

that is, we require the rate constants stay in the range $[r_{\min}, r_{\max}]$ and kinetic orders in the range $[k_{\min}, k_{\max}]$. Simple adaptations to Algorithm 1 lead to Algorithm 2 for solving the problem (6.10). The update rule (6.11) and (6.13) are slight modifications to (6.6) and (6.8). They efficiently make the objective function keep nonincreasing, which is proved in the following subsection.

6.2.3 Descent Property

Similar to [135] and [152], we make use of an auxiliary function to prove the nonincrease of the objective function in each iteration.

Algorithm 1 AFGCD for problem (6.4)

1. Assign initial values to $\boldsymbol{\theta}^{(0)} = (\alpha_i^{(0)}, g_i^{(0)}, \beta_i^{(0)}, h_i^{(0)})$ and set $\ell = 0$.

2. Let $y_t = -S_{it} - \beta_i^{(\ell)} \prod_{j \in H_i} x_{jt}^{h_{ij}^{(\ell)}}$, $t = 1, \ldots, n$ and cyclically update $g_{ik}^{(\ell)}$ to $g_{ik}^{(\ell+1)}$ for each $k \in G_i$ as follows
$$g_{ik}^{(\ell+1)} = g_{ik}^{(\ell)} - \frac{1}{2\tau_g(\sigma_g^{(\ell)})} \frac{\partial J_i(g_{ik}^{(\ell)})}{\partial g_{ik}}, \tag{6.6}$$
where
$$\tau_g(\sigma_g^{(\ell)}) = \sum_t a_t |y_t| x_{kt}^{g_{ik}^{(\ell)}} (\log^2 x_{kt}) \Delta(\sigma_g^{(\ell)} |\log x_{kt}|) + 2 \sum_t a_t^2 x_{kt}^{2g_{ik}^{(\ell)}} (\log^2 x_{kt}) \Delta(2\sigma_g^{(\ell)} |\log x_{kt}|)$$
$$\sigma_g^{(\ell)} = \frac{\left| \frac{\partial J_i(g_{ik}^{(\ell)})}{\partial g_{ik}} \right|}{\sum_t a_t |y_t| x_{kt}^{g_{ik}^{(\ell)}} \log^2 x_{kt} + 2 \sum_t a_t^2 x_{kt}^{2g_{ik}^{(\ell)}} \log^2 x_{kt}},$$
$$a_t = \alpha_i^{(\ell)} \prod_{j \in G_i, j \neq k} x_{jt}^{g_{ij}^{(\ell+1)I(j<k) + (\ell) I(j \geq k)}}$$
and the function $\Delta(\cdot)$ is defined in equation (6.18).

3. Compute
$$\alpha_i^{(\ell+1)} = \max \left\{ -\frac{\sum_t y_t p_t}{\sum_t p_t^2}, \delta \right\}, \text{ where } p_t = \prod_{j \in G_i} x_{jt}^{g_{ij}^{(\ell+1)}} \tag{6.7}$$

4. Let $y_t = S_{it} - \alpha_i^{(\ell+1)} \prod_{j \in G_i} x_{jt}^{g_{ij}^{(\ell+1)}}$, $t = 1, \ldots, n$ and cyclically update $h_{ik}^{(\ell)}$ to $h_{ik}^{(\ell+1)}$ for each $k \in H_i$ as follows
$$h_{ik}^{(\ell+1)} = h_{ik}^{(\ell)} - \frac{1}{2\tau_h(\sigma_h^{(\ell)})} \frac{\partial J_i(h_{ik}^{(\ell)})}{\partial h_{ik}}, \tag{6.8}$$
where
$$\tau_h(\sigma_h^{(\ell)}) = \sum_t b_t |y_t| x_{kt}^{h_{ik}^{(\ell)}} (\log^2 x_{kt}) \Delta(\sigma_h^{(\ell)} |\log x_{kt}|) + 2 \sum_t b_t^2 x_{kt}^{2h_{ik}^{(\ell)}} (\log^2 x_{kt}) \Delta(2\sigma_h^{(\ell)} |\log x_{kt}|)$$
$$\sigma_h^{(\ell)} = \frac{\left| \frac{\partial J_i(h_{ik}^{(\ell)})}{\partial h_{ik}} \right|}{\sum_t b_t |y_t| x_{kt}^{h_{ik}^{(\ell)}} \log^2 x_{kt} + 2 \sum_t b_t^2 x_{kt}^{2h_{ik}^{(\ell)}} \log^2 x_{kt}},$$
and
$$b_t = \beta_i^{(\ell)} \prod_{j \in H_i, j \neq k} x_{jt}^{h_{ij}^{(\ell+1)I(j<k) + (\ell) I(j \geq k)}}$$

5. Compute
$$\beta_i^{(\ell+1)} = \max \left\{ -\frac{\sum_t y_t d_t}{\sum_t d_t^2}, \delta \right\}, \text{ where } d_t = \prod_{j \in H_i} x_{jt}^{h_{ij}^{(\ell+1)}} \tag{6.9}$$

6. Let $\boldsymbol{\theta}^{(\ell+1)} = (\alpha_i^{(\ell+1)}, g_i^{(\ell+1)}, \beta_i^{(\ell+1)}, h_i^{(\ell+1)})$ and check the convergence. If converged, output $\boldsymbol{\theta}^{(\ell+1)}$, otherwise, $\ell \leftarrow \ell + 1$ and go to 2.

Algorithm 2 AFGCD for problem (6.10)

1. Assign initial values to $\boldsymbol{\theta}^{(0)} = (\alpha_i^{(0)}, g_i^{(0)}, \beta_i^{(0)}, h_i^{(0)})$ and set $\ell = 0$.

2. Let $y_t = -S_{it} - \beta_i^{(\ell)} \prod_{j \in H_i} x_{jt}^{h_{ij}^{(\ell)}}$, $t = 1, \ldots, n$ and cyclically update $g_{ik}^{(\ell)}$ to $g_{ik}^{(\ell+1)}$ for each $k \in G_i$ as follows
$$g_{ik}^{(\ell+1)} = \min\left(\max\left(g_{ik}^{(\ell)} - \frac{1}{2\tau_g(\sigma_g^{(\ell)})}\frac{\partial J_i(g_{ik}^{(\ell)})}{\partial g_{ik}}, L\right), R\right), \tag{6.11}$$
where $L = \max(g_{ik}^{(\ell)} - \sigma_g^{(\ell)}, k_{\min})$, $R = \min(g_{ik}^{(\ell)} + \sigma_g^{(\ell)}, k_{\max})$ and $\tau_g(\sigma_g^{(\ell)})$ is calculated in the same way as in Algorithm 1

3. Compute
$$\alpha_i^{(\ell+1)} = \max\left\{-\frac{\sum_t y_t p_t}{\sum_t p_t^2}, \delta\right\}, \text{ where } p_t = \prod_{j \in G_i} x_{jt}^{g_{ij}^{(\ell+1)}} \tag{6.12}$$

4. Let $y_t = S_{it} - \alpha_i^{(\ell+1)} \prod_{j \in G_i} x_{jt}^{g_{ij}^{(\ell+1)}}$, $t = 1, \ldots, n$ and cyclically update $h_{ik}^{(\ell)}$ to $h_{ik}^{(\ell+1)}$ for each $k \in H_i$ as follows
$$h_{ik}^{(\ell+1)} = \min\left(\max\left(h_{ik}^{(\ell)} - \frac{1}{2\tau_h(\sigma_h^{(\ell)})}\frac{\partial J_i(h_{ik}^{(\ell)})}{\partial h_{ik}}, L\right), R\right), \tag{6.13}$$
where $L = \max(h_{ik}^{(\ell)} - \sigma_h^{(\ell)}, k_{\min})$ and $R = \min(h_{ik}^{(\ell)} + \sigma_h^{(\ell)}, k_{\max})$.

5. Compute
$$\beta_i^{(\ell+1)} = \max\left\{-\frac{\sum_t y_t d_t}{\sum_t d_t^2}, \delta\right\}, \text{ where } d_t = \prod_{j \in H_i} x_{jt}^{h_{ij}^{(\ell+1)}} \tag{6.14}$$

6. Let $\boldsymbol{\theta}^{(\ell+1)} = (\alpha_i^{(\ell+1)}, g_i^{(\ell+1)}, \beta_i^{(\ell+1)}, h_i^{(\ell+1)})$ and check the convergence. If converged, output $\boldsymbol{\theta}^{(\ell+1)}$, otherwise, $\ell \leftarrow \ell + 1$ and go to Step 2 .

Definition 1. *$F(h, h')$ is an auxiliary function of $J(h)$ if there exists $\sigma(h')$, such that, if $|h - h'| \le \sigma(h')$*

$$F(h, h') \ge J(h) \quad \textit{and} \quad F(h, h) = J(h) \tag{6.15}$$

are satisfied.

The auxiliary function is useful because of the following lemma.

Lemma 3. *If F is an auxiliary function, then J is nonincreasing under the update*

$$h^{(\ell+1)} = \underset{h \in \{h : |h - h^{(\ell)}| \le \sigma(h^{(\ell)})\}}{\arg\min} F(h, h^{(\ell)}). \tag{6.16}$$

Proof.

$$J(h^{(\ell+1)}) \le F(h^{(\ell+1)}, h^{(\ell)}) \le F(h^{(\ell)}, h^{(\ell)}) = J(h^{(\ell)})$$

□

Lemma 4 shows a property of the exponential function, which is useful for the following discussions.

Lemma 4. *For some point h', the exponential function e^h satisfies*

$$e^h \leq e^{h'} + e^{h'}(h-h') + e^{h'}(h-h')^2\Delta(\sigma), \quad \text{if } |h-h'| \leq \sigma \tag{6.17}$$

where

$$\Delta(\sigma) = \frac{1}{\sigma^2}(e^\sigma - 1 - \sigma). \tag{6.18}$$

Proof.

$$\begin{aligned}
e^h &= e^{h'}e^{h-h'} \\
&= e^{h'}\left(1 + (h-h') + \frac{1}{2!}(h-h')^2 + \cdots + \frac{1}{n!}(h-h')^n + \cdots\right) \\
&= e^{h'} + e^{h'}(h-h') + e^{h'}(h-h')^2\left(\frac{1}{2!} + \cdots + \frac{1}{n!}(h-h')^{n-2} + \cdots\right) \\
&\leq e^{h'} + e^{h'}(h-h') + e^{h'}(h-h')^2\left(\frac{1}{2!} + \cdots + \frac{1}{n!}\sigma^{n-2} + \cdots\right) \\
&= e^{h'} + e^{h'}(h-h') + e^{h'}(h-h')^2\frac{1}{\sigma^2}\left(e^\sigma - 1 - \sigma\right).
\end{aligned}$$

□

Then, the descent property of algorithm 1 is shown in the following theorem.

Theorem 6.1. *The updating rules* (6.6)–(6.9) *in algorithm 1 keep the objective function in problem* (6.4) *nonincreasing.*

Proof. First, we show that the updating rules (6.6) and (6.8) can be obtained by minimizing constructed auxiliary functions. Here, we only prove the rule (6.8) and the rule (6.6) can be proved similarly.

Suppose in one iteration, we need to update the parameter $h_{ik}^{(\ell)}$ to $h_{ik}^{(\ell+1)}$ with all the other parameters fixed. Using the notations in Algorithm 1, the objective function is

$$J_i(h_{ik}) = \frac{1}{2}\sum_t \left(y_t + b_t x_{kt}^{h_{ik}}\right)^2. \tag{6.19}$$

Because other parameters except h_{ik} are fixed, J_i can be considered only depends on h_{ik}. The following auxiliary function is proposed

$$F(h_{ik}, h_{ik}^{(\ell)}) = J_i(h_{ik}^{(\ell)}) + J_i'(h_{ik}^{(\ell)})(h_{ik} - h_{ik}^{(\ell)}) + \tau_h(\sigma_h^{(\ell)})(h_{ik} - h_{ik}^{(\ell)})^2. \tag{6.20}$$

The definitions of $\tau_h(\cdot)$ and $\sigma_h^{(\ell)}$ can be found in Algorithm 1.

Obviously, $F(h_{ik}, h_{ik}) = J_i(h_{ik})$. We only need to verify the first condition in Definition 1. Using Lemma

4, we have, if $|h_{ik} - h_{ik}^{(\ell)}| \le \sigma_h^{(\ell)}$

$$\begin{aligned}
J_i(h_{ik}) &= \frac{1}{2}\sum_t y_t^2 + \sum_t b_t y_t e^{h_{ik}\log x_{kt}} + \frac{1}{2}\sum_t b_t^2 e^{2h_{ik}\log x_{kt}} \\
&\le \frac{1}{2}\sum_t y_t^2 + \sum_t b_t y_t e^{h_{ik}^{(\ell)}\log x_{kt}} + \frac{1}{2}\sum_t b_t^2 e^{2h_{ik}^{(\ell)}\log x_{kt}} \\
&+ \sum_t b_t y_t e^{h_{ik}^{(\ell)}\log x_{kt}}(\log x_{kt})(h_{ik} - h_{ik}^{(\ell)}) + \sum_t b_t^2 e^{2h_{ik}^{(\ell)}\log x_{kt}}(\log x_{kt})(h_{ik} - h_{ik}^{(\ell)}) \\
&+ \sum_t b_t |y_t| e^{h_{ik}^{(\ell)}\log x_{kt}}(\log^2 x_{kt})(h_{ik} - h_{ik}^{(\ell)})^2 \Delta(\sigma_h^{(\ell)}|\log x_{kt}|) \\
&+ 2\sum_t b_t^2 e^{2h_{ik}^{(\ell)}\log x_{kt}}(\log^2 x_{kt})(h_{ik} - h_{ik}^{(\ell)})^2 \Delta(2\sigma_h^{(\ell)}|\log x_{kt}|) \\
&= J_i(h_{ik}^{(\ell)}) + J_i'(h_{ik}^{(\ell)})(h - h_{ik}^{(\ell)}) + \tau_h(\sigma_h^{(\ell)})(h_{ik} - h_{ik}^{(\ell)})^2,
\end{aligned}$$

where the function $\Delta(\cdot)$ is defined in equation (6.18). Therefore, F defined in (6.20) is an auxiliary function. By Lemma 3, the minimization of F with respect to h_{ik} in the neighbourhood of $h_{ik}^{(\ell)}$ makes the objective function J keep nonincreasing. Thus, the updating rule is obtained by solving

$$\underset{h_{ik}\in\{h_{ik}:|h_{ik}-h_{ik}^{(\ell)}|\le\sigma_h^{(\ell)})\}}{\arg\min} F(h_{ik}, h_{ik}^{(\ell)}). \tag{6.21}$$

Note that the auxiliary function (6.20) is quadratic and its global minimum is attained at

$$h_{ik}^* = h_{ik}^{(\ell)} - \frac{1}{2\tau_h(\sigma_h^{(\ell)})} J_i'(h_{ik}^{(\ell)}), \tag{6.22}$$

Also note that

$$|h_{ik}^* - h_{ik}^{(\ell)}| = \frac{|J_i'(h_{ik}^{(\ell)})|}{2\tau_h(\sigma_h^{(\ell)})} \le \frac{|J_i'(h_{ik}^{(\ell)})|}{\sum_t b_t|y_t|x_{kt}^{h_{ik}^{(\ell)}}\log^2 x_{kt} + 2\sum_t b_t^2 x_{kt}^{2h_{ik}^{(\ell)}}\log^2 x_{kt}} = \sigma_h^{(\ell)}, \tag{6.23}$$

the inequality is due to that $\Delta(\sigma) \ge \frac{1}{2}$ for all $\sigma > 0$. Therefore, the updating rule is (6.22) or (6.8) and the objective function keeps nonincreasing during the iterations.

The updating steps (6.7) and (6.9) are obtained by simply solving an univariate least squares problem with a simple constraint, whose objective function is quadratic and thus has an analytical solution. The derivation is simple and we omit it here. Therefore, the objective function keeps nonincreasing in every iterations of Algorithm 1. □

For Algorithm 2, we have the same descent property.

Theorem 6.2. *The updating rules* (6.11)–(6.14) *in Algorithm 2 keep the objective function in problem* (6.10) *nonincreasing.*

Proof. Similar to Theorem 6.1, considering the updating of $h_{ik}^{(\ell)}$, the auxiliary function is defined as (6.20).

The updating rule is obtained by solving

$$\begin{aligned} \underset{h_{ik}}{\text{argmin}} \quad & F(h_{ik}, h_{ik}^{(\ell)}) \\ \text{subject to} \quad & h_{ik}^{(\ell)} - \sigma_h^{(\ell)} \le h_{ik} \le h_{ik}^{(\ell)} + \sigma_h^{(\ell)} \\ & k_{\min} \le h_{ik} \le k_{\max}. \end{aligned} \tag{6.24}$$

Since F is quadratic in h_{ik} and the constraints are very simple, analytical solution (6.13) exists. The remaining proof is similar to Theorem 6.1 and we omit it here. □

6.3 Simulation Examples

To study the performances of the proposed methods, we apply the algorithms to simulated data and compare the estimated parameters with the corresponding true ones.

6.3.1 Performances of Algorithm 1

4-dimensional Model

Consider the following S-system of 4 molecular species [60]:

$$\begin{aligned} \dot{X}_1 &= 12X_3^{-0.8} - 10X_1^{0.5}, \\ \dot{X}_2 &= 8X_1^{0.5} - 3X_2^{0.75}, \\ \dot{X}_3 &= 3X_2^{0.75} - 5X_3^{0.5}X_4^{0.2}, \\ \dot{X}_4 &= 2X_1^{0.5} - 6X_4^{0.8}. \end{aligned} \tag{6.25}$$

The noise-free time series data are obtained by numerically solving the S-system with an initial condition $X(0) = [x_{10}, x_{20}, x_{30}, x_{40}]^T$. The data are sampled at time points in the interval $[0, 5]$ with $\Delta t = 0.1$.

In this example, the data are generated with $X(0) = [10, 1, 2, 3]^T$. The time series data are shown in Figure 6.1, from which we can see all states of X_i's are eventually in the steady states. Algorithm 1 is applied to estimating the parameters from these data with initial values for all parameters chosen by

$$\theta^{\text{initial}} = \theta^{\text{true}}(1 + s\epsilon), \quad \text{for any } \theta \in \{\alpha_i, g_i, \beta_i, h_i\}, \tag{6.26}$$

where ϵ is a standard Gaussian random variable and s is a positive constant. Since (6.25) is a nonlinear optimization problem, to avoid falling into the local optimum, we apply Algorithm 1 for 100 times initiated with different values and choose the one with minimum objective value as the final solution. Here, we set $s = 90\%$. The objective values J_i of the first 100 iterations in one run are illustrated in Figure 6.2. It can be seen that the objective values decrease with the increase of iteration steps. Table 6.1 shows the estimated

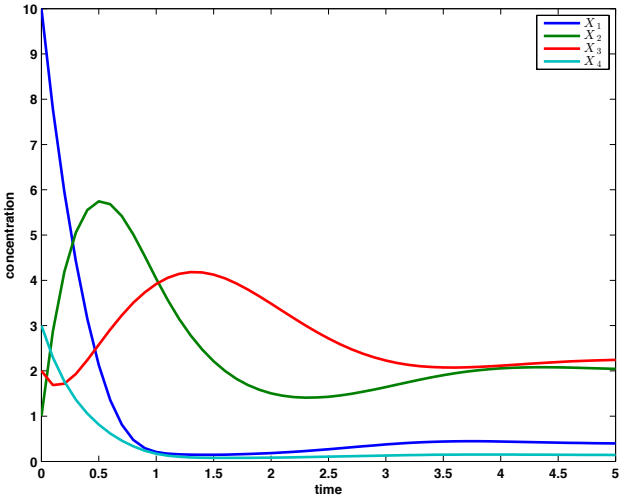


Figure 6.1: Time series data of the 4-dimensional model.

Table 6.1: Estimated results of the 4-dimensional model from Algorithm 1.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	12	11.9856	0.12%	6.809×10^{-4}
β_1	10	9.9819	0.18%	
g_{13}	-0.8	-0.8022	0.27%	
h_{11}	0.5	0.5013	0.25%	
α_2	8	7.9857	0.18%	3.650×10^{-4}
β_2	3	2.9912	0.29%	
g_{21}	0.5	0.5005	0.11%	
h_{22}	0.75	0.7511	0.14%	
α_3	3	3.0377	1.26%	5.8635×10^{-5}
β_3	5	5.0375	0.75%	
g_{32}	0.75	0.7444	0.74%	
h_{33}	0.5	0.4957	0.86%	
h_{34}	0.2	0.1978	1.11%	
α_4	2	1.9965	0.17%	4.0889×10^{-5}
β_4	6	5.9977	0.04%	
g_{41}	0.5	0.5014	0.29%	
h_{44}	0.8	0.8014	0.18%	

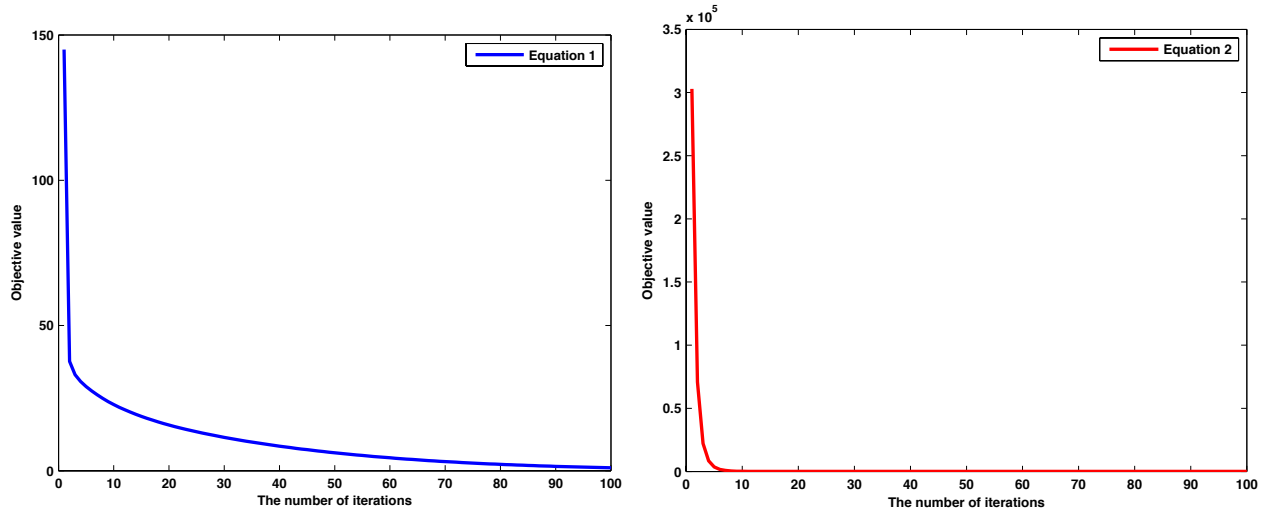


Figure 6.2: Objective values over various numbers of iterations in Algorithm 1.

results, from which we can see that the estimated values are quite close to their true values and the optimal objective values are all very small.

5-dimensional model

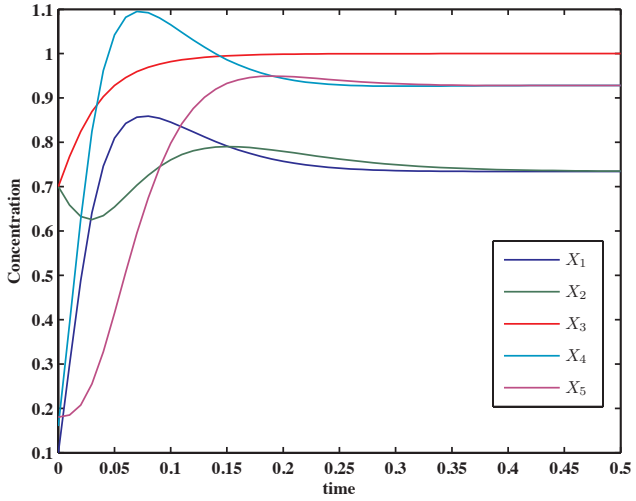


Figure 6.3: Time series data of the 5-dimensional model.

A benchmark 5-dimensional model [1, 65] is considered,

$$\begin{aligned}
\dot{X}_1 &= 5X_3X_5^{-1} - 10X_1^2, \\
\dot{X}_2 &= 10X_1^2 - 10X_2^2, \\
\dot{X}_3 &= 10X_2^{-1} - 10X_2^{-1}X_3^2, \\
\dot{X}_4 &= 8X_3^2X_5^{-1} - 10X_4^2, \\
\dot{X}_5 &= 10X_4^2 - 10X_5^2.
\end{aligned} \tag{6.27}$$

The data used in this example are generated with the initial condition $X(0) = [0.1, 0.7, 0.7, 0.16, 0.18]^T$, the same as in [1]. Figure 6.3 shows the time series data that are sampled in the interval $[0, 0.5]$ with $\Delta t = 0.01$. Note that the states of all variables quickly converge to the steady state. Hence, only limited information on the dynamics of the system is contained in the data. We run Algorithm 1 for 100 times with different initial values obtained from (6.26) with $s = 80\%$ and select the one with minimum objective value as the solution. The results in Table 6.2 show the effectiveness of this algorithm: estimated values of parameters are close to the true values and the optimal objective values are all very small.

Table 6.2: Estimated results of the 5-dimensional model from Algorithm 1.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	5	4.7238	5.52%	3.7096×10^{-4}
β_1	10	9.7188	2.81%	
g_{13}	1	1.1732	17.32%	
g_{15}	-1	-1.0500	4.99%	
h_{11}	2	2.0783	3.92%	
α_2	10	10.0280	0.28%	3.8752×10^{-4}
β_2	10	9.9796	0.20%	
g_{21}	2	1.9953	0.24%	
h_{22}	2	1.9778	1.11%	
α_3	10	9.0097	9.90%	4.2896×10^{-4}
β_3	10	9.0220	9.78%	
g_{32}	-1	-1.1133	11.33%	
h_{32}	-1	-1.1082	10.82%	
h_{33}	2	2.1220	6.10%	
α_4	8	7.5761	5.30%	6.5891×10^{-4}
β_4	10	9.5605	4.40%	
g_{43}	2	2.1641	8.21%	
g_{45}	-1	-1.0457	4.57%	
h_{44}	2	2.0790	3.95%	
α_5	10	9.9333	0.67%	6.6801×10^{-4}
β_5	10	9.9402	0.60%	
g_{54}	2	2.0138	0.69%	
h_{55}	2	2.0240	1.20%	

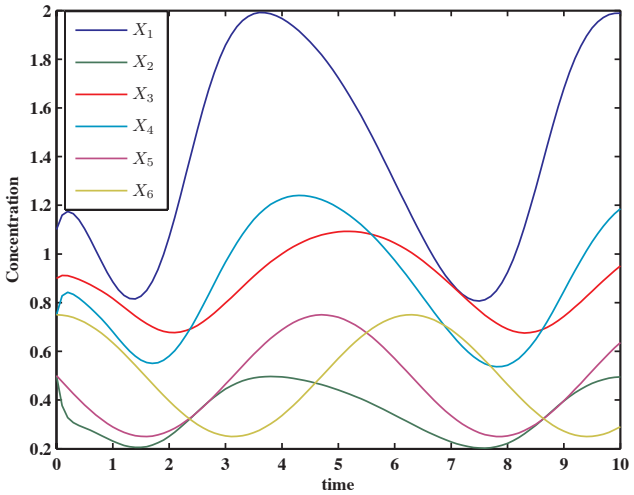


Figure 6.4: Time series data of the 6-dimensional model.

Table 6.3: Estimated results of the 6-dimensional model from Algorithm 1.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	10	9.7145	2.85%	4.2743×10^{-4}
β_1	5	4.8192	3.61%	
g_{13}	-2	-2.0397	1.99%	
g_{15}	1	1.0146	1.46%	
h_{11}	0.5	0.5047	0.94%	
α_2	5	4.7858	4.28%	3.4432×10^{-4}
β_2	10	9.9353	0.65%	
g_{21}	0.5	0.5266	5.32%	
h_{22}	0.5	0.5269	5.39%	
α_3	2	2.0059	0.29%	4.1841×10^{-6}
β_3	1.25	1.2580	0.64%	
g_{32}	0.5	0.4963	0.74%	
h_{33}	0.5	0.4964	0.72%	
α_4	8	8.0231	0.29%	1.2605×10^{-4}
β_4	5	5.0138	0.28%	
g_{42}	0.5	0.5002	0.04%	
h_{44}	0.5	0.5002	0.05%	
α_5	0.5	0.4962	0.76%	1.9439×10^{-6}
β_5	1	0.9978	0.22%	
h_{56}	1	1.0086	0.86%	
α_6	1	0.9978	0.22%	1.9949×10^{-6}
β_6	0.5	0.4962	0.76%	
g_{65}	1	1.0087	0.87%	

6-dimensional model

In this example, Algorithm 1 is applied to estimate the parameters in the following 6-dimensional S-system [1]:

$$\begin{aligned}
\dot{X}_1 &= 10X_3^{-2}X_5 - 5X_1^{0.5}, \\
\dot{X}_2 &= 5X_1^{0.5} - 10X_2^{0.5}, \\
\dot{X}_3 &= 2X_2^{0.5} - 1.25X_3^{0.5}, \\
\dot{X}_4 &= 8X_2^{0.5} - 5X_4^{0.5}, \\
\dot{X}_5 &= 0.5 - X_6, \\
\dot{X}_6 &= X_5 - 0.5.
\end{aligned} \tag{6.28}$$

The noise-free time series data are generated with the initial condition $X(0) = [1.1, 0.5, 0.9, 0.75, 0.5, 0.75]^T$ which matches that in [1]. Figure 6.4 illustrates the time series data which are sampled in the interval $[0, 10]$ with $\Delta t = 0.1$. Figure 6.4 also shows the periodic oscillating behaviour of the data. The initial values are also chosen by (6.26) with $s = 50\%$. The solution shown in Table 6.3 is the best one among 100 runs of Algorithm 1 with different initial values. The results in Table 6.3 indicate that the estimated parameters are close to their true values and the optimal objective values are all very small.

6.3.2 Performances of Algorithm 2

Table 6.4: Estimated results of the 4-dimensional model from Algorithm 2.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	12	11.945	0.4585%	8.2968×10^{-4}
β_1	10	9.9288	0.7119%	
g_{13}	-0.8	-0.807	0.8789%	
h_{11}	0.5	0.5034	0.6858%	
α_2	8	7.9857	0.1787%	3.65×10^{-4}
β_2	3	2.9912	0.2931%	
g_{21}	0.5	0.5005	0.1093%	
h_{22}	0.75	0.7511	0.1411%	
α_3	3	3.0337	1.1233%	7.9833×10^{-5}
β_3	5	5.0244	0.4879%	
g_{32}	0.75	0.7435	0.8691%	
h_{33}	0.5	0.4937	1.26%	
h_{34}	0.2	0.1964	1.7893%	
α_4	2	2.0051	0.2528%	1.991×10^{-4}
β_4	6	5.9847	0.2551%	
g_{41}	0.5	0.4927	1.457%	
h_{44}	0.8	0.7938	0.7776%	

The performances of Algorithm 2 is studied in this example. We apply the algorithm to the data generated from the 4-dimensional, 5-dimensional and 6-dimensional models, respectively. The time series data we use are exactly the same as above-mentioned examples. Since Algorithm 2 solves the problem (6.10) which includes a range constraint for each parameter, the initial values for the algorithm are randomly generated in those ranges. In this study, the rate constants are restricted in the range $[0.1, 10]$ and kinetic orders are in the range $[-2, 3]$. For each model, we run the algorithm for 100 times with different initial values and choose the one with minimum objective value as the final result. The estimated results from algorithm 2 for each model are reported in the Tables 6.4–6.6. From the results, it can be seen that estimated parameters for the 4-dimensional and 6-dimensional models have very small relative errors compared with their corresponding true values. For the 5-dimensional model, most parameters are estimated with low errors, while the estimations of the parameters in the 3rd ODE have relatively large errors. This may due to the limited information contained in the time series data. We can also see that, the estimated objective function values for these 3 models are all very small. All these results show the effectiveness of the proposed algorithm.

Table 6.5: Estimated results of the 5-dimensional model from Algorithm 2.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	5	4.7704	4.5918%	3.1738×10^{-4}
β_1	10	9.7686	2.3138%	
g_{13}	1	1.1535	15.3495%	
g_{15}	-1	-1.0424	4.2402%	
h_{11}	2	2.0652	3.2612%	
α_2	10	10.0133	0.1333%	3.9712×10^{-4}
β_2	10	9.9644	0.3563%	
g_{21}	2	2.0022	0.1122%	
h_{22}	2	1.9845	0.7763%	
α_3	10	8.1857	18.1435%	9.5442×10^{-4}
β_3	10	8.1839	18.1609%	
g_{32}	-1	-1.2361	23.6145%	
h_{32}	-1	-1.2363	23.6264%	
h_{33}	2	2.2322	11.6124%	
α_4	8	7.5565	5.5442%	6.7221×10^{-4}
β_4	10	9.5398	4.6023%	
g_{43}	2	2.1688	8.4413%	
g_{45}	-1	-1.0476	4.7607%	
h_{44}	2	2.0827	4.1342%	
α_5	10	9.9649	0.3505%	5.6977×10^{-4}
β_5	10	9.958	0.4204%	
g_{54}	2	2.0152	0.7598%	
h_{55}	2	2.0047	0.2345%	

Table 6.6: Estimated results of the 6-dimensional model from Algorithm 2.

Parameter	True Value	Estimation	Relative Error	Objective Value
α_1	10	9.7512	2.4881%	4.3511×10^{-4}
β_1	5	4.9344	1.3118%	
g_{13}	-2	-1.9891	0.5461%	
g_{15}	1	0.9872	1.2845%	
h_{11}	0.5	0.4899	2.0172%	
α_2	5	4.6623	6.7549%	3.3703×10^{-4}
β_2	10	9.6115	3.8853%	
g_{21}	0.5	0.5221	4.4167%	
h_{22}	0.5	0.5218	4.3684%	
α_3	2	1.9983	0.084%	4.0053×10^{-6}
β_3	1.25	1.2484	0.1248%	
g_{32}	0.5	0.5005	0.0925%	
h_{33}	0.5	0.5005	0.0912%	
α_4	8	7.9848	0.1896%	1.2578×10^{-4}
β_4	5	4.9926	0.1487%	
g_{42}	0.5	0.4996	0.0864%	
h_{44}	0.5	0.4995	0.0953%	
α_5	0.5	0.4991	0.1864%	1.2022×10^{-7}
β_5	1	0.9994	0.0609%	
h_{56}	1	1.002	0.1989%	
α_6	1	0.9977	0.2326%	2.2004×10^{-6}
β_6	0.5	0.496	0.8035%	
g_{65}	1	1.0091	0.9094%	

6.4 Conclusions

The S-system is an effective mathematical model to characterize and analyze the molecular biological systems. The parameters in S-systems have significant biological meanings. However, the estimation of these parameters from time series biological data is non-trivial, because of the nonlinearity and complexity of this model. An novel method, the auxiliary function guided coordinate descent (AFGCD), is proposed in this study to estimate the parameters of S-systems. To solve the nonlinear optimization problem involved in the parameter estimation, the proposed method optimizes one parameter at a time. Taking advantage of a property of the exponential function, an auxiliary function is constructed for each kinetic order parameter. We prove that updating the parameter value by optimizing the auxiliary function keeps the objective function nonincreasing. As the auxiliary function is quadratic, the analytical solution exists and therefore the updating rule for each parameter is simple and efficient. Based on this idea, two algorithms are developed: one estimates the parameters in S-system with the only non-negative constraints on rate constants; the other puts constraints both on rate constants and kinetic orders. Their performances are studied in simulation examples, which show that the estimated parameter values from the proposed algorithms are very close to the corresponding true values and the optimized objective functions has very low values. All of these results demonstrate the effectivenesses of the proposed algorithms.

In this study, the topology information of the system is assumed to be known. One direction of the future work is to extend the current method with structure identification method to infer the S-system without knowing the system topology.

Acknowledgement

This study is supported by Natural Science and Engineering Research Council of Canada (NSERC).

Chapter 7

Inference of Biological S-system Using Separable Estimation Method and Genetic Algorithm

Published as: L.Z. Liu, F.X. Wu, and W.J. Zhang, "Inference of biological s-system using the separable estimation method and the genetic algorithm," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 9, no. 4, pp. 955–965, 2012 [64]. This work is an extension to our conference paper: L.Z. Liu, F.X. Wu, L.L. Han, and W.J. Zhang, "Structure identification and parameter estimation of biological S-systems," in *Bioinformatics and Biomedicine (BIBM), 2010 IEEE International Conference on*, pp. 329 334, Dec. 2010 [147].

Chapters 5 and 6 focus on estimating the parameters in S-systems from time-course data when the system structure is known. In practice, the system structure is not always available and the main task is to infer the system structure from the data.

In this chapter, we assume the system structure is unknown and aim at identifying the structure and estimating the parameters of S-system from time-course data alone. By considering the sparseness of the network and special mathematical form of the model, a method called pruning separable parameter estimation algorithm (PSPEA) is developed to locally infer the S-system from time-course data. PSPEA is then combined with a continuous genetic algorithm (CGA) to form a hybrid algorithm which can globally identify the structure and estimate the parameters of S-system from time-course data. Chapters 5, 6 and 7 accomplish Objective 3 of this thesis.

Abstract

Reconstruction of a biological system from its experimental time series data is a challenging task in systems biology. The S-system which consists of a group of nonlinear ordinary differential equations is an effective model to characterize molecular biological systems and analyze the system dynamics. However, inference of S-systems without the knowledge of system structure is not a trivial task due to its nonlinearity and complexity. In this paper, a pruning separable parameter estimation algorithm is proposed for inferring S-systems. This novel algorithm combines the separable parameter estimation method and a pruning strategy, which includes adding an ℓ_1 regularization term to the objective function and pruning the solution with a threshold value. Then, this algorithm is combined with the continuous genetic algorithm to form a hybrid algorithm who owns

the properties of these two combined algorithms. The performance of the pruning strategy in the proposed algorithm is evaluated from two aspects: the parameter estimation error and structure identification accuracy. The results show that the proposed algorithm with the pruning strategy has much lower estimation error and much higher identification accuracy than the existing method.

7.1 Introduction

Biological systems, such as gene regulation networks, metabolic pathways, and signal transduction cascades are all complex systems with many interacting components. The reconstruction of structure and mechanisms of interactions among components in biological systems from available experimental data is one of the most challenging tasks of systems biology. Recent advances in biological technologies such as DNA microarrays make more and more biological data available. These data are important information sources for understanding the biological systems or processes. The information contained in the data can be retrieved with the parameter estimation methods when the system structure is known. The system structure can also be extracted from the data with system structure identification methods. However, due to the complexity of biological systems, both the parameter estimation and the structure identification are not trivial tasks.

The reconstruction or inference of biological systems from experimental data is refer to as a reverse engineering problem. It encompasses the choice of the model, fitting of the structures and parameters of the model into the available data, and inference of the underlying principles of the biological systems. Many mathematical models and algorithms have been proposed to describe and infer biological systems. Some of the typical methods, especially for the inference of genetic regulatory networks, include the generalized Bayesian networks [41, 92], boolean networks [34, 35, 101, 38] and linear and nonlinear ordinary differential equations (ODEs) [2, 20, 153]. Besides, many new methods are developed recently. A recurrent neural network (RNN) is proposed to model the genetic regulatory networks in [102]. The network is inferred from gene expression time series by a particle swarm optimization (PSO) based search algorithm. The interactions among genes are explained through a connection weight matrix. Tenenhaus et al. [14] considers modeling the genetic regulatory network as Gaussian Graphical Model (GGM), in which the conditional independence between any two genes is measured by the partial correlation which is estimated by partial least squares (PLS) regression method. Ram and Chetty [154] use the Markov blanket method to model the genetic regulatory networks. The relationships among genes are inferred from time series data with the aid of a novel genetic algorithm. In [22], the role of soft computing methodologies, such as fuzzy sets, evolutionary strategies and neurocomputing, and their hybridizations for reconstruction of the genetic network are well surveyed.

Most of models for molecular biological systems are nonlinear and are required to be general enough to capture all the essence of observed experimental data. Thus, the structure identification and parameter estimation in these nonlinear models always turn out to be nonlinear optimization problems [155, 156, 157].

In this paper, the parameter estimation for one of the most popular models which is referred to as the

S-system model is studied. The S-system model in Biochemical System Theory (BST) is considered as an effective mathematical framework and a consistent model to represent and analyze the biological systems [59]. It is a type of power-law formalism and a particular set of nonlinear ordinary differential equations. Each parameter in an S-system model has its own unique specific meaning and role in the system. The dynamical and structural characteristics of an S-system can be mapped onto its parameters. Therefore, the identification of system structure has been reduced to a parameter estimation task which is a major advantage of its representation. However, due to the complexity of S-systems, it is not easy to estimate the parameters from experimental time series data. This reverse engineering problem is a large-scale and time-consuming parameter optimization problem.

Several evolutionary computation techniques have been proposed to obtain the global optimization of parameters in S-systems. Kikuchi et al. [61] used a real coded genetic algorithm to optimize the parameters in S-systems, whose objective has a penalty term to prune unnecessary connections in investigated gene networks. A simple crossover method was incorporated into the genetic algorithm and a gradual optimization strategy was introduced in the estimation procedure. Daisuke and Horton [158] introduced a distributed genetic algorithm with the scale-free restriction. Kimura et al.[159] used a genetic local search with a distance independent diversity control method to estimate the parameters based on the decomposition strategy. One drawback of this decomposition strategy is that there is no information exchange between the sub-problems. Kimura et al. [62] proposed a cooperative coevolutionary algorithm to solve the decomposed subtasks simultaneously. The values calculated in each sub-problem are transferred to other sub-problems, that is, the sub-problems interact with each other. A two-stage evolutionary algorithm (iTEA) was proposed by Ho et al. [63]. The first stage applied a novel intelligent genetic algorithm with intelligent crossover based on an orthogonal experimental design (OED) to solve each sub-problem. The solutions from the first stage are then combined as an initial solution for the OED-based simulated annealing algorithm at the second stage. Tsai and Wang [160] used the hybrid differential evolution method to get satisfied solution and a gradient-based optimization method is used to refine the solution. An S-tree along with genetic programming has been introduced by Cho et al. [161] to identify the network structure and find parameter values without adding a penalty term to the objective function.

The computation cost for methods based on stochastic search algorithms is very expensive. Besides the evolutionary computation, there are some other methods proposed in literatures. Alternating regression [141] estimates the parameters by using iterative linear regression based on decoupling of S-systems. This method is very fast. However, sometimes, it doesn't converge. Vilela et al. [65] proposed a novel parameter estimation method which is based on eigenvector optimization of a matrix formed from multiple regression equations of the linearized decoupled S-systems. Tucker and Moulton [162] used an interval analysis technique to estimate parameters.

In this paper, algorithms to infer the S-system without any structure information are proposed. The proposed algorithms are extensions of the method proposed by Wu and Mu [146], which take advantage of

the special structure of the S-system, that is, one group of parameters is linear and the other is nonlinear, such that the parameter searching space and computation time is reduced. However, the separable parameter estimation method can only be applied with the knowledge of structure information. Without knowing the system structure, all the parameters in S-system are free and the algorithm has to search the full model. Since the number of parameters become very large even for an intermediate-scale network, the estimation task turns out to be very complex. To reduce the complexity of the problem, a prior knowledge that biological network is sparse is incorporated into the algorithm by implementing a pruning strategy which includes adding an ℓ_1 regularization term to the original objective function and using a threshold to prune the solutions in the optimization process. To improve the robustness to the initial values and searching abilities, the proposed algorithm is further combined with the continuous genetic algorithm.

Briefly, the remainder of the paper is organized as follows. In Section 7.2, the S-system and its parameter estimation problem are described. In Section 7.3, the pruning separable parameter estimation algorithm is introduced. In Section 7.4, A hybrid algorithm is developed by combing the pruning separable parameter estimation algorithm with the continuous genetic algorithm. In Section 7.5, the performance of pruning strategy used in our proposed algorithms is discussed. Finally, Section 7.6 concludes this study and points out some directions of future work.

7.2 Problem Statement

The S-system model [60] is a type of power-law formalism, based on a particular set of nonlinear differential equations, which is described as follows:

$$\dot{X}_i = \alpha_i \prod_{j=1}^{N} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_j^{h_{ij}}, \tag{7.1}$$

where $i = 1, 2, \ldots, N$, X_i is the state variable and represents the concentration of reaction component i and N is the number of components in the system. For each ODE, the right hand side is composed of two terms. The first term represents the production of reactants and the other term corresponds to the degradation. The change of state variable over the time is described by the difference between these two terms. α_i and β_i are nonnegative rate constants, and g_{ij} and h_{ij} are real-valued kinetic orders that reflect the interaction intensity of X_j to X_i. If $g_{ij} > 0$, X_j activates the production of X_i while if $g_{ij} < 0$, X_j inhibits the production of X_i. h_{ij} has the same effects but on degradation. A zero-value kinetic order indicates X_j has no such effect on X_i. In this way, the system structure is mapped onto the kinetic order parameters in S-systems. Therefore, the system structure can be determined by estimating the parameters.

Given the observed experimental time series data, x_{i,t_k}, $k = 1, 2, \ldots, n$, $i = 1, 2, \ldots, N$ which represent the concentration of component i at time point t_k, the parameter estimation problem is in fact to search for particular values for the parameters such that the solutions from the model with these parameter values

can fit well with the given observation data in some specified manner. For example, find the parameter values which minimize the least squares errors between observed and estimated data. Therefore, parameter estimation is formulated as an optimization problem.

7.3 Pruning Separable Parameter Estimation Algorithm (PSPEA)

In this section, the separable parameter estimation method (SPEM) [146] is introduced first. Then, a new algorithm named pruning separable parameter estimation algorithm (PSPEA) is proposed to infer the S-system. This algorithm extends the SPEM by applying a pruning strategy. Details are shown below.

7.3.1 Separable Parameter Estimation Method

Suppose the system has N components and for each component X_i, a time series data consisting of n time points, $x_{i,t_1}, x_{i,t_2}, \ldots, x_{i,t_n}$ have been observed. Here, the derivative of X_i at time point t_k in equation (7.1) is substituted by the deduced slop $S_i(t_k)$. Then the system is reformulated as

$$\begin{aligned}
S_i(t_1) &\approx \alpha_i \prod_{j=1}^{N} x_{j,t_1}^{g_{ij}} - \beta_i \prod_{j=1}^{N} x_{j,t_1}^{h_{ij}}, \\
&\vdots \\
S_i(t_k) &\approx \alpha_i \prod_{j=1}^{N} x_{j,t_k}^{g_{ij}} - \beta_i \prod_{j=1}^{N} x_{j,t_k}^{h_{ij}}, \\
&\vdots \\
S_i(t_n) &\approx \alpha_i \prod_{j=1}^{N} x_{j,t_n}^{g_{ij}} - \beta_i \prod_{j=1}^{N} x_{j,t_n}^{h_{ij}},
\end{aligned}$$

where $i = 1, 2, \ldots, N$. Thus, the original N coupled differential equations can be analyzed in the form of $n \times N$ uncoupled algebraic equations [60, 59]. The estimation of the slop is a crucial step in the decoupling procedure. To increase the accuracy of the proposed method, the five-point numerical derivative method is applied in this paper, that is,

$$S_i(t_k) = \frac{-x_{i,t_{k+2}} + 8x_{i,t_{k+1}} - 8x_{i,t_{k-1}} + x_{i,t_{k-2}}}{12\Delta t}, \tag{7.2}$$

where Δt is the sampling interval and $\Delta t = t_{k+1} - t_k$ for all k. Then, for each i, we estimate the parameters by minimizing the following objective function:

$$\begin{aligned}
&J_i(\alpha_i, \beta_i, g_i, h_i) \\
&= \sum_{k=1}^{n} \left[S_i(t_k) - \left(\alpha_i \prod_{j=1}^{N} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_j^{h_{ij}} \right) \right]^2,
\end{aligned} \tag{7.3}$$

where $g_i = [g_{i1}, g_{i2}, \ldots, g_{iN}]^T$ and $h_i = [h_{i1}, h_{i2}, \ldots, h_{iN}]^T$. Let $\Gamma_i = (\alpha_i, \beta_i)^T$, $S_i = [S_i(t_1), S_i(t_2), \ldots, S_i(t_n)]^T$ and

$$\tilde{X}_i = \begin{bmatrix} \prod_{j=1}^{N} x_{j,t_1}^{g_{ij}} & -\prod_{j=1}^{N} x_{j,t_1}^{h_{ij}} \\ \vdots & \vdots \\ \prod_{j=1}^{N} x_{j,t_n}^{g_{ij}} & -\prod_{j=1}^{N} x_{j,t_n}^{h_{ij}} \end{bmatrix}.$$

Then equation (7.3) can be rewritten as

$$J_i(\Gamma_i, g_i, h_i) = \|S_i - \tilde{X}_i \Gamma_i\|_2^2, \tag{7.4}$$

where $\|\cdot\|_2$ is the ℓ_2 norm. Note that $\tilde{X}_i$ is the function of (g_i, h_i) and does not depend on Γ_i, that is, the parameters are in a separable structure. Therefore, the method SPEM [146] can be applied.

For a given value of $(g_i, h_i) = (\bar{g}_i, \bar{h}_i)$, $\tilde{X}_i$ can be considered as constant. Thus, using the least squares method, Γ_i is estimated as follows

$$\hat{\Gamma}_i = (\tilde{X}_i{}^T \tilde{X}_i)^{-1} \tilde{X}_i{}^T S_i. \tag{7.5}$$

Substitute equation (7.5) into (7.4) to obtain

$$\tilde{J}_i(\bar{g}_i, \bar{h}_i) = J_i(\Gamma_i, \bar{g}_i, \bar{h}_i) = S_i^T [I - \tilde{X}_i (\tilde{X}_i^T \tilde{X}_i)^{-1} \tilde{X}_i^T] S_i. \tag{7.6}$$

Note that only kinetic order parameters appear in (7.6). Then, the minimization of (7.4) is transformed into minimization of (7.6), that is, once $(\tilde{g}_i, \tilde{h}_i)$ that minimizes (7.6) is founded, the optimal $\tilde{\Gamma}_i$ can be computed from (7.5) by letting $(g_i, h_i) = (\tilde{g}_i, \tilde{h}_i)$. In this way, both the dimension of parameter searching space and the computation effort can be reduced as discussed in [146].

7.3.2 Proposed Algorithm

The S-system parameter estimation by SPEM with the knowledge of system structure information was studied in [146]. In this paper, we focus on inferring S-systems without structure information. Without knowledge of the structure, the algorithm to optimize (7.6) has to search a full model, that is, all parameters are free and can be any value. When the number of variables N is not small, the number of parameters $2N^2$ in (7.6), becomes very large and the estimation process becomes extremely difficult. To reduce the complexity of the estimation task, a priori knowledge is incorporated here. Recall that if one variable does not affect the production or degradation of another variable, its corresponding kinetic order in the S-system is zero. Since biological networks have been known to be sparse, most of the kinetic orders should be zero.

According to optimization theory [163], ℓ_1 regularization can produce the sparse solution. We add an ℓ_1 regularization term to the objective function (7.6) as follows:

$$\tilde{J}_i^*(g_i, h_i, \lambda) = \tilde{J}_i(g_i, h_i) + \lambda(\|g_i\|_1 + \|h_i\|_1) \tag{7.7}$$

where λ is a nonnegative scalar and $\|\cdot\|_1$ is the ℓ_1 norm, $\|g_i\|_1 = \sum_i^N |g_{ij}|$. The ℓ_1 regularization term in (7.7) is also called a pruning penalty term, whose variants have been used in some literatures [61, 159, 62, 63]. Note that for non-zero λ, the large value kinetic orders receive heavy penalties. Therefore, this term can make zero kinetic orders to be zero. The first term in (7.7) aims at the reproduction of given time series, while the second term, on the other hand, expresses the sparseness of the network. In this paper, we estimate the kinetic order by minimizing (7.7) and get the rate constants from (7.5). In practice, by optimizing function (7.7) zero kinetic orders may not be exactly zero, but are very close to zero. In this study, a small value threshold θ is applied to prune the solutions such that the very small-value parameter to be zero, that is, if $|g_{ij}|$(or $|h_{ij}|$) $\leq \theta$, set g_{ij}(or h_{ij}) $= 0$, otherwise, it does not change. Based on (7.7), the pruning separable parameter estimation algorithm (PSPEA) is proposed as follows:

Step 1. For each time series, use (7.2) to estimate the slop, $S_i(t_k)$, at each time point.

Step 2. Use the Matlab *fmincon* function to optimize objective function (7.7) several times with various initial values. Among all the solutions, pick up the one $(\hat{g}_i, \hat{h}_i)$ that minimizes (7.6) as the elite solution.

Step 3. Use the elite solution from Step 2 as the initial value to optimize (7.7) by Matlab *fminunc* function. Prune the solution with threshold θ.

Step 4. If any parameter, g_{ij} or h_{ij} has been pruned in Step 3, then take the new solution as initial value and go to Step 3, else go to Step 5.

Step 5. Use the solution from Step 4 as initial value to optimize (7.6) and then calculate the rate constants by substituting the solution from Step 4 into (7.5).

In the algorithm above, Steps 2–4 are to identify S-system structure while Step 5 is to estimate parameters in the S-system. The proposed algorithm combines a pruning strategy and the method SPEM. The pruning strategy includes adding an ℓ_1 regularization term as shown in (7.7) and using the pruning threshold θ. The regularization term forces kinetic orders to approach zero and the threshold makes some of them become zero. This strategy is mainly employed in Steps 2–4 which are the crucial steps of the proposed algorithm. The purpose of these steps is to determine the network structure, that is, finding the non-zero kinetic orders and their signs. Then the parameter values are further refined in the following step. To avoid falling into the local minimum, we calculate Step 2 for several times with different initial values and pick up the elite one to be used in the next steps.

7.3.3 Experiments

To evaluate the performance of the PSPEA, it is applied to infer two S-systems from simulated time series data.

Example 1. We first consider the dynamics of pathway shown in Figure 7.1. There are four dependent

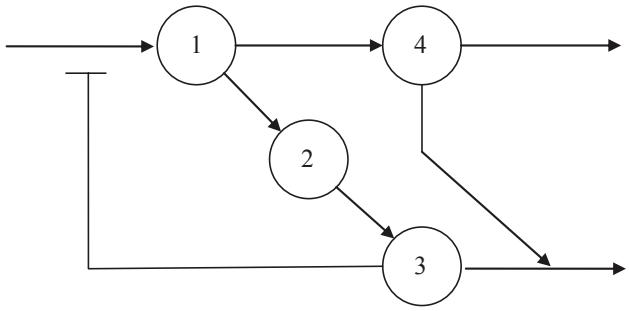


Figure 7.1: A 4-dimensional metabolic pathway branch

variables in the system. The production of X_1 is affected by the inhibition exerted by X_3, which is produced by X_1 via intermediate X_2. X_1 can be used for the production of X_4, which promotes the degradation of X_3. Furthermore, the degradation for each variable is also promoted by itself. The S-system for this metabolic network is described by ordinary differential equations below.

$$\begin{aligned}
\dot{X}_1 &= \alpha_1 X_3^{g_{13}} - \beta_1 X_1^{h_{11}}, \\
\dot{X}_2 &= \alpha_2 X_1^{g_{21}} - \beta_2 X_2^{h_{22}}, \\
\dot{X}_3 &= \alpha_3 X_2^{g_{32}} - \beta_3 X_3^{h_{33}} X_4^{h_{34}}, \\
\dot{X}_4 &= \alpha_4 X_1^{g_{41}} - \beta_4 X_4^{h_{44}}.
\end{aligned} \tag{7.8}$$

The corresponding values of rate constants and kinetic orders are shown in Table 7.1. There are totally 40 parameters: 8 positive rate constants, 9 non-zero kinetic orders and 23 zero kinetic orders.

Table 7.1: Values of parameters in 4-dimensional S-system.

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}
1	12	0	0	-0.8	0	10	0.5	0	0	0
2	8	0.5	0	0	0	3	0	0.75	0	0
3	3	0	0.75	0	0	5	0	0	0.5	0.2
4	2	0.5	0	0	0	6	0	0	0	0.8

In order to investigate the proposed algorithm, a set of noise-free simulated time series data is generated from (7.8) with initial state $x_1 = 10$, $x_2 = 1$, $x_3 = 2$ and $x_4 = 3$ and parameter values in Table 7.1. The trajectories for all state variables are plotted in Figure 7.2. The time t is from 0 to 5 and the sampling interval Δt is 0.1.

The PSPEA is applied to estimate the parameters. Here, the threshold θ is set to be 0.05 and $\lambda = 1$. The optimization of objective (7.7) is formulated as a nonlinear constraint problem. The kinetic orders are searched in the interval $[-1, 1]$. We assume that at most only one of g_{ij} and h_{ij} is non-zero for each i, since it

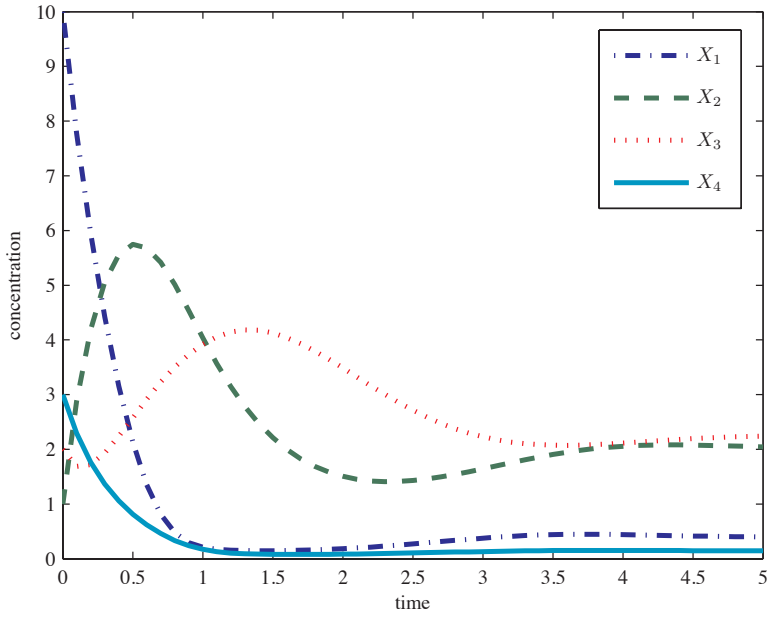


Figure 7.2: Trajectories of the variables in (7.8).

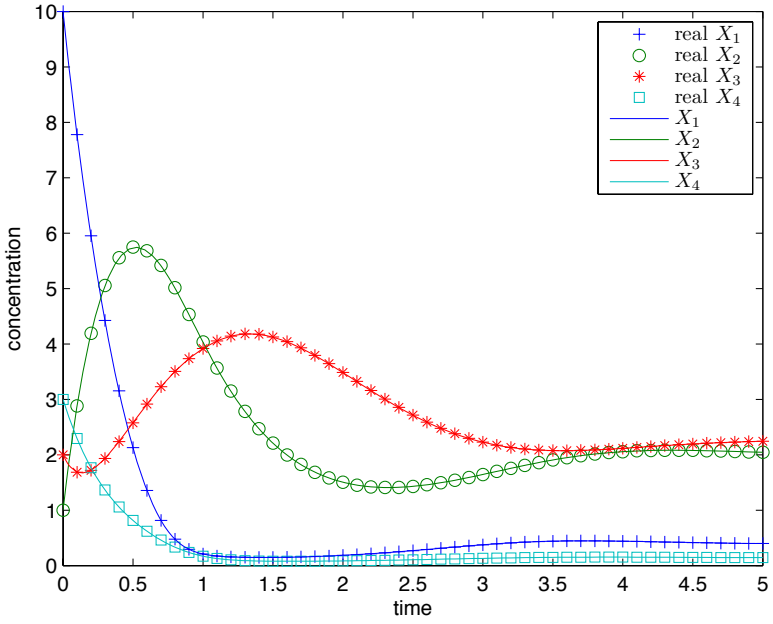


Figure 7.3: Fitness of the integrated system (full lines) found by PSPEA.

is relatively rare that the same variable is present in both the production and degradation terms of another variable [59]. In this example, Step 2 runs for 8 times with different initial guesses generated by the strategy similar as [146]. We employ the true kinetic order value plus a fraction of Gaussian noise as initial guess values, that is, $(g_i^0; h_i^0) = (g_i; h_i) + \sigma\varepsilon$, where ε is a random vector whose element follows a standard normal distribution and σ is a positive constant. Here, σ is chosen as 20%. The estimated results are shown in Table 7.2.

Table 7.2: Results from PSPEA for the 4-dimensional example.

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}
1	11.9723	0	0	-0.8032	0	9.9678	0.5018	0	0	0
2	8.0066	0.4994	0	0	0	3.0059	0	0.7491	0	0
3	3.0184	0	0.7466	0	0	5.0126	0	0	0.4970	0.1981
4	1.9990	0.5012	0	0	0	6.0013	0	0	0	0.8010

As illustrated in Table 7.2, the network structure has been exactly identified, that is, all zero-valued kinetic orders have been recognized and the signs of the other estimated kinetic orders are the same with the corresponding true parameter values. It can also be seen that the non-zero estimated values are very close to the corresponding true values. The final values of optimized objective function (7.6) are smaller than 10^{-2}. The dynamics and fitness of the integrated system found by the proposed algorithm are shown in Figure 7.3. The inferred system (solid line) perfectly fits the simulated data (other lines).

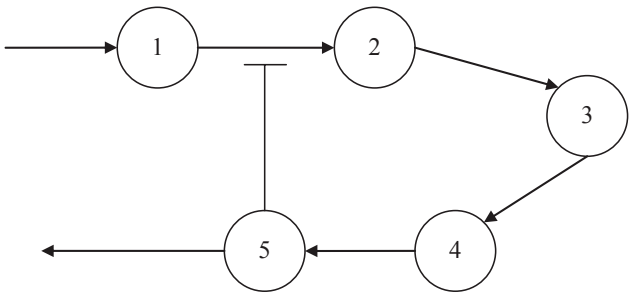


Figure 7.4: A 5-dimensional linear pathway with feedback.

Table 7.3: Values of parameters in 5-dimensional S-system.

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	2	0	0	0	0	0	2	0.5	0	0	0	-1
2	2	0.5	0	0	0	-1	4	0	0.5	0	0	0
3	4	0	0.5	0	0	0	4	0	0	0.8	0	0
4	4	0	0	0.8	0	0	1	0	0	0	1	0
5	1	0	0	0	1	0	4	0	0	0	0	0.5

Example 2. Another 5 dimensional S-system from [60] is inferred by the proposed algorithm in the following. The system to be estimated is a linear pathway with feedback as depicted in Figure 7.4. In this system, there are 5 dependent variables which are represented by an S-system (7.9) with the associated parameter values shown in Table 7.3.

$$\begin{aligned}
\dot{X}_1 &= \alpha_1 - \beta_1 X_1^{h_{11}} X_5^{h_{15}}, \\
\dot{X}_2 &= \alpha_2 X_1^{g_{21}} X_5^{g_{25}} - \beta_2 X_2^{h_{22}}, \\
\dot{X}_3 &= \alpha_3 X_2^{g_{32}} - \beta_3 X_3^{h_{33}}, \\
\dot{X}_4 &= \alpha_4 X_3^{g_{43}} - \beta_4 X_4^{h_{44}}, \\
\dot{X}_5 &= \alpha_5 X_4^{g_{54}} - \beta_5 X_5^{h_{55}}.
\end{aligned} \tag{7.9}$$

There are totally 60 parameters consisting of 10 positive rate constants, 11 non-zero kinetic orders and 39 zero-valued kinetic orders. A set of simulated noise-free time series data generated from the S-system with the initial state $x_1 = 10$, $x_2 = 1$, $x_3 = 2$, $x_4 = 3$, and $x_5 = 4$ is used to estimate the system. The time t starts from 0 to 10 and time interval Δt is 0.1. As in the first example, we set $\lambda = 1$, $\theta = 0.05$ and $\sigma = 20\%$. To avoid falling into the local minimum, for this example, Step 2 was run with 30 different initial values. The results from the proposed algorithm are shown in Table 7.4. Comparing Table 7.3 and Table 7.4, similar conclusions can be drawn as the 4-dimensional example. The network structure has been exactly identified and parameter values are very close to their true values only except α_2. The values of the optimized objective function (7.6) are again all less than 10^{-2}.

Table 7.4: Results from PSPEA for 5-dimensional example.

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	2.0014	0	0	0	0	0	2.0008	0.5000	0	0	0	-1.0000
2	1.0142	0.9949	0	0	0	-0.9958	4.0271	0	0.4975	0	0	0
3	3.9986	0	0.4997	0	0	0	3.9986	0	0	0.7996	0	0
4	4.0005	0	0	0.8000	0	0	1.0001	0	0	0	1.0000	0
5	1.0024	0	0	0	0.9990	0	4.0042	0	0	0	0	0.4996

7.4 PSPEA with Genetic Algorithm

The crucial parts of PSPEA are Steps 2–4, because these steps are responsible for the structure identification which is the main concern of this paper. In these steps, the objective function (7.7) is optimized by Matlab function *fmincon*. Due to the nonlinearity of (7.7), the optimization is likely to fall into the local minimum. To overcome this problem, in PSPEA, various different initial values are used to optimize (7.7). However, using different initial values can only alleviate this problem to some extent. A strategy which can help the algorithm search the whole parameter space are needed. In this section, we propose to combine the genetic

algorithm and PSPEA to infer the biological S-system.

7.4.1 Continuous Genetic Algorithm

Since the kinetic orders in S-system are real numbers, it is naturel to consider using the genetic algorithm which encodes the variables by floating-point numbers. In this paper, we employ the continuous genetic algorithm (CGA) in [164]. Compared with binary coded genetic algorithm, CGA requires less storage and runs faster in that the variable is represented by a single floating-point number instead of many integers and do not need to be decoded prior to the evaluation of the cost function. In addition, CGA represents the continuous variables more precisely than the binary GA. Given an objective function, the procedure of CGA is briefly described as follows.

Step 1 (Initiation). Randomly generate an initial population with N_{pop} feasible individuals of N_{var} real-value parameters in the variable range.

Step 2 (Evaluation). Evaluate the fitness for each individual by the cost function. Let I_{best} be the best individual in the population.

Step 3 (Selection). In one generation, only top $N_{keep} = P_s \times N_{pop}$ individuals are kept for mating and the rest are discarded to make room for the new offspring, where P_s is selection fraction.

Step 4 (Crossover). Using the ranking selection method, two individuals are chosen and paired from N_{keep} kept individuals to produce two new offspring by performing the crossover operation. Paring and crossover take place until $N_{pop} - N_{keep}$ are born to replace the discarded individuals.

Step 5 (Mutation). Mutate all the values except the best individual I_{best} according to a mutation probability P_m. The best individual is excluded to prevent the best cost value from deteriorating. The mutation is realized by replacing the value of mutated variable by a new random value in the variable range.

Step 6 (Termination). If the specified number N_{stop} of iterative generations is achieved or some stopping condition is met, terminate the algorithm. Otherwise, go to Step 2.

In this paper, we use the crossover operator introduced in [164] which will be described in detail in the following. Suppose the S-system to be inferred is N-dimensional, that is, it has $2N^2$ kinetic orders. After the aforementioned Step 1, a population of N_{pop} individuals is generated. Each individual, I_k, is a $2N^2$-dimensional vector of values of kinetic orders, where $k = 1, 2, \ldots, N_{pop}$. Then, after Step 2 and Step 3, top N_{keep} individuals are kept and the others are abandoned.

In Step 4, first, we choose two from the pool of N_{keep} individuals as parents. The N_{keep} individuals are ranked according to their cost values in ascending order. The individual with the rank j is chosen with the

probability p_j as

$$p_j = \frac{N_{keep} - j + 1}{\sum_{j=1}^{N_{keep}} j}.$$

When two parents, I_{p1} and I_{p2}, are chosen, for convenience, we denote them as

$$I_{p1} = [I_{p1}^{(1)}, I_{p1}^{(2)}, \ldots, I_{p1}^{(2N^2)}],$$
$$I_{p2} = [I_{p2}^{(1)}, I_{p2}^{(2)}, \ldots, I_{p2}^{(2N^2)}],$$

where $I_{pk}^{(i)}$, $i = 1, 2, \ldots, 2N^2$, is the ith element of vector I_{pk}, $k = 1, 2$. Then, the crossover begins with randomly select a variable in the pair of parents to be the crossover point. Assume that the crossover position is η, then the selected variables are combined to form new variables that will appear in the children as follows.

$$I_{p1}^{(\eta,new)} = I_{p1}^{(\eta)} - \omega(I_{p1}^{(\eta)} - I_{p2}^{(\eta)}),$$
$$I_{p2}^{(\eta,new)} = I_{p2}^{(\eta)} + \omega(I_{p1}^{(\eta)} - I_{p2}^{(\eta)}),$$

where ω is a random value between 0 and 1. The values of new variables do not lie outside the variable range as $\omega \in (0, 1)$. The final step is to complete the crossover similarly as the binary genetic algorithm, resulting in two offspring.

$$I_{child,1} = [I_{p1}^{(1)}, I_{p1}^{(2)}, \ldots, I_{p1}^{(\eta,new)}, I_{p2}^{(\eta+1)}, \ldots, I_{p2}^{(2N^2)}],$$
$$I_{child,2} = [I_{p2}^{(1)}, I_{p2}^{(2)}, \ldots, I_{p2}^{(\eta,new)}, I_{p1}^{(\eta+1)}, \ldots, I_{p1}^{(2N^2)}].$$

That is, the variables to the right of the crossover point are swapped. If the last variable is selected as the crossover point, then the variables to the left of the selected variables are swapped.

7.4.2 Hybrid Algorithm

Results in Section 7.3 show that PSPEA can correctly infer the S-system with the initial values chosen around the true parameter values. In other words, the PSPEA can locally infer the S-system. In the following, we combine the CGA and PSPEA to develop a hybrid algorithm (PSPE-CGA). The proposed hybrid algorithm combines the global-search properties of the CGA with the local S-system inference ability of PSPEA. Given a set of time-series data, the procedure to infer an N dimensional S-system, whose kinetic orders fall into the range $[K_l, K_h]$, by the proposed hybrid algorithm is described as follows.

Step 1. For each time series, estimate the slop, $S_i(t_k)$, at each time point by (7.2).

Step 2. Randomly generate an initial population with N_{pop} feasible individuals of $2N^2$ real-value parameters in the range $[K_l, K_h]$.

Step 3. For each individual, I_k, $k = 1, 2, \ldots, N_{pop}$, take it as initial values and use the Matlab *fmincon* function to optimize the objective function (7.7). The parameters are subjected to the range $[K_l, K_h]$. Prune the solution with threshold θ.

Step 4. If any parameter, g_{ij} or h_{ij}, has been pruned in Step 3, then take the pruned solution as initial values and go to Step 3, in which the pruned parameters are constrained to be zero. Otherwise go to Step 5.

Step 5. Fix the zero-value parameters in solution from Step 4 to be zeros and refine the nonzero parameters by optimizing objective function (7.6) with Matlab function *fmincon*.

Step 6. After Step 5, an updated individual I'_k for I_k is obtained. Do Step 3–5 for each individual in the population, then we get an updated population I'_k, $k = 1, 2, \ldots, N_{pop}$.

Step 7. The optimized objective value of (7.6) is assigned as cost value to corresponding updated individual I'_k. Then, perform the Selection, Crossover and Mutation operations mentioned above in CGA.

Step 8. A new generation of population is obtained from Step 7. If the specified number N_{stop} of iterative generations is achieved or some stopping condition is met, then terminate the algorithm and go to Step 9, else keep the structure each individual represents in the new generation and go to Step 3.

Step 9. Substitute the solution from Step 8 into (7.5) to calculate the rate constants.

In the hybrid algorithm, PSPE-CGA, the main part of PSPEA is embedded into the CGA. For each individual solution in a generation, Step 3 and Step 4 iteratively prune the solution to make most of the parameters, especially those unnecessary parameters, become zero. Then, the structure that the solution represents is kept and the nonzero parameters are further refined in Step 5. After that, a population of estimated solutions of kinetic orders are obtained. However, each solution may only be a local inference, because the Matlab function *fmincon* can only provide a local solution to the nonlinear optimization problems in this paper. Thus, we perform the selection, crossover and mutation in CGA to these solutions and try to find the global inference of the S-system. PSPEA has the ability of finding a local inference of the S-system, while the CGA has the global-search property. The hybrid algorithm, PSPE-CGA, combines the traits of these two algorithms to make global inference of the S-system. Figure 7.5 shows the flowchart of the hybrid algorithm.

7.4.3 Experiments

To show the effectiveness of the hybrid algorithm, we apply it to infer the S-systems from simulated time series data.

Example 3. Consider a 3-dimensional linear pathway with feedback in Figure 7.6 [65]. In this system, there are 3 dependent variables which are represented by an S-system (7.10) with the associated parameter

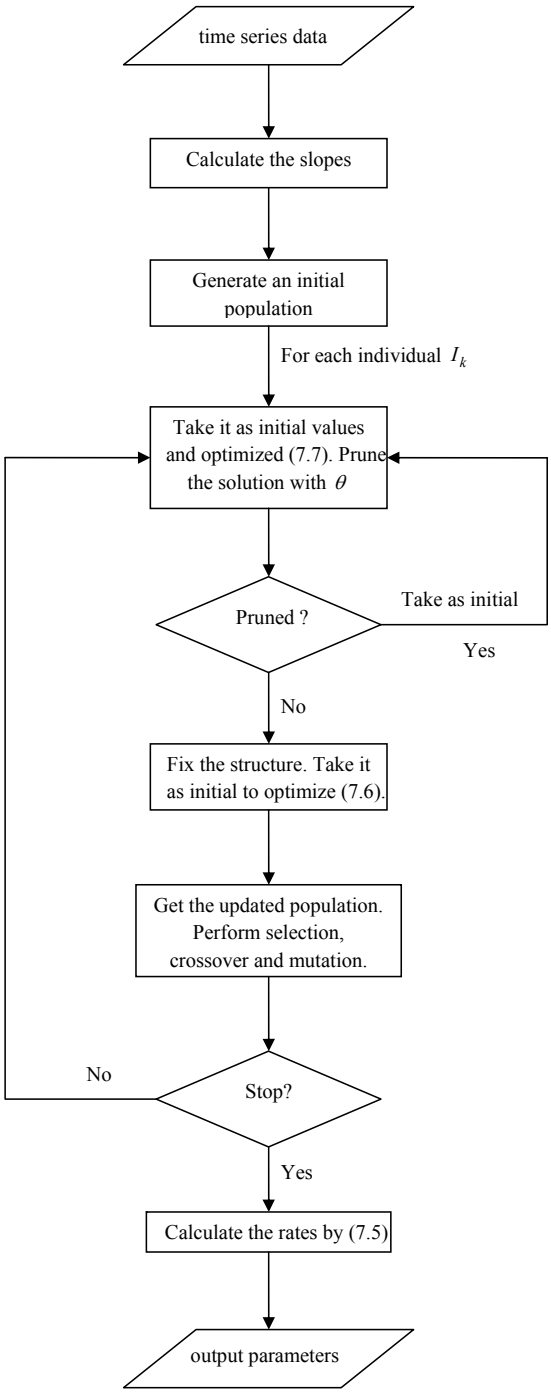


Figure 7.5: Flow chart of the hybrid algorithm, PSPE-CGA.

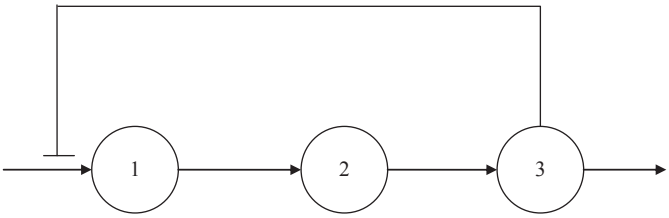


Figure 7.6: A 3-dimensional linear pathway with feedback.

values shown in Table 7.5. For this system, the efflux from X_1 is identical to the influx into X_2, and the efflux from X_2 is equal to the influx into X_3. Hence, as shown in Table 7.5, the degradation term of X_1 is the same as the production term of X_2, and the degradation term of X_2 is exactly the same as the production term of X_3.

$$\begin{aligned} \dot{X}_1 &= \alpha_1 X_3^{g_{13}} - \beta_1 X_1^{h_{11}}, \\ \dot{X}_2 &= \alpha_2 X_1^{g_{21}} - \beta_2 X_2^{h_{22}}, \\ \dot{X}_3 &= \alpha_3 X_2^{g_{32}} - \beta_3 X_3^{h_{33}}. \end{aligned} \tag{7.10}$$

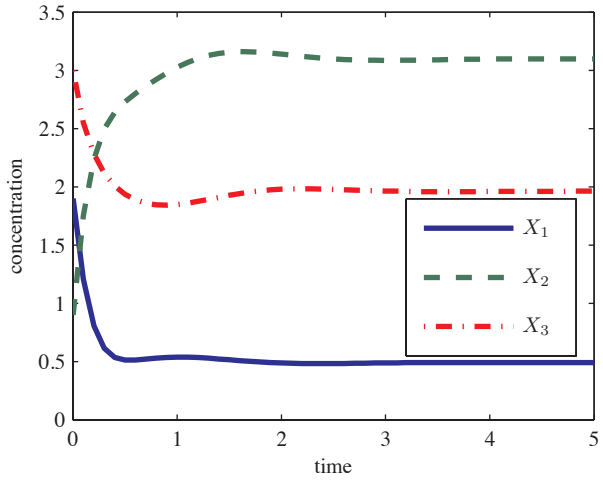


Figure 7.7: Trajectories of variables in (7.10).

Table 7.5: Values of parameters in 3-dimensional linear pathway.

i	α_i	g_{i1}	g_{i2}	g_{i3}	β_i	h_{i1}	h_{i2}	h_{i3}
1	12	0	0	-0.8	10	0.5	0	0
2	10	0.5	0	0	3	0	0.75	0
3	3	0	0.75	0	5	0	0	0.5

For this 3-dimensional S-system, there are totally 24 parameters consisting of 6 positive rate constants,

6 nonzero kinetic orders and 12 zero-value kinetic orders. A set of simulated noise-free time series data is generated from (7.10) with the initial state $x_1 = 1.9$, $x_2 = 0.9$ and $x_3 = 3.0$. The trajectories of the state variables are plotted in Figure 7.7. Time t starts from 0 to 5 with the sampling interval $\Delta t = 0.1$. These three variables reach steady state at the end.

Table 7.6: Results from hybrid algorithm for 3-dimensional linear pathway.

i	α_i	g_{i1}	g_{i2}	g_{i3}	β_i	h_{i1}	h_{i2}	h_{i3}
1	10.9658	0	0	-0.9686	8.8079	0.6078	0	0
2	9.0123	0.5882	0	0	2.1709	0	0.8886	0
3	2.9355	0	0.7573	0	4.9190	0	0	0.5044

We apply the hybrid algorithm to infer the S-system from these time series data. The parameters in the algorithm are set as follows. Let $\theta = 0.001$ and $\lambda = 0.5$. Set the population size $N_{pop} = 14$, selection fraction $P_s = 0.5$ and mutation probability $P_m = 0.2$. All the kinetic orders are searched in the interval $[-1, 1]$. When using the Matlab function *fmincon*, we choose the interior-point method. The estimated results are shown in Table 7.6. As shown in the results, all the non-zero kinetic order parameters are correctly recognized, that is, the structure of the system is successfully identified. The value of the non-zeros parameters are close to their true values and the final values of the optimized objective function (7.6) are less than 10^{-3}.

Table 7.7: Results from the hybrid algorithm for the 4-dimensional S-system example.

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}
1	11.9723	0	0	-0.8032	0	9.9678	0.5018	0	0	0
2	8.0066	0.4994	0	0	0	3.0059	0	0.7491	0	0
3	3.0184	0	0.7466	0	0	5.0126	0	0	0.4970	0.1981
4	1.9990	0.5012	0	0	0	6.0013	0	0	0	0.8010

Example 4. In this example, we apply the hybrid algorithm to infer the 4-dimensional S-system (7.8) in Example 1. Use the same time series data in Example 1 and set the threshold $\theta = 0.05$, $\lambda = 1$, population size $N_{pop} = 12$, selection fraction $P_s = 0.5$, and mutation probability $P_m = 0.2$. Search the kinetic orders in the interval $[-2, 2]$ and choose interior-point method when using the Matlab *fmincon* function. The estimated results are shown in Table 7.7. The results show that the structure of S-system is correctly identified and the estimated values of non-zeros parameters are very close to their true values. The optimized objective function (7.6) is less than 10^{-2}.

7.5 Discussion

This paper aims at the inference of the S-system from time-series data, which is achieved through two phases: First, the identification of its structure, that is, the recognization of zero or non-zero kinetic order parameters. Second, with the identified structure, the values of non-zero parameters are further estimated. Obviously,

the structure identification is very important which is realized by the pruning strategy introduced in this paper. The pruning strategy plays important roles in both PSPEA and the hybrid algorithm, PSPE-CGA. To further investigate the effect of pruning strategy, we perform the following comparisons.

Since the main extension to SPEM in PSPEA is the implementation of the pruning strategy, In the following, we employ the relative error defined in (7.11) and accuracy in (7.12) to compare the performances of SPEM and PSPEA.

Let $\gamma = [\Gamma_1^T, \ldots, \Gamma_N^T, g_1^T, \ldots, g_N^T, h_1^T, \ldots, h_N^T]^T$, which contains all the parameters of an S-system in a vector. Then the relative estimation error is defined as follows:

$$\text{error} = \frac{\|\gamma_{est} - \gamma_{true}\|_2}{\|\gamma_{true}\|_2} \tag{7.11}$$

where γ_{est} is the estimated parameter value and γ_{true} is the real parameter value.

Knowing the system structure is very important. Recall that the network structure is mapped onto the S-system kinetic orders. The signs of kinetic orders reflect the relationships among the components in the network. Therefore, the system structure identification can be achieved by studying the signs of kinetic orders. Let $r = [g_1^T, \ldots, g_N^T, h_1^T, \ldots, h_N^T]^T$, which includes all the kinetic order parameters. The length of vector r is $2N^2$. Suppose r_{true} is true parameter value and r_{est} is the estimated value. The structure identification accuracy is defined as follows:

$$\begin{aligned} &\text{accuracy} \\ =&\frac{\#\{1 \le i \le 2N^2 : \text{sign}[r_{true}(i)] = \text{sign}[r_{est}(i)]\}}{2N^2} \end{aligned} \tag{7.12}$$

where # represents the number of elements in the following set, $r_{true}(i)$ and $r_{est}(i)$ are the ith components of vector r_{true} and r_{est}, respectively.

In the following, the effect of pruning strategy is studied from two aspects: the parameter estimation error and structure identification accuracy. For comparison, both PSPEA and SPEM are applied to infer the previous 4-dimensional S-system from the same simulated time series data as before. The initial value for optimization process is still generated as $(g_1^0; h_i^0) = (g_i; h_i) + \sigma\varepsilon$. σ changes from 0.1 to 1 with the step length 0.1. For each value of σ, both methods run 100 times with various random initial guess values and then 100 estimation results are obtained for each method. We compare the average estimation error and average identification accuracy of each method with respect to a specific σ value. The results are plotted in Figure 7.8 and 7.9.

The estimation errors are illustrated in Figure 7.8(a) and their boxplot is shown in Figure 7.9(a). It can be seen that the proposed algorithm PSPEA has much lower estimation error, around 0.078, than the method SPEM, around 0.576. Figure 7.8(b) shows the identification accuracy for each method and their boxplot is also shown in Figure 7.9(b). PSPEA shows much higher identification accuracy, almost 100%, than SPEM, around 55%. Therefore, the pruning strategy introduced in this paper can significantly reduce the estimation

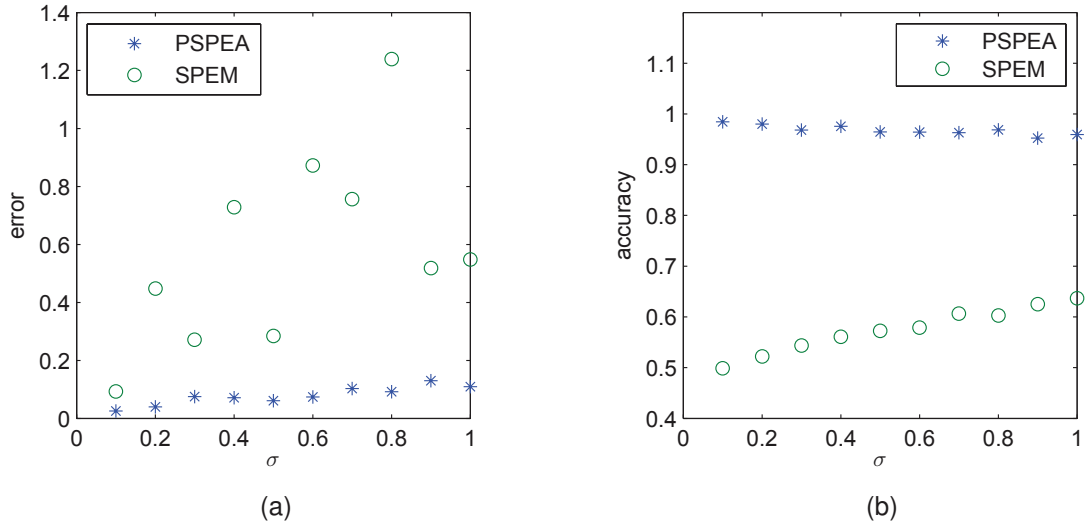


Figure 7.8: (a) Estimation error and (b) identification accuracy for each σ.

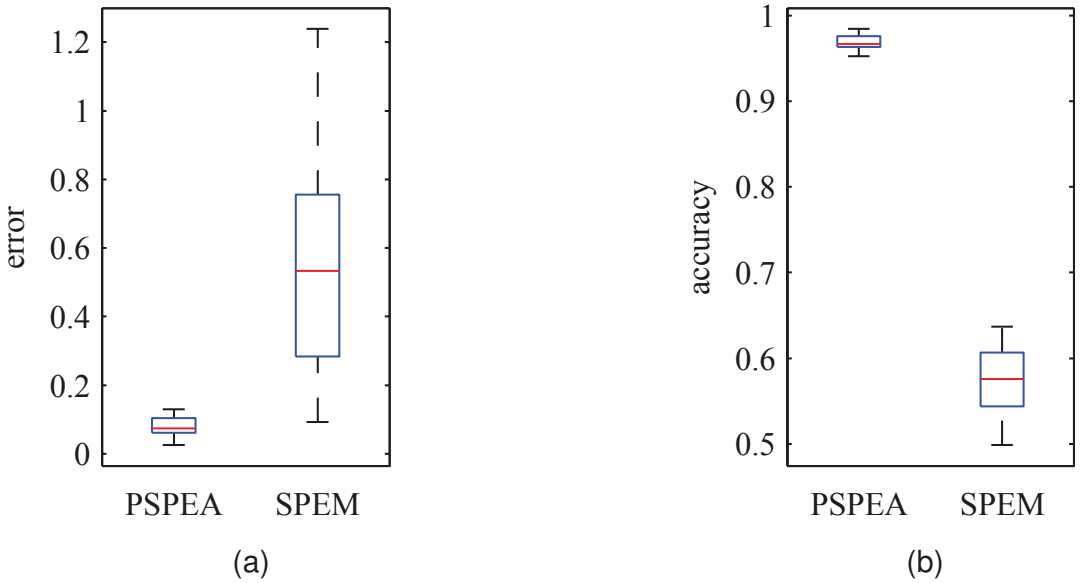


Figure 7.9: Boxplot of (a) average estimation error and (b) average identification accuracy.

error and improve the structure identification accuracy.

7.6 Conclusion

This paper aims to infer the biological S-system from time-series data without any structure information. A novel algorithm called PSPEA is developed by combing a pruning strategy and the separable parameter estimation method. The pruning strategy includes adding an ℓ_1 regularization term to the original objective function and applying a threshold value to prune the solution in the optimization process. Using the separable parameter estimation method can reduce the dimension of parameter searching space. There is no need to provide initial values for rate constants. As examples, a 4-dimensional and a 5-dimensional S-systems have been studied by our proposed algorithm. In each example, the system structure, that is, all zero-valued parameters and the signs of the other non-zero parameters, has been exactly identified and the estimated results of the non-zero parameters are very close to their true values. To improve the robustness to the initial values, we developed a hybrid algorithm, PSPE-CGA, which embeds the main part of PSPEA into the frame of CGA, so that the hybrid algorithm has the global-search properties of CGA and the local inference ability of PSPEA. A 3-dimensional and a 4-dimensional S-systems are successfully inferred by the hybrid algorithm. The effect of the pruning strategy has been studied from two aspects: the parameter estimation error and structure identification accuracy. The results show that compared with the method SPEM, PSPEA with the pruning strategy has much lower estimation error and much higher identification accuracy.

In the proposed algorithms, there are two parameters whose values are assigned by our experience. One parameter is the scalar associated with ℓ_1 regularization term and the other is the pruning threshold. One direction of the future work is to develop a method to determine the values of these two parameters. Another direction of future work is to develop a method to infer the S-system from multiple datasets to improve the inference accuracy.

Acknowledgments

This study is supported by Natural Science and Engineering Research Council of Canada (NSERC).

Chapter 8

Conclusions and Future Work

8.1 Overview and Conclusions

Reverse engineering of the GRNs from gene expression data is a challenging problem in systems biology. It has broad applications in both research and practice. The difficulties come from the problem itself and the limited information contained in the data. This thesis aims at approaching to the solution of the problem with three objectives listed in Chapter 1. The works to achieve those objectives constitute the main context of the thesis.

Chapter 2 provides a comprehensive review of the methods for reverse engineering of GRNs and accomplishes Objective 1. Based on the linear model, Chapter 3 studies the properties of two sparse penalties used to infer GRNs from time-course gene expression data and Chapter 4 proposes a robust method to infer the GRNs from multiple time-course data. These two chapters complete Objective 2. Objective 3 focuses on inferring GRNs based on nonlinear models, specifically, S-systems. Due to the difficulty of the problem, It has been divided into two specific objectives: (i) estimating the parameters with known system structure; (ii) inferring the S-systems without knowing the system structure. Chapter 5 and Chapter 6 develop two methods to estimate the parameters of S-systems from time-course data and thus fulfill the Objective 3(i). Chapter 7 proposes a method to globally infer the S-systems from time-course data and thus fulfills Objective 3(ii). Therefore, all the objectives of this thesis are achieved.

In summary, the following works have been done and presented in this thesis:

- Reviewing and discussing current methods of reverse engineering of GRNs.
- Based on the linear model, analyzing the statistical properties of two penalties, adaptive LASSO and SCAD, used for inferring GRNs from time-course gene expression data.
- Based on the linear model, developing a method to infer the GRNs from multiple time-course gene expression datasets, which is also robust to the large noises and outliers.
- Developing methods to estimate the parameters in S-systems from time-course data with the system structure known.
- Developing methods to infer the S-systems from time-course data without knowing the system structure.

The following conclusions can be drawn from the research results:

- Reverse engineering of GRNs from biological data is an extremely difficult problem. Each method has its advantages and disadvantages. The results from all methods have limited accuracy because of the limited information in the data. Robust methods that use prior biological knowledge and other source of information should be developed.
- Based on the linear model, the adaptive LASSO and SCAD penalties have the "oracle properties" and therefore can asymptotically reconstruct the GRNs from time-course gene expression data.
- Based on the linear model, the developed method, Huber group LASSO, can effectively infer the GRNs from multiple time-course gene expression datasets and is robust to large errors and outliers in the data. The convergence of the proposed inference algorithm is mathematically proved.
- The developed method, alternating weighted least squares (AWLS), for estimating parameters of S-system from time-course data is derived from a clear optimization objective function, takes advantage of the special form of the model and outperforms the existing method, alternating regression (AR), significantly.
- The developed method, auxiliary function guided coordinate descent (AFGCD), estimates the parameters of S-systems from time-course data in an efficient way and its convergence is mathematically guaranteed.
- The developed method, pruning separable parameter estimation algorithm (PSPEA), can locally infer the S-system from time-course data. The hybrid algorithm which combines PSPEA and the continuous genetic algorithm (CGA) can globally infer the S-systems.

8.2 Contributions

Briefly, this thesis has advanced the reverse engineering of GRNs by providing a comprehensive review, developing novel methods based on linear and nonlinear models. The specific contributions of the thesis are summarized as follows:

- A comprehensive review of the mainstream methods for GRN reconstruction has been provided. It has analyzed the major challenges of the problem and discussed the pros and cons of each method.
- A theoretical foundation has been provided for the methods using adaptive LASSO or SCAD penalties based on the linear model to infer the GRNs from time-course data. With this foundation, these methods and their extensions are promising to succeed in GRN inference.
- A method which is able to integrate multiple time-course gene expression datasets to infer the GRNs and is robust to large errors and outliers has been developed in the thesis. The convergence analysis of the inference algorithm has also been provided.

- A method, AWLS, which takes advantage of the special form of S-systems model, has been developed to estimate the parameters of S-systems from time-course data. It outperforms the existing method, AR, significantly.

- A method, AFGCD, which utilizes the auxiliary function and coordinate descent techniques, has been developed to estimate the parameters of S-systems from time-course data. The convergence of this method is mathematically guaranteed.

- A novel method, PSPEA, has been developed to locally infer the S-system without knowing the system structure. This method has also been combined with CGA to form a hybrid algorithm which can globally infer the S-system from time-course data.

8.3 Future Work

The methods developed in this thesis have some limitations. First, only the time-course gene expression data are used in the study. As the information contained in the gene expression data is very limited, the accuracy of the methods is still not very high. Second, because of the nonlinearity of real biological systems, the linear models should be extended such that they can capture the nonlinear behaviors of the system to a certain degree. Third, the method developed in this study for the inference of S-systems can only be applied to relatively small networks. The method is too slow for very large networks. Fourth, the system structure is assumed to remain the same during the period the time-course data are collected. However, since the GRNs are dynamic systems, its system structure may happen to change during the experiments.

To overcome these limitations, some directions of future work along this study are listed below:

(1) Since the information in gene expression data is limited, to improve the inference accuracy, other source of information should be incorporated into the methods. One source of the information is the prior biological knowledge. In this thesis, the sparseness of the GRN is the most important prior knowledge and also constraint used in the methods. Other knowledge, such as the scale-free property of the GRN, could be able to improve the quality of the inferred network. Another source is various other types of biological data. For example, in protein protein interaction (PPI) data, if there exists a link between two proteins, it has a high probability that these two proteins form a complex and therefore the corresponding two genes tend to regulate the same target gene.

(2) Although methods based on linear models can be applied to large networks, the real biological system is nonlinear and cannot be well approximated by linear models. Some extensions to the linear models may be considered and lead to nonlinear models which are not as complex as models such as S-systems and can still be applied to large networks by using similar techniques for linear models. For example, the regulation from one gene to the target gene can be modeled by nonlinear function while effects from different regulatory genes are additive.

(3) In this thesis, the AFGCD method has been developed to estimate the parameters in S-systems with the system structure known. This method could be extended to infer the S-systems without knowing the system structure by utilizing the sparse penalties. The extended method may be efficient enough to be applied to relatively large networks.

(4) Considering the system structure may change during the experiments, models that can describe the structures switching can be used to infer and detect the changing behaviors of the system. The piecewise linear function could be a good candidate while the inference of changing point is challenging.

(5) The methods developed in this thesis could be extended to other relevant biological studies, such as drug target predication and repositioning, genome-wide association study (GWAS) and expression quantitative trait loci (eQTL) mapping.

References

[1] X. Yang, J. E. Dent, and C. Nardini, "An s-system parameter estimation method (spem) for biological networks," *Journal of Computational Biology*, vol. 19, no. 2, pp. 175–187, 2012.

[2] H. D. Jong, "Modeling and simulation of genetic regulatory systems: A literature review," *Journal of Computational Biology*, vol. 9, pp. 67–103, 2002.

[3] P. Brazhnik, A. de la Fuente, and P. Mendes, "Gene networks: how to put the function in genomics," *Trends in Biotechnology*, vol. 20, no. 11, pp. 467–472, 2002.

[4] D. J. Duggan, M. Bittner, Y. Chen, P. Meltzer, and J. M. Trent, "Expression profiling using cdna microarrays," *Nature Genetics*, vol. 21, no. 10, pp. 10–14, 1999.

[5] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown, "Quantitative monitoring of gene expression patterns with a complementary dna microarray," *Science*, vol. 270, no. 5235, pp. 467–470, 1995.

[6] M. Chee, R. Yang, E. Hubbell, A. Berno, X. C. Huang, D. Stern, J. Winkler, D. J. Lockhart, M. S. Morris, and S. P. A. Fodor, "Accessing genetic information with high-density dna arrays," *Science*, vol. 274, no. 5287, pp. 610–614, 1996.

[7] Z. Bar-Joseph, "Analyzing time series gene expression data," *Bioinformatics*, vol. 20, no. 16, pp. 2493–2503, 2004.

[8] T. S. Gardner and J. J. Faith, "Reverse-engineering transcription control networks," *Physics of Life Reviews*, vol. 2, no. 1, pp. 65 – 88, 2005.

[9] Y. Huang, I. Tienda-Luna, and Y. Wang, "Reverse engineering gene regulatory networks," *Signal Processing Magazine, IEEE*, vol. 26, pp. 76 –97, jan. 2009.

[10] F.-X. Wu, L. Wu, J. Wang, J. Liu, and L. Chen, "Transittability of complex networks and its applications to regulatory biomolecular networks," *Scientific Reports*, vol. 4, no. 4819, 2014.

[11] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabsi, "Observability of complex systems," *Proceedings of the National Academy of Sciences*, 2013.

[12] A. Shojaie and G. Michailidis, "Discovering graphical granger causality using the truncating lasso penalty," *Bioinformatics*, vol. 26, no. 18, pp. i517–i523, 2010.

[13] J. Peng, P. Wang, N. Zhou, and J. Zhu, "Partial correlation estimation by joint sparse regression models," *Journal of the American Statistical Association*, vol. 104, no. 486, pp. 735–746, 2009.

[14] A. Tenenhaus, V. Guillemot, X. Gidrol, and V. Frouin, "Gene association networks from microarray data using a regularized estimation of partial correlation based on pls regression," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 7, pp. 251 –262, april-june 2010.

[15] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society B*, vol. 58, pp. 267–288, 1996.

[16] H. Zou, "The adaptive lasso and its oracle properties," *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006.

[17] J. Fan and R. Li, "Variable Selection via Nonconcave Penalized Likelihood and Its Oracle Properties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[18] L.-Z. Liu, F.-X. Wu, and W.-J. Zhang, "Reverse engineering of gene regulatory networks from biological data," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 5, pp. 365–385, 2012.

[19] A. J. Hartemink, "Reverse engineering gene regulatory networks," *Nat Biotech*, vol. 23, pp. 554–555, May 2005.

[20] T. S. Gardner, D. di Bernardo, D. Lorenz, and J. J. Collins, "Inferring genetic networks and identifying compound mode of action via expression profiling," *Science*, vol. 301, no. 5629, pp. 102–105, 2003.

[21] M. Hecker, S. Lambeck, S. Toepfer, E. van Someren, and R. Guthke, "Gene regulatory network inference: Data integration in dynamic models–a review," *Biosystems*, vol. 96, no. 1, pp. 86 – 103, 2009.

[22] S. Mitra, R. Das, and Y. Hayashi, "Genetic networks and soft computing," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 8, pp. 94 –107, jan.-feb. 2011.

[23] J. Hubble, J. Demeter, H. Jin, M. Mao, M. Nitzberg, T. B. K. Reddy, F. Wymore, Z. K. Zachariah, G. Sherlock, and C. A. Ball, "Implementation of genepattern within the stanford microarray database," *Nucleic Acids Research*, vol. 37, no. suppl 1, pp. D898–D901, 2009.

[24] G. D. Stormo, "Dna binding sites: representation and discovery," *Bioinformatics*, vol. 16, no. 1, pp. 16–23, 2000.

[25] T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, N. M. Hannett, C. T. Harbison, C. M. Thompson, I. Simon, J. Zeitlinger, E. G. Jennings, H. L. Murray, D. B. Gordon, B. Ren, J. J. Wyrick, J.-B. Tagne, T. L. Volkert, E. Fraenkel, D. K. Gifford, and R. A. Young, "Transcriptional regulatory networks in saccharomyces cerevisiae," *Science*, vol. 298, no. 5594, pp. 799–804, 2002.

[26] N. Nariai, S. Kim, S. Imoto, and S. Miyano, "Using protein-protein interactions for refining gene networks estimated from microarray data by bayesian networks.," *Pac Symp Biocomput*, pp. 336–347, 2004.

[27] Y. Tamada, S. Kim, H. Bannai, S. Imoto, K. Tashiro, S. Kuhara, and S. Miyano, "Estimating gene networks from gene expression data by combining bayesian network model with promoter element detection," *Bioinformatics*, vol. 19, no. suppl 2, pp. ii227–ii236, 2003.

[28] A. J. Hartemink, D. K. Gifford, T. S. Jaakkola, and R. A. Young, "Combining location and expression data for principled discovery of genetic regulatory network models.," *Pac Symp Biocomput*, pp. 437–449, 2002.

[29] S. Imoto, T. Higuchi, T. Goto, K. Tashiro, S. Kuhara, and S. Miyano, "Combining microarrays and biological knowledge for estimating gene networks via bayesian networks.," *J Bioinform Comput Biol*, vol. 2, pp. 77–98, Mar 2004.

[30] S. T. Jensen, G. Chen, and C. J. Stoeckert, "Bayesian variable selection and data integration for biological regulatory networks," *Annals of Applied Statistics*, vol. 1, no. 2, pp. 612–633, 2007.

[31] C. Sabatti and G. M. James, "Bayesian sparse hidden components analysis for transcription regulation networks," *Bioinformatics*, vol. 22, no. 6, pp. 739–746, 2006.

[32] S. A. Kauffman, *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, USA, 1 ed., June 1993.

[33] T. Akutsu, S. Kuhara, O. Maruyama, and S. Miyano, "Identification of genetic networks by strategic gene disruptions and gene overexpressions under a boolean model," *Theoretical Computer Science*, vol. 298, no. 1, pp. 235 – 251, 2003. ¡ce:title¿Selected Papers in honour of Setsuo Arikawa¡/ce:title¿.

[34] T. Akutsu, S. Miyano, and S. Kuhara, "Identification of genetic networks from a small number of gene expression patterns under the boolean network model.," *Pac Symp Biocomput*, pp. 17–28, 1999.

[35] S. Liang, S. Fuhrman, and R. Somogyi, "Reveal, a general reverse engineering algorithm for inference of genetic network architectures.," *Pac Symp Biocomput*, pp. 18–29, 1998.

[36] H. H. McAdams and A. Arkin, "Stochastic mechanisms in gene expression," *Proceedings of the National Academy of Sciences*, vol. 94, no. 3, pp. 814–819, 1997.

[37] I. Shmulevich, E. R. Dougherty, and W. Zhang, "Gene perturbation and intervention in probabilistic boolean networks," *Bioinformatics*, vol. 18, no. 10, pp. 1319–1331, 2002.

[38] I. Shmulevich, E. R. Dougherty, S. Kim, and W. Zhang, "Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks," *Bioinformatics*, vol. 18, no. 2, pp. 261–274, 2002.

[39] I. Shmulevich, E. Dougherty, and W. Zhang, "From boolean to probabilistic boolean networks as models of genetic regulatory networks," *Proceedings of the IEEE*, vol. 90, pp. 1778 – 1792, nov 2002.

[40] l. Shmulevich, E. R. Dougherty, and W. Zhang, "Control of stationary behavior in probabilistic Boolean networks by means of structural intervention," *Journal of Biological Systems*, vol. 10, no. 4, pp. 431–445, 2002.

[41] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using bayesian networks to analyze expression data.," *J Comput Biol*, vol. 7, no. 3-4, pp. 601–620, 2000.

[42] N. Friedman, "Inferring cellular networks using probabilistic graphical models," *Science*, vol. 303, no. 5659, pp. 799–805, 2004.

[43] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: The combination of knowledge and statistical data," *Machine Learning*, vol. 20, pp. 197–243, Sept. 1995.

[44] J. Hasty, D. McMillen, F. Isaacs, and J. J. Collins, "Computational studies of gene regulatory networks: in numero molecular biology," *Nat Rev Genet*, vol. 2, pp. 268–279, Apr. 2001.

[45] I. M. Ong, J. D. Glasner, and D. Page, "Modelling regulatory pathways in e. coli from time series expression profiles," *Bioinformatics*, vol. 18, no. suppl 1, pp. S241–S248, 2002.

[46] S. Kim, S. Imoto, and S. Miyano, "Dynamic bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data," *Biosystems*, vol. 75, no. 13, pp. 57 – 65, 2004.

[47] B.-E. Perrin, L. Ralaivola, A. Mazurie, S. Bottani, J. Mallet, and F. dAlchBuc, "Gene networks inference using dynamic bayesian networks," *Bioinformatics*, vol. 19, no. suppl 2, pp. ii138–ii148, 2003.

[48] S. Y. Kim, S. Imoto, and S. Miyano, "Inferring gene networks from time series microarray data using dynamic bayesian networks," *Briefings in Bioinformatics*, vol. 4, no. 3, pp. 228–235, 2003.

[49] M. K. S. Yeung, J. Tegnr, and J. J. Collins, "Reverse engineering gene networks using singular value decomposition and robust regression," *Proceedings of the National Academy of Sciences*, vol. 99, no. 9, pp. 6163–6168, 2002.

[50] M. Bansal, V. Belcastro, A. Ambesi-Impiombato, and D. di Bernardo, "How to infer gene networks from expression profiles," *Mol Syst Biol*, vol. 3, pp. –, Feb. 2007.

[51] M. Bansal, G. D. Gatta, and D. di Bernardo, "Inference of gene regulatory networks and compound mode of action from time course gene expression profiles," *Bioinformatics*, vol. 22, no. 7, pp. 815–822, 2006.

[52] D. di Bernardo, M. J. Thompson, T. S. Gardner, S. E. Chobot, E. L. Eastwood, A. P. Wojtovich, S. J. Elliott, S. E. Schaus, and J. J. Collins, "Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks," *Nat Biotech*, vol. 23, pp. 377–383, Mar. 2005.

[53] F. Gregoretti, V. Belcastro, D. di Bernardo, and G. Oliva, "A parallel implementation of the network identification by multiple regression (nir) algorithm to reverse-engineer regulatory gene networks," *PLoS ONE*, vol. 5, p. e10179, 04 2010.

[54] Y. Wang, T. Joshi, X.-S. Zhang, D. Xu, and L. Chen, "Inferring gene regulatory networks from multiple microarray datasets," *Bioinformatics*, vol. 22, no. 19, pp. 2413–2420, 2006.

[55] R. Laubenbacher and B. Stigler, "A computational algebra approach to the reverse engineering of gene regulatory networks," *Journal of Theoretical Biology*, vol. 229, no. 4, pp. 523 – 537, 2004.

[56] M. Gustafsson, M. Hornquist, and A. Lombardi, "Constructing and analyzing a large-scale gene-to-gene regulatory network-lasso-constrained inference and biological validation," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 2, pp. 254–261, 2005.

[57] J. Mazur, D. Ritter, G. Reinelt, and L. Kaderali, "Reconstructing nonlinear dynamic models of gene regulation using stochastic sampling," *BMC Bioinformatics*, vol. 10, no. 1, p. 448, 2009.

[58] U. Alon, *An Introduction to Systems Biology: Design Principles of Biological Circuits (Chapman & Hall/CRC Mathematical & Computational Biology)*. Chapman and Hall/CRC, 1 ed., July 2006.

[59] E. O. Voit and J. Almeida, "Decoupling dynamical systems for pathway identification from metabolic profiles," *Bioinformatics*, vol. 20, no. 11, pp. 1670–1681, 2004.

[60] E. O. Voit, *Computational Analysis of Biochemical Systems: A Practical Guide for Biochemists and Molecular Biologists*. Cambridge University Press, Sept. 2000.

[61] S. Kikuchi, D. Tominaga, M. Arita, K. Takahashi, and M. Tomita, "Dynamic modeling of genetic networks using genetic algorithm and s-system," *Bioinformatics*, vol. 19, no. 5, pp. 643–650, 2003.

[62] S. Kimura, K. Ide, A. Kashihara, M. Kano, M. Hatakeyama, R. Masui, N. Nakagawa, S. Yokoyama, S. Kuramitsu, and A. Konagaya, "Inference of s-system models of genetic networks using a cooperative coevolutionary algorithm," *Bioinformatics*, vol. 21, no. 7, pp. 1154–1163, 2005.

[63] S.-Y. Ho, C.-H. Hsieh, F.-C. Yu, and H.-L. Huang, "An intelligent two-stage evolutionary algorithm for dynamic pathway identification from gene expression profiles," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 4, pp. 648 –704, oct.-dec. 2007.

[64] L.-Z. Liu, F.-X. Wu, and W. Zhang, "Inference of biological s-system using the separable estimation method and the genetic algorithm," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 9, no. 4, pp. 955–965, 2012.

[65] M. Vilela, I.-C. Chou, S. Vinga, A. Vasconcelos, E. Voit, and J. Almeida, "Parameter optimization in s-system models," *BMC Systems Biology*, vol. 2, no. 1, p. 35, 2008.

[66] S. Kimura, Y. Amano, K. Matsumura, and M. Okada-Hatakeyama, "Effective parameter estimation for s-system models using lpms and evolutionary algorithms," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pp. 1 –8, july 2010.

[67] A. J. Butte, P. Tamayo, D. Slonim, T. R. Golub, and I. S. Kohane, "Discovering functional relationships between rna expression and chemotherapeutic susceptibility using relevance networks," *Proceedings of the National Academy of Sciences*, vol. 97, no. 22, pp. 12182–12186, 2000.

[68] A. de la Fuente, N. Bing, I. Hoeschele, and P. Mendes, "Discovery of meaningful associations in genomic data using partial correlation coefficients," *Bioinformatics*, vol. 20, no. 18, pp. 3565–3574, 2004.

[69] A. Wille and P. Bühlmann, "Low-Order Conditional Independence Graphs for Inferring Genetic Networks," *Statistical Applications in Genetics and Molecular Biology*, vol. 5, no. 1, 2006.

[70] R. Castelo and A. Roverato, "A Robust Procedure For Gaussian Graphical Model Search From Microarray Data With p Larger Than n," *J. Mach. Learn. Res.*, vol. 7, pp. 2621–2650, 2006.

[71] R. Opgen-Rhein and K. Strimmer, "Learning causal networks from systems biology time course data: an effective model selection procedure for the vector autoregressive process," *BMC Bioinformatics*, vol. 8, no. Suppl 2, p. S3, 2007.

[72] T. Shimamura, S. Imoto, R. Yamaguchi, A. Fujita, M. Nagasaki, and S. Miyano, "Recursive regularization for inferring gene networks from time-course gene expression profiles," *BMC Systems Biology*, vol. 3, no. 1, p. 41, 2009.

[73] O. Hirose, R. Yoshida, S. Imoto, R. Yamaguchi, T. Higuchi, D. S. Charnock-Jones, C. Print, and S. Miyano, "Statistical inference of transcriptional module-based gene networks from time course gene expression profiles by using state space models," *Bioinformatics*, vol. 24, no. 7, pp. 932–942, 2008.

[74] C. Rangel, J. Angus, Z. Ghahramani, M. Lioumi, E. Sotheran, A. Gaiba, D. L. Wild, and F. Falciani, "Modeling t-cell activation using gene expression profiling and state-space models," *Bioinformatics*, vol. 20, no. 9, pp. 1361–1372, 2004.

[75] M. J. Beal, F. Falciani, Z. Ghahramani, C. Rangel, and D. L. Wild, "A bayesian approach to reconstructing genetic regulatory networks with hidden factors," *Bioinformatics*, vol. 21, no. 3, pp. 349–356, 2005.

[76] F.-X. Wu, "Gene regulatory network modelling: a state-space approach," *Int. J. Data Min. Bioinformatics*, vol. 2, no. 1, pp. 1–14, 2008.

[77] F. X. Wu, W. J. Zhang, and A. J. Kusalik, "Modeling gene expression from microarray expression data with state-space equations.," *Pac Symp Biocomput*, pp. 581–592, 2004.

[78] F.-X. Wu, W. Zhang, and A. J. Kusalik, "State-space model with time delays for gene regulatory networks," *Journal of Biological Systems*, vol. 12, pp. 483–500, 2004.

[79] J. VOHRADSKY, "Neural network model of gene expression," *The FASEB Journal*, vol. 15, no. 3, pp. 846–854, 2001.

[80] R. Xu, D. Wunsch, and R. Frank, "Inference of genetic regulatory networks with recurrent neural network models using particle swarm optimization," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 4, pp. 681 –692, oct.-dec. 2007.

[81] N. Sun, R. J. Carroll, and H. Zhao, "Bayesian error analysis model for reconstructing transcriptional regulatory networks," *Proceedings of the National Academy of Sciences*, vol. 103, no. 21, pp. 7988–7993, 2006.

[82] A. Bernard and A. J. Hartemink, "Informative structure priors: joint learning of dynamic regulatory networks from multiple types of data.," *Pac Symp Biocomput*, pp. 459–470, 2005.

[83] A. Werhli and D. Husmeier, "Reconstructing Gene Regulatory Networks with Bayesian Networks by Combining Expression Data with Multiple Sources of Prior Knowledge," *Statistical Applications in Genetics and Molecular Biology*, vol. 6, no. 1, p. 15, 2007.

[84] Y. Tamada, H. Bannai, S. Imoto, T. Katayama, M. Kanehisa, and S. Miyano, "Utilizing evolutionary information and gene expression data for estimating gene networks with bayesian network models.," *J Bioinform Comput Biol*, vol. 3, pp. 1295–1313, Dec. 2005.

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

[86] J. C. Liao, R. Boscolo, Y.-L. Yang, L. M. Tran, C. Sabatti, and V. P. Roychowdhury, "Network component analysis: Reconstruction of regulatory signals in biological systems," *Proceedings of the National Academy of Sciences*, vol. 100, no. 26, pp. 15522–15527, 2003.

[87] K. C. Kao, Y.-L. Yang, R. Boscolo, C. Sabatti, V. Roychowdhury, and J. C. Liao, "Transcriptome-based determination of multiple transcription regulator activities in escherichia coli by using network component analysis," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 2, pp. 641–646, 2004.

[88] A.-L. Boulesteix and K. Strimmer, "Predicting transcription factor activities from combined analysis of microarray and chip data: a partial least squares approach," *Theoretical Biology and Medical Modelling*, vol. 2, no. 1, p. 23, 2005.

[89] N. Banerjee and M. Q. Zhang, "Identifying cooperativity among transcription factors controlling the cell cycle in yeast," *Nucleic Acids Research*, vol. 31, no. 23, pp. 7024–7031, 2003.

[90] R.-S. Wang, Y. Wang, X.-S. Zhang, and L. Chen, "Inferring transcriptional regulatory networks from high-throughput data," *Bioinformatics*, vol. 23, no. 22, pp. 3056–3064, 2007.

[91] M. Zavlanos, A. Julius, S. Boyd, and G. Pappas, "Identification of stable genetic networks using convex programming," in *American Control Conference, 2008*, pp. 2755 –2760, june 2008.

[92] D. Husmeier, "Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic bayesian networks," *Bioinformatics*, vol. 19, no. 17, pp. 2271–2282, 2003.

[93] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*. Cambridge, MA, USA: The MIT Press, second ed., Jan. 2001.

[94] S. Saito and K. Horimoto, "Co-expressed gene assessment based on the path consistency algorithm: Operon detention in escherichia coli," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pp. 4280 –4286, oct. 2009.

[95] M. Tan, M. Alshalalfa, R. Alhajj, and F. Polat, "Influence of prior knowledge in constraint-based learning of gene regulatory networks," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 8, pp. 130 –142, jan.-feb. 2011.

[96] X. Zhang, X.-M. Zhao, K. He, L. Lv, Y. Cao, J. Liu, J.-K. Hao, Z.-P. Liu, and L. Chen, "Inferring gene regulatory networks from gene expression data by pc-algorithm based on conditional mutual information," *Bioinformatics*, 2011.

[97] L. E. Brown, I. Tsamardinos, and C. F. Aliferis, "A comparison of novel and state-of-the-art polynomial bayesian network learning algorithms," in *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*, AAAI'05, pp. 739–745, AAAI Press, 2005.

[98] P. Sheridan, T. Kamimura, and H. Shimodaira, "A scale-free structure prior for graphical models with applications in functional genomics," *PLoS ONE*, vol. 5, p. e13580, 11 2010.

[99] L.-Z. Liu, F.-X. Wu, and W.-J. Zhang, "Properties of sparse penalties on inferring gene regulatory networks from time-course gene expression data," *IET Systems Biology*, April 2014. published.

[100] A.-C. Haury, F. Mordelet, P. Vera-Licona, and J.-P. Vert, "Tigress: Trustful inference of gene regulation using stability selection," *BMC Systems Biology*, vol. 6, no. 1, p. 145, 2012.

[101] S. Martin, Z. Zhang, A. Martino, and J.-L. Faulon, "Boolean dynamics of genetic regulatory networks inferred from microarray time series data," *Bioinformatics*, vol. 23, no. 7, pp. 866–874, 2007.

[102] R. Xu, D. Wunsch Ii, and R. Frank, "Inference of genetic regulatory networks with recurrent neural network models using particle swarm optimization.," *IEEE/ACM Trans Comput Biol Bioinform*, vol. 4, no. 4, pp. 681–692, 2007.

[103] D. Marbach, R. J. Prill, T. Schaffter, C. Mattiussi, D. Floreano, and G. Stolovitzky, "Revealing strengths and weaknesses of methods for gene network inference," *Proceedings of the National Academy of Sciences*, vol. 107, no. 14, pp. 6286–6291, 2010.

[104] D. Thieffry, A. M. Huerta, E. Prez-Rueda, and J. Collado-Vides, "From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in escherichia coli.," *Bioessays*, vol. 20, pp. 433–440, May 1998.

[105] M. M. Zavlanos, A. A. Julius, S. P. Boyd, and G. J. Pappas, "Inferring stable genetic networks from steady-state data," *Automatica*, vol. 47, no. 6, pp. 1113 – 1122, 2011.

[106] A. Fujita, J. Sato, H. Garay-Malpartida, R. Yamaguchi, S. Miyano, M. Sogayar, and C. Ferreira, "Modeling gene expression regulatory networks with the sparse vector autoregressive model," *BMC Systems Biology*, vol. 1, no. 1, p. 39, 2007.

[107] X. Hu, M. Ng, F.-X. Wu, and B. Sokhansanj, "Mining, modeling, and evaluation of subnetworks from large biomolecular networks and its comparison study," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 13, no. 2, pp. 184–194, 2009.

[108] M. D. Hoon, S. Imoto, and S. Miyano, "Inferring gene regulatory networks from time-ordered gene expression data of bacillus subtilis using differential equations," in *Pac. Symp. Biocomput*, pp. 17–28, 2003.

[109] F.-X. Wu, L.-Z. Liu, and Z.-H. Xia, "Identification of gene regulatory networks from time course gene expression data," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, pp. 795–798, 2010.

[110] P. Zhao and B. Yu, "On model selection consistency of lasso," *J. Mach. Learn. Res.*, vol. 7, pp. 2541–2563, Dec. 2006.

[111] F. Abegaz and E. Wit, "Sparse time series chain graphical models for reconstructing genetic networks," *Biostatistics*, vol. 14, no. 3, pp. 586–599, 2013.

[112] E. Serpedin, A. Noor, M. Nounou, and H. N. Nounou, "Inferring gene regulatory networks via nonlinear state-space models and exploiting sparsity," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 4, pp. 1203–1211, 2012.

[113] N. Lim, Y. enbabaolu, G. Michailidis, and F. dAlch Buc, "Okvar-boost: a novel boosting algorithm to infer nonlinear dynamics and interactions in gene regulatory networks," *Bioinformatics*, vol. 29, no. 11, pp. 1416–1423, 2013.

[114] J. H. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, pp. 1–22, 2 2010.

[115] N. Meinshausen and P. Buhlmann, "Stability selection," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 4, pp. 417–473, 2010.

[116] H. Zou and R. Li, "One-step sparse estimates in nonconcave penalized likelihood models," *The Annals of Statistics*, vol. 36, pp. 1509–1533, Aug. 2008.

[117] H. Lütkepohl, *New Introduction to Multiple Time Series Analysis*. Springer, 1st ed. 2006. corr. 2nd printing ed., Oct. 2007.

[118] J. D. Hamilton, *Time-series analysis*. Princeton Univerity Press, 1 ed., Jan. 1993.

[119] C. J. Geyer, "On the asymptotics of constrained M-estimation.," *Ann. Stat.*, vol. 22, no. 4, pp. 1993–2010, 1994.

[120] K. Knight and W. Fu, "Asymptotics for Lasso-type estimators.," *Ann. Stat.*, vol. 28, no. 5, pp. 1356–1378, 2000.

[121] M. Kabir, N. Noman, and H. Iba, "Reverse engineering gene regulatory network from microarray data using linear time-variant model," *BMC Bioinformatics*, vol. 11, no. Suppl 1, p. S56, 2010.

[122] S. Kimura, S. Nakayama, and M. Hatakeyama, "Genetic network inference as a series of discrimination tasks," *Bioinformatics*, vol. 25, no. 7, pp. 918–925, 2009.

[123] Y.-T. Hsiao and W.-P. Lee, "Inferring robust gene networks from expression data by a sensitivity-based incremental evolution method," *BMC Bioinformatics*, vol. 13, no. Suppl 7, p. S8, 2012.

[124] F. Montefusco, C. Cosentino, and F. Amato, "Core-net: exploiting prior knowledge and preferential attachment to infer biological interaction networks," *Systems Biology, IET*, vol. 4, no. 5, pp. 296–310, 2010.

[125] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, "Comprehensive identification of cell cycleregulated genes of the yeast saccharomyces cerevisiae by microarray hybridization," *Molecular Biology of the Cell*, vol. 9, no. 12, pp. 3273–3297, 1998.

[126] C. Stark, B.-J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, "Biogrid: a general repository for interaction datasets," *Nucleic Acids Research*, vol. 34, no. suppl 1, pp. D535–D539, 2006.

[127] L.-Z. Liu, F.-X. Wu, and W.-J. Zhang, "A group lasso-based method for robustly inferring gene regulatory networks from multiple time-course datasets," *BMC Systems Biology*, March 2014. in press.

[128] L.-Z. Liu, F.-X. Wu, and W.-J. Zhang, "Robust inference of gene regulatory networks from multiple microarray datasets," in *Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on*, pp. 544–547, Dec 2013.

[129] S. Saito, T. Hirokawa, and K. Horimoto, "Discovery of chemical compound groups with common structures by a network analysis approach (affinity prediction method)," *Journal of Chemical Information and Modeling*, vol. 51, no. 1, pp. 61–68, 2011.

[130] K. Basso, A. Margolin, G. Stolovitzky, U. Klein, D. Riccardo, and A. Califano, "Reverse engineering of regulatory networks in human b cells.," *Nature genetics*, vol. 37, no. 4, pp. 382–390, 2005.

[131] A. Pinna, N. Soranzo, and A. de la Fuente, "From knockouts to networks: establishing direct cause-effect relationships through graph analysis.," *PloS one*, vol. 5, no. 10, 2010.

[132] B.-L. Chen, L.-Z. Liu, and F.-X. Wu, "Inferring gene regulatory networks from multiple time course gene expression datasets," in *Systems Biology (ISB), 2011 IEEE International Conference on*, pp. 12–17, 2011.

[133] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[134] S. Rosset and J. Zhu, "Piecewise linear regularized solution paths.," *Ann. Stat.*, vol. 35, no. 3, pp. 1012–1030, 2007.

[135] D. Seung and L. Lee, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, pp. 556–562, 2001.

[136] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of Optimization Theory and Applications*, vol. 109, no. 3, pp. 475–494, 2001.

[137] I. Cantone, L. Marucci, F. Iorio, M. A. Ricci, V. Belcastro, M. Bansal, S. Santini, M. di Bernardo, D. di Bernardo, and M. P. Cosma, "A yeast synthetic network for in vivo assessment of reverse-engineering and modeling approaches.," *Cell*, vol. 137, pp. 172–181, Apr 2009.

[138] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)*. New York: Springer, 2011.

[139] R. Mazumder, J. H. Friedman, and T. Hastie, "Sparsenet: Coordinate descent with nonconvex penalties," *Journal of the American Statistical Association*, vol. 106, no. 495, pp. 1125–1138, 2011.

[140] L.-Z. Liu, F.-X. Wu, and W.-J. Zhang, "Alternating weighted least squares parameter estimation for biological s-systems," in *Systems Biology (ISB), 2012 IEEE 6th International Conference on*, pp. 6–11, 2012.

[141] I.-C. Chou, H. Martens, and E. Voit, "Parameter estimation in biochemical systems models with alternating regression," *Theoretical Biology and Medical Modelling*, vol. 3, no. 1, p. 25, 2006.

[142] E. Klipp, R. Herwig, A. Kowald, C. Wierling, and H. Lehrach, *Systems Biology in Practice: Concepts, Implementation and Application*. Wiley-VCH, May 2005.

[143] J. V. Beck and K. J. Arnold, *Parameter estimation in engineering and science*. Wiley New York, 1977.

[144] O. R. Gonzalez, C. Kper, K. Jung, P. C. Naval, and E. Mendoza, "Parameter estimation using Simulated Annealing for S-system models of biochemical networks," *Bioinformatics*, vol. 23, no. 4, pp. 480–486, 2007.

[145] H. Wang, L. Qian, and E. Dougherty, "Inference of gene regulatory networks using S-system: a unified approach," *Systems Biology, IET*, vol. 4, pp. 145 –156, march 2010.

[146] F.-X. Wu and L. Mu, "Separable Parameter Estimation Method for Nonlinear Biological Systems," in *Bioinformatics and Biomedical Engineering , 2009. ICBBE 2009. 3rd International Conference on*, pp. 1 –4, june 2009.

[147] L.-Z. Liu, F.-X. Wu, L.-L. Han, and W.-J. Zhang, "Structure identification and parameter estimation of biological S-systems," in *Bioinformatics and Biomedicine (BIBM), 2010 IEEE International Conference on*, pp. 329 –334, Dec. 2010.

[148] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least Angle Regression," *Annals of Statistics*, vol. 32, pp. 407–499, 2004.

[149] L.-Z. Liu, F.-X. Wu, and W.-J. Zhang, "Estimating parameters of S-systems by an auxiliary function guided coordinate descent method," *Systems Science & Control Engineering*, vol. 2, no. 1, pp. 125–134, 2014.

[150] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience, 2006.

[151] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

[152] L. Li, L. Wu, H. Zhang, and F. Wu, "Fast multiplicative update algorithms for nonnegative matrix factorization and their convergence," *IEEE Transactions on Neural Networks and Learning Systems*, 2013. accepted.

[153] H. de Jong and M. Page, "Search for steady states of piecewise-linear differential equation models of genetic regulatory networks," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 5, pp. 208–222, April 2008.

[154] R. Ram and M. Chetty, "A markov-blanket-based model for gene regulatory network inference," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 8, pp. 353–367, March 2011.

[155] L. Cheng, Z.-G. Hou, Y. Lin, M. Tan, W. Zhang, and F.-X. Wu, "Recurrent neural network for non-smooth convex optimization problems with application to the identification of genetic regulatory networks," *Neural Networks, IEEE Transactions on*, vol. 22, pp. 714–726, May 2011.

[156] I.-C. Chou and E. O. Voit, "Recent developments in parameter estimation and structure identification of biochemical and genomic systems," *Mathematical Biosciences*, vol. 219, no. 2, pp. 57 – 83, 2009.

[157] F.-X. Wu, Z.-K. Shi, and L. Mu, "Estimating parameters in the caspase-activated apoptosis system," *International Journal of Biomedical Engineering and Technology*, vol. 4, no. 4, pp. 1752–6418, 2010.

[158] T. DAISUKE and P. HORTON, "Inference of scale-free networks from gene expression time series," *Journal of Bioinformatics and Computational Biology*, vol. 04, no. 02, pp. 503–514, 2006.

[159] S. Kimura, M. Hatakeyama, and A. Konagaya, "Inference of s-system models of genetic networks from noisy time-series data," *Chem-Bio Informatics Journal*, vol. 4, no. 1, pp. 1–14, 2004.

[160] K.-Y. Tsai and F.-S. Wang, "Evolutionary optimization with data collocation for reverse engineering of biological networks," *Bioinformatics*, vol. 21, no. 7, pp. 1180–1188, 2005.

[161] D.-Y. Cho, K.-H. Cho, and B.-T. Zhang, "Identification of biochemical networks by s-tree based genetic programming," *Bioinformatics*, vol. 22, no. 13, pp. 1631–1640, 2006.

[162] W. Tucker and V. Moulton, "Parameter reconstruction for biochemical networks using interval analysis," *Reliable Computing*, vol. 12, no. 5, pp. 389–402, 2006.

[163] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[164] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. Wiley-Interscience, 2004.

Appendix A

List of Publications

Journal

[1] **L.Z. Liu**, F.X. Wu, and W.J. Zhang, "A group lasso-based method for robustly inferring gene regulatory networks from multiple time-course datasets," *BMC Systems Biology*, March 2014, in press

[2] **L.Z. Liu**, F.X. Wu, and W.J. Zhang, "Properties of sparse penalties on inferring gene regulatory networks from time-course gene expression data," *IET Systems Biology*, April 2014, published

[3] **L.Z. Liu**, F.X. Wu, and W.J. Zhang, "Estimating parameters of S-systems by an auxiliary function guided coordinate descent method," *Systems Science & Control Engineering*, vol. 2, no. 1, pp. 125-134, 2014

[4] L.P. Tian, **L.Z. Liu**, and F.X. Wu, "Nonlinear-Model-Based Analysis Methods for Time-Course Gene Expression Data," *The Scientific World Journal*, vol. 2014, 2014, Article ID 313747, 9 pages

[5] L.P. Tian, Z.K. Shi, **L.Z. Liu**, and F.X. Wu, "M-matrix-based stability conditions for genetic regulatory networks with time-varying delays and noise perturbations," *IET Systems Biology*, vol. 7, no. 5, pp. 214-222, 2013

[6] L.P. Tian, **L.Z. Liu**, and F.X. Wu, "Matrix Decomposition Methods in Bioinformatics," *Current Bioinformatics*, vol. 8, no. 2, pp. 259-266, 2013

[7] **L.Z. Liu**, F.X. Wu, and W.J. Zhang, "Reverse engineering of gene regulatory networks from biological data," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 5, pp. 365-385, 2012

[8] **L.Z. Liu**, F.X. Wu, and W.J. Zhang, "Inference of biological s-system using the separable estimation method and the genetic algorithm," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 9, no. 4, pp. 955-965, 2012

[9] L.P. Tian, **L.Z. Liu**, Q.W. Zhang and F.X. Wu, "Nonlinear model-based methods for clustering periodically expressed genes," *The Scientific World Journal*, vol. 11, pp. 2051-2061, 2011

Conference

[1] **L.Z. Liu**, F.X. Wu, and W.J. Zhang, "Robust inference of gene regulatory networks from multiple microarray datasets," *in Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on*, pp. 544-547, Dec 2013

[2] **L.Z. Liu**, F.X. Wu, and W.J. Zhang, "Alternating weighted least squares parameter estimation for biological S-systems," *in Systems Biology (ISB), 2012 IEEE 6th International Conference on*, pp. 6-11, 2012

[3] L.P. Tian, **L.Z. Liu** and F.X. Wu, "State Estimation for Genetic Regulatory Networks," *International Conference on Bioinformatics and Biomedical Engineering (iCBBE 2012)*, 4 pages in IEEE format on disc

[4] B.L. Chen, **L.Z. Liu**, and F.X. Wu, "Inferring stable gene regulatory networks from multiple time course gene expression datasets", *in Systems Biology (ISB), 2011 IEEE 5th International Conference on*, pp. 12-17, 2011

[5] L.P. Tian, **L.Z. Liu**, and F.X. Wu, "Estimating parameters in Genetic regulatory networks with SUM logic" *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pp. 1371-1374, 2011

[6] L.P. Tian, **L.Z. Liu**, and F.X. Wu, "Identification of Pseudo-Periodic Gene Expression Profiles," *BIOCOMP'11*, pp. 42-46, 2011

[7] L.P. Tian, **L.Z. Liu**, and F.X. Wu,"Parameter estimation method for periodical gene identification," *International Conference on Bioinformatics and Biomedical Engineering (iCBBE 2011)*, 4 pages in IEEE format on disc

[8] **L.Z. Liu**, F.X. Wu, L.L. Han, and W.J. Zhang, "Structure identification and parameter estimation of biological S-systems," *in Bioinformatics and Biomedicine (BIBM), 2010 IEEE International Conference on*, pp. 329 334, Dec. 2010

[9] L.P. Tian, **L.Z. Liu**, and F.X. Wu, "Detecting nearly periodical expressed genes from their microarray time-course expression," *IASTED CompBio 2010*, pp. 612-619, 2010

[10] F.X. Wu, **L.Z. Liu**, and Z.H. Xia, "Identification of gene regulatory networks from time course gene expression data," *Engineering in Medicine and Biology Society, EMBC, 2010 Annual International Conference of the IEEE*, pp. 795-798, 2010

[11] L.P. Tian, **L.Z. Liu**, and F.X. Wu, "Parameter estimation method for improper fractional models and its application to molecular biological systems," *Engineering in Medicine and Biology Society, EMBC, 2010 Annual International Conference of the IEEE*, pp. 1477-1480, 2010

Appendix B

Copyright Permissions

JOHN WILEY AND SONS LICENSE TERMS AND CONDITIONS

May 15, 2014

This is a License Agreement between Lizhi Liu ("You") and John Wiley and Sons ("John Wiley and Sons") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by John Wiley and Sons, and the payment terms and conditions.

All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.

License Number	3390330083950
License date	May 15, 2014
Licensed content publisher	John Wiley and Sons
Licensed content publication	WILEY INTERDISCIPLINARY REVIEWS: DATA MINING AND KNOWLEDGE DISCOVERY
Licensed content title	Reverse engineering of gene regulatory networks from biological data
Licensed copyright line	Copyright © 2012 John Wiley & Sons, Inc.
Licensed content author	Li-Zhi Liu,Fang-Xiang Wu,Wen-Jun Zhang
Licensed content date	Aug 17, 2012
Start page	365
End page	385
Type of use	Dissertation/Thesis
Requestor type	Author of this Wiley article
Format	Electronic
Portion	Full article
Will you be translating?	No
Title of your thesis / dissertation	Structure Identification and Parameter Estimation of Biological Systems
Expected completion date	Jul 2014
Expected size (number of pages)	150
Total	0.00 USD
Terms and Conditions	

TERMS AND CONDITIONS

This copyrighted material is owned by or exclusively licensed to John Wiley & Sons, Inc. or one of its group companies (each a"Wiley Company") or handled on behalf of a society with which a Wiley Company has exclusive publishing rights in relation to a particular work (collectively "WILEY"). By clicking accept in connection with completing this licensing transaction, you agree that the following terms and conditions apply to

this transaction (along with the billing and payment terms and conditions established by the Copyright Clearance Center Inc., ("CCC's Billing and Payment terms and conditions"), at the time that you opened your Rightslink account (these are available at any time at http://myaccount.copyright.com).

Terms and Conditions

- The materials you have requested permission to reproduce or reuse (the "Wiley Materials") are protected by copyright.

- You are hereby granted a personal, non-exclusive, non-sub licensable (on a stand-alone basis), non-transferable, worldwide, limited license to reproduce the Wiley Materials for the purpose specified in the licensing process. This license is for a one-time use only and limited to any maximum distribution number specified in the license. The first instance of republication or reuse granted by this licence must be completed within two years of the date of the grant of this licence (although copies prepared before the end date may be distributed thereafter). The Wiley Materials shall not be used in any other manner or for any other purpose, beyond what is granted in the license. Permission is granted subject to an appropriate acknowledgement given to the author, title of the material/book/journal and the publisher. You shall also duplicate the copyright notice that appears in the Wiley publication in your use of the Wiley Material. Permission is also granted on the understanding that nowhere in the text is a previously published source acknowledged for all or part of this Wiley Material. Any third party content is expressly excluded from this permission.

- With respect to the Wiley Materials, all rights are reserved. Except as expressly granted by the terms of the license, no part of the Wiley Materials may be copied, modified, adapted (except for minor reformatting required by the new Publication), translated, reproduced, transferred or distributed, in any form or by any means, and no derivative works may be made based on the Wiley Materials without the prior permission of the respective copyright owner. You may not alter, remove or suppress in any manner any copyright, trademark or other notices displayed by the Wiley Materials. You may not license, rent, sell, loan, lease, pledge, offer as security, transfer or assign the Wiley Materials on a stand-alone basis, or any of the rights granted to you hereunder to any other person.

- The Wiley Materials and all of the intellectual property rights therein shall at all times remain the exclusive property of John Wiley & Sons Inc, the Wiley Companies, or their respective licensors, and your interest therein is only that of having possession of and the right to reproduce the Wiley Materials pursuant to Section 2 herein during the continuance of this Agreement. You agree that you own no right, title or interest in or to the Wiley Materials or any of the intellectual property rights therein. You shall have no rights hereunder other than the license as provided for above in Section 2. No right, license or interest to any trademark, trade name, service

mark or other branding ("Marks") of WILEY or its licensors is granted hereunder, and you agree that you shall not assert any such right, license or interest with respect thereto.

- NEITHER WILEY NOR ITS LICENSORS MAKES ANY WARRANTY OR REPRESENTATION OF ANY KIND TO YOU OR ANY THIRD PARTY, EXPRESS, IMPLIED OR STATUTORY, WITH RESPECT TO THE MATERIALS OR THE ACCURACY OF ANY INFORMATION CONTAINED IN THE MATERIALS, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OF MERCHANTABILITY, ACCURACY, SATISFACTORY QUALITY, FITNESS FOR A PARTICULAR PURPOSE, USABILITY, INTEGRATION OR NON-INFRINGEMENT AND ALL SUCH WARRANTIES ARE HEREBY EXCLUDED BY WILEY AND ITS LICENSORS AND WAIVED BY YOU

- WILEY shall have the right to terminate this Agreement immediately upon breach of this Agreement by you.

- You shall indemnify, defend and hold harmless WILEY, its Licensors and their respective directors, officers, agents and employees, from and against any actual or threatened claims, demands, causes of action or proceedings arising from any breach of this Agreement by you.

- IN NO EVENT SHALL WILEY OR ITS LICENSORS BE LIABLE TO YOU OR ANY OTHER PARTY OR ANY OTHER PERSON OR ENTITY FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, INDIRECT, EXEMPLARY OR PUNITIVE DAMAGES, HOWEVER CAUSED, ARISING OUT OF OR IN CONNECTION WITH THE DOWNLOADING, PROVISIONING, VIEWING OR USE OF THE MATERIALS REGARDLESS OF THE FORM OF ACTION, WHETHER FOR BREACH OF CONTRACT, BREACH OF WARRANTY, TORT, NEGLIGENCE, INFRINGEMENT OR OTHERWISE (INCLUDING, WITHOUT LIMITATION, DAMAGES BASED ON LOSS OF PROFITS, DATA, FILES, USE, BUSINESS OPPORTUNITY OR CLAIMS OF THIRD PARTIES), AND WHETHER OR NOT THE PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION SHALL APPLY NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY PROVIDED HEREIN.

- Should any provision of this Agreement be held by a court of competent jurisdiction to be illegal, invalid, or unenforceable, that provision shall be deemed amended to achieve as nearly as possible the same economic effect as the original provision, and the legality, validity and enforceability of the remaining provisions of this Agreement shall not be affected or impaired thereby.

- The failure of either party to enforce any term or condition of this Agreement shall not constitute a waiver of either party's right to enforce each and every term and condition of this Agreement. No breach under this agreement shall be deemed waived or excused by either party unless such waiver or consent is in writing signed by the party granting such waiver or consent. The waiver by or consent of a party to a breach of any provision of this Agreement shall not operate or be construed as a waiver of or consent to any other or subsequent breach by such other party.

- This Agreement may not be assigned (including by operation of law or otherwise) by you without WILEY's prior written consent.

- Any fee required for this permission shall be non-refundable after thirty (30) days from receipt by the CCC.

- These terms and conditions together with CCC s Billing and Payment terms and conditions (which are incorporated herein) form the entire agreement between you and WILEY concerning this licensing transaction and (in the absence of fraud) supersedes all prior agreements and representations of the parties, oral or written. This Agreement may not be amended except in writing signed by both parties. This Agreement shall be binding upon and inure to the benefit of the parties' successors, legal representatives, and authorized assigns.

- In the event of any conflict between your obligations established by these terms and conditions and those established by CCC s Billing and Payment terms and conditions, these terms and conditions shall prevail.

- WILEY expressly reserves all rights not specifically granted in the combination of (i) the license details provided by you and accepted in the course of this licensing transaction, (ii) these terms and conditions and (iii) CCC s Billing and Payment terms and conditions.

- This Agreement will be void if the Type of Use, Format, Circulation, or Requestor Type was misrepresented during the licensing process.

- This Agreement shall be governed by and construed in accordance with the laws of the State of New York, USA, without regards to such state s conflict of law rules. Any legal action, suit or proceeding arising out of or relating to these Terms and Conditions or the breach thereof shall be instituted in a court of competent jurisdiction in New York County in the State of New York in the United States of America and each party hereby consents and submits to the personal jurisdiction of such court, waives any objection to venue in such court and consents to service of process by registered or certified mail, return receipt requested, at the last known address of such party.

WILEY OPEN ACCESS TERMS AND CONDITIONS

Wiley Publishes Open Access Articles in fully Open Access Journals and in Subscription journals offering Online Open. Although most of the fully Open Access journals publish open access articles under the terms of the Creative Commons Attribution (CC BY) License only, the subscription journals and a few of the Open Access Journals offer a choice of Creative Commons Licenses:: Creative Commons Attribution (CC-BY) license Creative Commons Attribution Non-Commercial (CC-BY-NC) license and Creative Commons Attribution Non-Commercial-NoDerivs (CC-BY-NC-ND) License. The license type is clearly identified on the article.

Copyright in any research article in a journal published as Open Access under a Creative Commons License is retained by the author(s). Authors grant Wiley a license to publish the article and identify itself as the

original publisher. Authors also grant any third party the right to use the article freely as long as its integrity is maintained and its original authors, citation details and publisher are identified as follows: [Title of Article/Author/Journal Title and Volume/Issue. Copyright (c) [year] [copyright owner as specified in the Journal]. Links to the final article on Wiley s website are encouraged where applicable.

The Creative Commons Attribution License

The Creative Commons Attribution License (CC-BY) allows users to copy, distribute and transmit an article, adapt the article and make commercial use of the article. The CC-BY license permits commercial and non-commercial re-use of an open access article, as long as the author is properly attributed.

The Creative Commons Attribution License does not affect the moral rights of authors, including without limitation the right not to have their work subjected to derogatory treatment. It also does not affect any other rights held by authors or third parties in the article, including without limitation the rights of privacy and publicity. Use of the article must not assert or imply, whether implicitly or explicitly, any connection with, endorsement or sponsorship of such use by the author, publisher or any other party associated with the article.

For any reuse or distribution, users must include the copyright notice and make clear to others that the article is made available under a Creative Commons Attribution license, linking to the relevant Creative Commons web page.

To the fullest extent permitted by applicable law, the article is made available as is and without representation or warranties of any kind whether express, implied, statutory or otherwise and including, without limitation, warranties of title, merchantability, fitness for a particular purpose, non-infringement, absence of defects, accuracy, or the presence or absence of errors.

Creative Commons Attribution Non-Commercial License

The Creative Commons Attribution Non-Commercial (CC-BY-NC) License permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.(see below)

Creative Commons Attribution-Non-Commercial-NoDerivs License

The Creative Commons Attribution Non-Commercial-NoDerivs License (CC-BY-NC-ND) permits use, distribution and reproduction in any medium, provided the original work is properly cited, is not used for commercial purposes and no modifications or adaptations are made. (see below)

Use by non-commercial users

For non-commercial and non-promotional purposes, individual users may access, download, copy, display and redistribute to colleagues Wiley Open Access articles, as well as adapt, translate, text- and data-mine the content subject to the following conditions:

- The authors' moral rights are not compromised. These rights include the right of "paternity" (also known as "attribution" - the right for the author to be identified as such) and "integrity" (the right for the author not to have the work altered in such a way that the author's reputation or integrity may be impugned).

- Where content in the article is identified as belonging to a third party, it is the obligation of the user to ensure that any reuse complies with the copyright policies of the owner of that content.

- If article content is copied, downloaded or otherwise reused for non-commercial research and education purposes, a link to the appropriate bibliographic citation (authors, journal, article title, volume, issue, page numbers, DOI and the link to the definitive published version on Wiley Online Library) should be maintained. Copyright notices and disclaimers must not be deleted.

- Any translations, for which a prior translation agreement with Wiley has not been agreed, must prominently display the statement: "This is an unofficial translation of an article that appeared in a Wiley publication. The publisher has not endorsed this translation."

Use by commercial "for-profit" organisations

Use of Wiley Open Access articles for commercial, promotional, or marketing purposes requires further explicit permission from Wiley and will be subject to a fee. Commercial purposes include:

- Copying or downloading of articles, or linking to such articles for further redistribution, sale or licensing;

- Copying, downloading or posting by a site or service that incorporates advertising with such content;

- The inclusion or incorporation of article content in other works or services (other than normal quotations with an appropriate citation) that is then available for sale or licensing, for a fee (for example, a compilation produced for marketing purposes, inclusion in a sales pack)

- Use of article content (other than normal quotations with appropriate citation) by for-profit organisations for promotional purposes

- Linking to article content in e-mails redistributed for promotional, marketing or educational purposes;

- Use for the purposes of monetary reward by means of sale, resale, licence, loan, transfer or other form of commercial exploitation such as marketing products

- Print reprints of Wiley Open Access articles can be purchased from: corporatesales@wiley.com

Further details can be found on Wiley Online Library
http://olabout.wiley.com/WileyCDA/Section/id-410895.html

Other Terms and Conditions:

v1.9

If you would like to pay for this license now, please remit this license along with your payment made payable to "COPYRIGHT CLEARANCE CENTER" otherwise you will be invoiced within 48 hours of the license date. Payment should be in the form of a check or money order referencing your account number and this invoice number 501303987. Once you receive your invoice for this order, you may pay your invoice by credit card. Please follow instructions provided at that time.

Make Payment To:
Copyright Clearance Center
Dept 001
P.O. Box 843006
Boston, MA 02284-3006

For suggestions or comments regarding this order, contact RightsLink Customer Support: customercare@copyright.com or +1-877-622-5543 (toll free in the US) or +1-978-646-2777.

Gratis licenses (referencing $0 in the Total field) are free. Please retain this printable license for your reference. No payment is required.

From: **Boxer,Emma** EBoxer@theiet.org
Subject: RE: Permission to use
Date: May 19, 2014 at 9:57 AM
To: Lizhi.Liu@usask.ca lizhi.liu@usask.ca

Dear Lizhi,

Thank you for your enquiry. Permission to reproduce this work is given, provided that the source of the material (including the author, title, date, and publisher) is acknowledged. A reproduction fee is not due to the IET on this occasion because you are an author of the original work.

If you wish to reproduce the paper as a whole then, as per our post-print policy (http://digital-library.theiet.org/files/pre_postprint_policy.pdf), you will only be able to use the PDF that you received when the updated proof was finalised (i.e. typeset by the IET, but not the final published PDF containing the issue's bibliographic details).

Please feel free to contact me if you have any further queries.

Kind regards,

Emma Boxer
IET Research Journals

From: Lizhi Liu [mailto:Lizhi.Liu@usask.ca]
Sent: Thursday, May 15, 2014 10:07 PM
To: iet_syb
Subject: Permission to use

Dear Editor,

I am Lizhi Liu, the author of the following paper in IET Systems Biology.

L.Z. Liu, F.X. Wu, and W.J. Zhang, "Properties of sparse penalties on inferring gene regulatory networks from time-course gene expression data," IET Systems Biology, April 2014.

I am completing my Ph.D. Dissertation. Could you please give me the permission to use this paper as one of the chapters in the thesis? I appreciate it very much.

Regards,

Lizhi

Copyright

Research articles

- Copyright on any research article in a journal published by BioMed Central is retained by the author(s).
- Authors grant BioMed Central a license to publish the article and identify itself as the original publisher.
- Authors also grant any third party the right to use the article freely as long as its integrity is maintained and its original authors, citation details and publisher are identified.
- Creative Commons Attribution License 4.0 formalizes these and other terms and conditions of publishing research articles.
- In accordance with our Open Data policy, the Creative Commons CC0 1.0 Public Domain Dedication waiver applies to all published data in BioMed Central open access articles

Where an author is prevented from being the copyright holder (for instance in the case of US government employees or those of Commonwealth governments), minor variations may be required. In such cases the copyright line and license statement in individual articles will be adjusted, for example to state '2014 licensee BioMed Central Ltd'.

Other articles

- In addition to fully open access research articles, seven of BioMed Central's journals publish commissioned content which is available by subscription for the first 6 or 12 months (depending on the journal) immediately following publication.
- From January 2014, author(s) of such articles retain copyright but grant BioMed Central an exclusive license to publish and distribute the Article for the initial journal-dependent subscription period after online publication of the Article in the Journal. BioMed Central makes the Article available under the Creative Commons Attribution license after expiry of the initial subscription period, or earlier at BioMed Central's discretion.

Authors' certification

In submitting a research article ('article') to any of the journals published by BioMed Central Ltd ('BioMed Central') authors are requested to certify that:

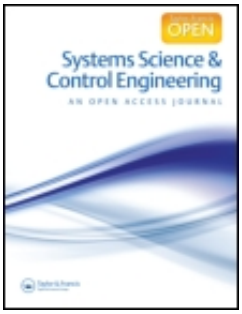
They are authorized by their co-authors to enter into these arrangements.

They warrant, on behalf of themselves and their co-authors, that:

- the article is original, has not been formally published in any other peer-reviewed journal, is not under consideration by any other journal and does not infringe any existing copyright or any other third party rights;
- they are the sole author(s) of the article and have full authority to enter into this agreement and in granting rights to BioMed Central are not in breach of any other obligation. If the law requires that the article be published in the public domain, they will notify BioMed Central at the time of submission;
- the article contains nothing that is unlawful, libellous, or which would, if published, constitute a breach of contract or of confidence or of commitment given to secrecy;
- they have taken due care to ensure the integrity of the article. To their - and currently accepted scientific - knowledge all statements contained in it purporting to be facts are true and any formula or instruction contained in the article will not, if followed accurately, cause any injury, illness or damage to the user.
- they agree to all terms of the Creative Commons Attribution License 4.0 and Open Data policy.

Systems Science & Control Engineering: An Open Access Journal

Volume 2, Issue 1, 2014



Estimating parameters of S-systems by an auxiliary function guided coordinate descent method

DOI:
10.1080/21642583.2014.886800

Li-Zhi Liu[a], Fang-Xiang Wu[a*] & Wen-Jun Zhang[a]

pages 125-134

Publishing models and article dates explained

- Received: 20 Nov 2013
- Accepted: 20 Jan 2014
- Accepted author version posted online: 22 Jan 2014
  Published online: 22 Jan 2014

Abstract

The S-system, a set of nonlinear ordinary differential equations and derived from the generalized mass action law, is an effective model to describe various biological systems. Parameters in S-systems have significant biological meanings, yet difficult to be estimated because of the nonlinearity and complexity of the model. Given time series biological data, its parameter estimation turns out to be a nonlinear optimization problem. A novel method, auxiliary function guided coordinate descent, is proposed in this paper to solve the optimization problem by cyclically optimizing every parameter. In each iteration, only one parameter value is updated and it proves that the objective function keeps nonincreasing during the iterations. The updating rules in each iteration is simple and efficient. Based on this idea, two algorithms are developed to estimate the S-systems for two different constraint situations. The performances of algorithms are studied in several simulation examples. The results demonstrate the effectiveness of the proposed method.

Keywords

- parameter estimation,
- nonlinear programming,
- optimization,
- S\-system,
- coordinate descent

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.