Reverse Engineering of Temporal Gene Expression Data Using Dynamic Bayesian Networks And Evolutionary Search

by

Maryam Salehi

A thesis submitted to the

School of Computing

in conformity with the requirements for

the degree of Master of Science

Queen's University

Kingston, Ontario, Canada

September 2008

Abstract

Capturing the mechanism of gene regulation in a living cell is essential to predict the behavior of cell in response to intercellular or extra cellular factors. Such prediction capability can potentially lead to development of improved diagnostic tests and therapeutics [21]. Amongst reverse engineering approaches that aim to model gene regulation are Dynamic Bayesian Networks (DBNs). DBNs are of particular interest as these models are capable of discovering the causal relationships between genes while dealing with noisy gene expression data. At the same time, the problem of discovering the optimum DBN model, makes structure learning of DBN a challenging topic. This is mainly due to the high dimensionality of the search space of gene expression data that makes exhaustive search strategies for identifying the best DBN structure, not practical.

In this work, for the first time the application of a covariance-based evolutionary search algorithm is proposed for structure learning of DBNs. In addition, the convergence time of the proposed algorithm is improved compared to the previously reported covariance-based evolutionary search approaches. This is achieved by keeping a fixed number of good sample solutions from previous iterations. Finally, the proposed approach, M-CMA-ES, unlike gradient-based methods has a high probability to converge to a global optimum.

To assess how efficient this approach works, a temporal synthetic dataset is developed. The proposed approach is then applied to this dataset as well as Brainsim dataset, a well known simulated temporal gene expression data [58]. The results indicate that the proposed method is quite efficient in reconstructing the networks in both the synthetic and Brainsim datasets. Furthermore, it outperforms other algorithms in terms of both the predicted structure accuracy and the mean square error of the reconstructed time series of gene expression data.

For validation purposes, the proposed approach is also applied to a biological dataset composed of 14 cell-cycle regulated genes in yeast *Saccharomyces Cerevisiae*. Considering the KEGG[1] pathway as the target network, the efficiency of the proposed reverse engineering approach significantly improves on the results of two previous studies of yeast cell cycle data in terms of capturing the correct interactions.

[1]Kyoto Encyclopedia of Genes and Genomes

Acknowledgments

This thesis would not transpire to exist without the help, support and love of a number of people to whom I will always be indebted. First and foremost, I would like to express my sincere gratitude to my supervisor Professor Parvin Mousavi for her valuable guidance, encouragement, technical advice and financial support. I would like to thank Professor Alan Ableson for his advice and thoughtful comments on the optimization algorithm, Professor Paul Young for his illuminating explanations on Yeast cell cycle pathway and Drs. Larry D. Greller and Sergio E. Baranzini for their valuable feedback and discussions. I would also like to thank Amir Dehghani and Mehdi Moradi for their friendship and support. Last, but certainly not least, I wish to thank my dearest parents and family, who I am indebted to, for their unconditional support and endless love.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Reverse Engineering of Gene Regulatory Networks

Every cell in a living organism contains the same copy of genetic material (DNA). However, at any time, in a particular cell, only a fraction of the proteins coded for in the DNA are produced. Proteins are responsible for carrying out most cellular processes and may act as structural elements, enzyme catalysts, antibodies, and so on [31]. The variability in the proteins synthesized in different cells results in different cell types in a multicellular organism. Furthermore, the type and the amount of proteins produced in a cell are essential in order for the cell to function properly and must be precisely regulated. The proteins synthesized in a cell might change in response to different factors ranging from normal development to repairing damage to the cell. This ability of cells to change the proteins is essential for a living organism as it increases the adaptability of the organism [31].

The regulation of protein production is a complex process controlled by a collection of proteins called transcription factors (TF). These proteins influence the transcription of particular genes by binding to specific parts of the DNA and determining when, and how much those particular genes are expressed. Transcription factors can affect the expression of a single gene or a large number of genes simultaneously. As regulatory proteins are themselves the products of expressed genes, they are under regulatory control and comprise gene regulatory networks [63].

Reconstructing gene regulatory networks is important as it provides the basis for the dynamical analysis of gene regulatory interactions. As such, by now, various techniques known as 'reverse engineering' methods have been developed for inferring these networks out of temporal gene expression profiles. Temporal gene expression profiles are gene expression values measured, based on the concentration of their corresponding mRNA, under various conditions and at a number of time points. The aim of reverse engineering techniques is to capture the pattern of regulatory excitations and inhibitations amongst a set of genes and reconstruct their underlying genetic network.

1.2 Motivation

Reverse engineering of gene regulatory networks is of particular interest in systems biology. Such interest arises by the capability of these models in describing the dynamical behavior of gene interaction networks. The study of gene regulatory networks can provide useful information about the pathway to which a gene belongs and the genes it interacts with. Furthermore, it describes the gene function in terms of how it affects other genes and indicates which genes are pathway initiators and therefore

potential drug targets [63]. Understanding gene regulatory networks is also an essential step to predict the behavior of cells in response to intercellular or extra cellular factors. Such prediction capability may potentially lead to development of improved diagnostic tests and therapeutics [21]. In addition, these networks help in understanding the complex mechanisms of development and evolution in living organisms. [14].

As such, by now, many different reverse engineering techniques have been developed for modeling of gene regulatory interactions. Given the fact that measurement noise exists in all time series expression experiments, among various reverse engineering techniques, probabilistic methods such as Bayesian Networks (BNs) and specially Dynamic Bayesian Networks(DBNs) are of particular interest. DBNs are suitable models when dealing with time series microarray data. DBNs are able to extract casual relationships between genes by relying on the temporal nature of the data. In addition, unlike BNs which are acyclic, DBNs allow for cycles which are very common in many biological systems. DBNs are directed graphical models that provide stochastic description of gene associations; they use probabilities to represent the uncertainty inherent in the measurements of gene expression data [16].

Learning the structure of a DBN which requires searching for an optimal structure, is an NP-hard problem. This is mainly due to the fact that the number of possible structures for DBNs in a given problem grows exponentially with respect to the number of genes [36]. As such, exhaustive approaches for learning DBN structure are computationally expensive or even not practical. Alternative heuristic approaches have been reported in the literature to reduce the computational time of learning DBN structure [52].

Previous heuristic methods for structure learning of DBN have mostly used gradient-based approaches (e.g. Hill Climbing) [29] or Markov Chain Monte Carlo based approaches (e.g. Simulated Annealing) [29]. Other approaches address the problem of structure learning of DBN by imposing a number of simplifying assumptions to limit the search space (e.g. K2; K2GA) [11, 28]. Gradient based approaches converge to different local optima depending on the starting search point, while Markov Chain Monte Carlo methods require a high computational time to find a solution close to the optimum. The efficiency of methods with simplifying assumptions depends on how these assumptions affect the accuracy of identified solution. Detailed descriptions of some of the above methods are presented in chapter 2.

1.3 Problem Statement and Objective

Capturing the pattern of regulatory interactions out of expression profiles of a set of genes is an important but challenging area of research in systems biology. An efficient and reliable reverse engineering technique must be insensitive to noisy data. All microarray expression data are inherently noisy. The existing noise is either systematic or random. Systematic noise is caused by systematic experimental errors and unlike random noise, depends on samples, microarray spots, or DNA sequences. This kind of noise can be partially reduced by different normalization techniques. Reducing random noise requires improvement in microarray technology, microarray image processing and data extraction techniques [30]. Furthermore, A reliable reverse engineering approach should be able to capture causal interactions among genes to handle cyclic networks to gene regulatory pathways [41].

In this work, a powerful reverse engineering approach is proposed based on DBNs

and a heuristic evolutionary search strategy to search for the best DBN structure from gene expression data. We name our approach 'Covariance Matrix Adaptation Evolutionary Strategy with Explicit Memory, or M-CMA-ES.

DBN is considered for our network inference approach as it is a directed graphical model capable of representing casual relationships among genes by relying on the temporal nature of the expression data. To avoid the problems usually caused by discretizing expression values (e.g. missing useful information), a continuous version of DBN is implemented. As genes connect to each other through non-linear relationships, in our implementation of DBN, non-parametric regression model is used for representing gene associations.

Among various heuristic search strategies for structure learning of DBN, Hill climbing is the most commonly used algorithm. However it is not very efficient as it is prone to getting trapped in local optima and the learned network is not very accurate. To reconstruct a more accurate model, an evolutionary search strategy is developed based on CMA-ES algorithm. This method outperforms many heuristic optimization algorithms (e.g. gradient-based methods) as it is less sensitive to outliers, noise, and local optima. CMA-ES unlike Hill Climbing has high probability to converge to global optimum, but it requires more computational time. To improve the computational time of the original CMA-ES algorithm, M-CMA-ES is proposed which has a faster convergence time.

M-CMA-ES is an iterative strategy that guarantees the best generated sample solution at each iteration is at least as optimum as that of previous iteration while the size of the generated candidate solutions does not change through iterations. M-CMA-ES is achieved by modifying the Covariance Matrix Adaptation Evolution Strategy

(CMA-ES) first proposed in [25] as a heuristic approach for parameter optimization of non-linear objective functions [55].

M-CMA-ES outperforms gradient-based methods as it is less prone to get trapped into a local optimum. It also has a short convergence time compared to the original CMA-ES.

1.4 Contributions

The primary contributions of the work are reported here:

- Presenting an iterative search strategy, M-CMA-ES for learning the structure of DBNs. To the best of the author's knowledge, covariance matrix based heuristic search strategies have never been used for this purpose in the literature.
- Proposing a solution for the fast convergence to the defined network model by guaranteing that the best generated candidate solution at each iteration is at least as optimum as that of previous iteration while the size of the generated population does not change through iterations.
- Modeling a time-delayed gene regulatory pathway of arbitrary structure. A temporal artificial dataset with the known pathway is then developed by following a stochastic formula. The generated dataset is used to evaluate and compare the efficiency of our purposed M-CMA-ES with hill climbing for discovering the best structure of DBN.
- validating DBNs and M-CMA-ES on Brainsim dataset, a simulated temporal gene expression dataset, representing gene interactions in response to the singing

behavior in a song bird.

- Applying DBN with M-CMA-ES to 14 cell cycle regulated genes in the yeast *Saccharomyces cerevisiae*. Evaluating the resulting interactions by comparing to the KEGG pathway as the target network as well as previous reverse engineering studies of yeast cell cycle data.

1.5 Organization of Thesis

In chapter 2, the basic biological concepts underlying gene regulatory networks are introduced. The chapter also includes a brief description of microarray expression data and preprocessing approaches. It provides a review of probabilistic approaches to reverse engineering of gene regulatory networks along with heuristic search strategies used for their structure learning.

In Chapter 3, the datasets used for evaluating and validating the proposed approaches are introduced. Preprocessing of each dataset is also separately described. In chapter 4, a detailed description of the reverse engineering approach and proposed methodology, M-CMA-ES, for structure learning of DBN is provided. The evaluation criteria used for validation is also introduced.

In chapter 5, the details of implementing the network inference approach are presented as well as the results obtained for each dataset. Conclusions and the future work are presented in Chapter 6.

Chapter 2

Background

2.1 Basic Concepts In Molecular Biology

All living organisms consist of basic functional units called cells. Each cell is a complex system composed of different building blocks whose main substances fall into one of four major classes of biological molecules: carbohydrates, lipids, proteins, and nucleic acids (e.g. Deoxyribo Nucleic acid or DNA). These classes of biological molecules are called macromolecules due to their large size compared to most other inorganic molecules. Along with macromolecules, water is also an important element of every cell and has a critical role in the transmission of signals between molecules [6, 64].

Among the above-mentioned biological molecules, nucleic acids, containing the genetic material, specify the structural and physiological characteristics of the cells in organisms [43].

Every cell belongs to one of the two main cell types: prokaryotes and eukaryotes. Prokaryotic cells are usually specified by their small size and their simple structure compared to complex Eukaryotic cells [15]. Eukaryotic cells are mostly found in

multicellular organisms known as eukaryotes including animals and plants. However, Brewer's yeast is a single-celled eukaryote. Prokaryotic cells are independent and by themselves comprise single-celled organisms named prokaryotes. Prokaryotes usually lack a cell nucleus [32]. Apart from eukaryotes and prokaryotes, there is another group of organisms called archaea which are single-celled but more similar to eukaryotes than prokaryotes [32].

Multi-cellular organisms are composed of many different types of cells such as blood, skin, and nerve cells. Each of these cell types is specialized to accomplish a particular task; This is known as cell differentiation. In multi-cellular organisms, different classes of cells can work together and accomplish many complex tasks that single cells can not [32].

All cells of multi-cellular organisms seek to replicate themselves. Eukaryotic cell replication is a process by which cells make a copy of their genetic material and then divide into two daughter cells. The series of events that lead to the replication of a eukaryotic cell is referred to as cell cycle (Figure 2.1). [15]. There are two periods in

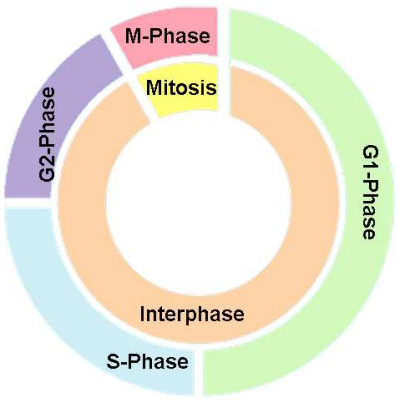

Figure 2.1: Phases of the eukaryotic cell cycle

each cell cycle, interphase and mitotic (M) phase. During interphase, the cell grows, duplicates its DNA and gets ready for mitosis. Interphase is a period between two subsequent cell divisions. During M-phase, the cell divides into two distinct cells called daughter cells. M-phase consists of two main processes: (1) nuclear division in which the cell's chromosomes are separated into two parts and (2) cytoplasmic division where the cell divides into two daughter cells. Activation of each phase is dependent on the proper completion of the previous one [14].

Despite the differences among cells of a multicellular organism, they all contain cytoplasm and genetic material and have the basic mechanisms for translating genetic messages into the protein. Proteins are the fundamental structural and functional units in cells and can act as structural elements, enzyme catalysts, and antibodies [31].

Eukaryotic cells have a variety of internal compartments or organelles where each organelle accomplishes a specific function. Nucleus, the largest organelle in the cell, contains the genetic material (DNA)[1] which is inherited through generations. In prokaryotic cells, DNA is found directly in the cytoplasm [6].

DNA is composed of two complementary strands of nucleotides shaped in a double helix structure (Figure 2.2). These two strands are called complementary as the sequence of nucleotides in one of the strands can be completely determined based on the sequence of nucleotides in the other strand. Each nucleotide consists of a sugar molecule, one or more phosphate groups and one of four nitrogenous bases adenine (A), guanine (G), cytosine (C) and thymine (T) [31]. Adenines in one strand held together by hydrogen bonds with thymines in the other strand (A-T). In a similar way,

[1] A possible exception is a group of viruses that have RNA genomes, but viruses are not normally considered living organisms (Wikipedia)

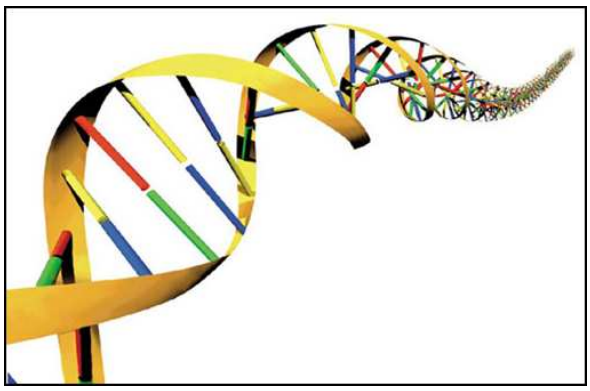

Figure 2.2: deoxyribonucleic acids (DNA).

guanines in one strand bond with cytosines (G-C) in the other strand. The structure of proteins is determined by the sequence of nucleotides in DNA. A segment of DNA that contains all the information necessary to code for a protein, is called a gene.

Proteins are made of amino acids arranged in a linear chain and joined together by peptide bonds. There are 20 naturally occurring amino acids and a combination of four types of nucleotides that have to code for them. As such at least three nucleotides[1] are required to code for an amino acid.

According to central dogma of biology, the information coded by DNA of a given gene is transcribed into a RNA molecule[2], called messenger RNA (mRNA) and translated into an amino acid chain which comprises a protein (Figure 2.3).

The process of protein production consists of three main steps: transcription, splicing and translation. In the first step, A portion of DNA is transcribed into a mRNA molecule. An RNA polymerase enzyme binds to a specific location of DNA molecule and determines which strand of DNA and in which direction will

[1]Nucleotide triplets are called codons.
[2]RNA is a single stranded molecule with a similar structure to DNA

be transcribed. In the second step, the coding regions of mRNA [3] splice together. In other words, the the non-coding regions of mRNA splice out and mRNA changes into the mature mRNA. In the last step, the translation of mature mRNA into proteins, the codons of mature mRNA map into amino acids via RNA molecules called transfer RNA(tRNA) [31].

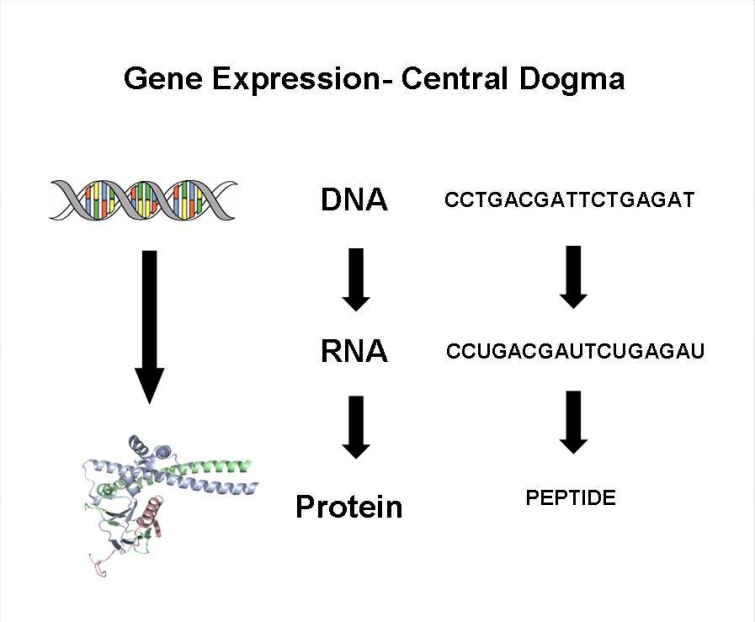

Figure 2.3: Protein production from DNA in two stages transcription and translation.

2.1.1 Transcriptional Regulatory Networks

All the cells in a multicellular organism contain exactly the same copy of DNA. However, different portions of genes are active at different cells. Therefore, at any

[3]DNA and mRNA (Not mature mRNA)consist of two kinds of regions, coding regions(exons) and non-coding regions (introns)

time, a particular cell produces only a portion of the proteins coded for in its DNA. In fact, the difference between the characteristics of various types of cells comes from the difference in their gene expressions. The type and the amount of protein produced in each particular cell are extremely important for the cell to properly function. In response to various environmental situations, the amount of proteins produced in a cell might change [31].

The regulation of protein production is a complex process controlled by a collection of proteins called transcription factors (TF). These proteins determine when, and how much a particular gene is expressed. Since regulatory proteins are themselves the products of expressed genes, they are under regulatory control and comprise complex interaction networks known as Gene Regulatory Networks (GRN) [63]. GRNs contain information about the pathway to which a gene belongs and the genes it interacts with. Furthermore, it describes the gene function in terms of how it affects other genes and indicates which genes are pathway initiators and therefore potential drug targets [63].

2.2 Microarrays Gene Expression Measurement

Recent advances in high throughout gene expression measurement technologies, allow us to monitor expression level of thousands of genes simultaneously. Microarrays, a collection of single stranded DNA segments deposited or synthesized on a solid surface, measure the mRNA abundance of genes in a high throughout fashion. The single stranded DNA segments are called probes and are complementary to specific RNA species in the cell [21, 30]. The amount of mRNA is proportional to the transcription rate of its corresponding genes. Therefore, the relative transcription rate of genes can

be estimated by measuring their corresponding mRNA levels [33].

There are several microarray technologies and multiple ways to categorize these technologies. One approach is to categorize microarrays according to the type of probes they use. Based on this, microarrays can be divided in two groups: cDNA microarrays and oligonucleotide microarrays.

2.2.1 cDNA microarrays

cDNA microarray is a widely used microarray technology in which two samples are usually analyzed simultaneously in a comparative fashion. "In cDNA microarray technology, probes of DNA are spotted onto a glass slide by a robot [33]."

cDNA microarrays

Measuring expression levels of genes from a cDNA microarray, contains the following steps (Figure 2.4).

1. Building arrays
2. Extracting mRNA from an experimental sample and a reference sample.
3. Synthesizing more stable Complementary DNA (cDNA)[1] from both mRNA samples by using the enzyme reverse transcriptase.
4. Labeling cDNA with fluorescent markers. Since mRNA is degraded easily, labeling cDNA is more convenient than mRNA. One of the samples is labeled by the fluorescent dye, cy3 (green), and the other sample by cy5 (red).
5. Mixing two fluorescently labeled samples and hybridizing the mixture to microarrays.

[1] cDNA is a single-stranded DNA.

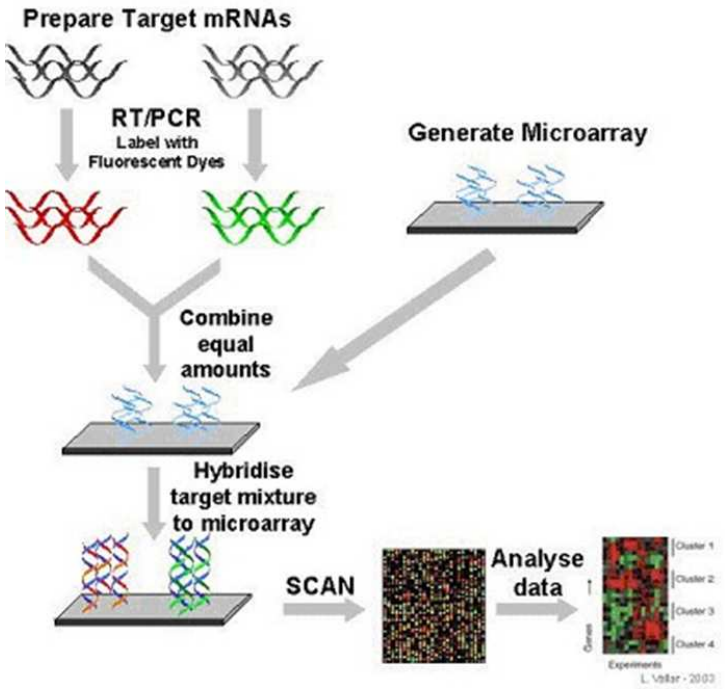

Figure 2.4: A cDNA microarray experiment. Picture is taken by permission from http://www.microarray.lu

6. Purification of the labeled products.

7. Scanning the microarrays and analyzing the resulting images.

During the image analysis stage, the abundance of mRNA on the spots are measured and represented numerically [30]. The gene expression levels measured at various conditions or over time are usually represented as gene expression matrices where rows and columns correspond to genes and conditions, respectively. Each entry (i,j) of these matrices indicates the expression value of a given gene, i, in a given condition, j. Different reverse engineering techniques are applied to these expression matrices to infer a model of gene regulatory interactions underlying the gene expression data [43, 33].

2.2.2 Oligonucleotide microarrays

In these microarrays, genes are represented by a set of 14 to 20 short sequences of DNA, called oligonucleotides. Affymetrix (Santa Clara, CA), is a provider of a commonly used oligonucleotide array. In this technology, each oligonucleotide sequence is represented by two probes, called (1)Perfect Match or PM and (2) Miss Match or MM. The DNA sequences in every pair of PM and MM are identical except for one nucleotide in the center of the sequences. PM is the exact sequence of the chosen fragment of the gene. In this approach, an incorrect hybridization affects both PM and MM while, the the correct gene will only hybridize to the PM. The expression level of each gene is the average difference between PM and MM [18].

Following are the steps of measuring gene expression levels from a oligonucleotide microarray [33]:

1. Building oligonucleotide arrays onto a chip via a chemical process.

2. Converting mRNA to fluorescently labeled cDNA.

3. Hybridizing the labeled cDNA samples to microarray.

4. Scanning the microarray.

5. Summarizing the values of PM and MM into an expression value for each gene.

2.3 Microarray Data Preprocessing

In order to make microarray data more reliable for later analysis, one requires to perform some data preprocessing. Microarray data preprocessing methods include the following:

- Missing Value Management

  Many gene expression analysis methods such as multivariate statistical analysis techniques[5] require a complete matrix of expression values and can not be applied to data with missing values.

  There are various reasons that might cause missing values in microarray data such as "image corruption, dust or scratches on the slides, experimental error during the laboratory process," or unreliable readings for particular genes [39]. As repeating microarray experiments is costly and time consuming, several techniques have been developed to estimate the missing values [39]. These span from simple averaging approaches to more complex techniques such as k-nearest neighbors imputation [62].

[5]multivariate statistical analysis techniques such as principal component analysis (PCA)and singular value decomposition(SVD)

- Data Normalization

  In order to compare gene expression profiles using particular similarity measures (e.g. "Euclidean distance"), data manipulation is necessary. Data re-scaling is sometimes necessary to expand or compress gene expression profiles to the same scale (0-1). For example, "Euclidean distance" identifies similar genes based on "their intensity of expression rather than their similarity in the geometry of the profile" [17]. As such, data scaling is necessary to achieve accurate distances of genes [17].

- Gene Filtering

  In every microarray experiment, the majority of genes, having constant profile, do not convey any important information. Those genes are not effective features in microarray analysis and only decrease the efficiency and increase the computational time of the analysis. As such, several approaches are developed for filtering constant genes and determining significant genes; Using fold change analysis, significant genes can be determined based on relative increase or decrease in their expression profiles [50]. Considering a small threshold value for the variance of genes with almost constant profiles is also a simple method for constant gene identification. Furthermore, statistical-based methods such as t-test are also used for detecting differentially expressed genes [13].

- Interpolation

  Due to time consuming and costly nature of microarray experiments, gene expression datasets usually contain less number of samples than genes. Generally speaking, in a pattern recognition problem, if the number of samples is less

than the number of features (genes), there is always the probability of finding several solutions that best fit the data. Interpolation address this problem by increasing the number of samples while adding new data points within the range of original measurements.

2.4 Network Inference Approaches

Given the gene expression data under various conditions and over time, a model of the gene interactions can be developed through different reverse engineering techniques [37]. The development of reverse engineering methods is a challenging area of research. Challenges arise from the noisy nature of microarray data as well as their high dimensionality.

Prior to reverse engineering, the dimension of the dataset must be greatly reduced to involve a set of genes which are biologically relevant without ignoring influential genes in the biological process being analyzed [45].

There are many different dimension reduction techniques that have been previously applied in the context of inferring genetic regulatory networks [45]. The dimensional reduction techniques can be divided into two groups; unsupervised and supervised. The goal of unsupervised dimension reduction methods is to eliminate numerically redundant genes in a dataset. These techniques combine co-expressed genes and consider them as meta-genes which are then used to reverse engineering regulatory networks. In supervised dimension reduction techniques, functional knowledge is used to select genes that are biologically relevant to the study being performed [45].

Many reverse engineering techniques have been applied to the problem of inferring genetic interactions. These include Artificial neural networks [38, 44], genetic networks [34], Linear Differential Equations [55], Boolean Networks [48], and probabilistic graphical models [65, 2]. In this thesis, the focus is on probabilistic graphical models in particular DBNs.

2.4.1 Probabilistic Graphical Models

Among Reverse engineering techniques, graphical models are promising tools for analysis of gene interactions as they allow for the stochastic description of complex gene associations and interdependencies [57].

Probabilistic graphical models are a combination of probability theory and graph theory. In these models, probability is used to model uncertainty in data. These models are simply models of joint distributions of a set of variables assuming a random sampling paradigm [16].

The application of probabilistic graphical models in systems biology is strongly limited by the amount of experimental data [57]. In a typical microarray dataset, the number of genes, is much more than the number of different conditions or time points. This creates serious challenges to any statistical inference procedure [56].

So far, different graphical models such as relevance networks (RNs), Graphical Gaussian models (GGMs), graphical chain models, Bayesian networks (BNs), and Dynamic Bayesian Networks (DBN) have been applied to infer gene regulatory interactions.

Relevance Networks

A simple method for inferring the network of linear dependencies among a set of genes is to compute the Pearson correlation coefficient between any pair of genes. If the pairwise correlation coefficient exceeds a specified threshold, then an edge is drawn between the corresponding genes. The resulting graph is called a "relevance network" where missing edges indicate marginal independence[1][65]. The main problem with this approach is that a high correlation coefficient between two genes may be indicative of a direct interaction (Figure 2.5-a) or indirect interaction (Figure 2.5-b,c,d). However, in learning a genetic network, it is important to be able to distinguish between these alternatives.

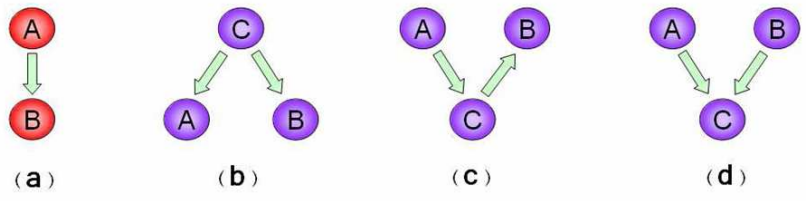

Figure 2.5: (a) Direct interaction(b) Regulation of two genes by a common regulator(c) signaling chain via intermediate regulator(d)Co-regulating a node by two regulators.

Bickel [5] generalizes relevance networks to time series of gene expression data by introducing a time-lag for correlation and applies it to 4489 genes in yeast cell cycle data whose expression profile contains no missing values. Butte et al. [8] enhance the correlation networks by using a more flexible (non-linear) similarity measure, pairwise

[1]The marginal independence between two random variables X_a, X_b is depicted as $X_a \perp X_b$. The value of $X_a \perp X_b$ indicates the degree of linear independency between two variables X_a and X_b.

mutual information. his technique is used on a public data set of 79 RNA expression measurements of 2,467 genes to construct 22 relevance networks.

Graphical Gaussian models

Graphical Gaussian models also known as "covariance selection models" are among popular tools to study gene association networks[57]. The concept of conditional independence is fundamental to graphical Gaussian modeling, i.e. the conditional independence structure of the data is characterized by a conditional independence graph. In this graph, each variable is represented by a node and two nodes are connected by an edge if there is a direct association between them. In other words, two vertices are not connected if they are conditionally independent, given all of the other variables. The conditional independence between two variables X_a and X_b given X_c is depicted as $X_a \perp X_b \mid X_c$. The value of $X_a \perp X_b \mid X_c$ indicates that X_a and X_b are independent, given the variable X_c. [19].

The key idea behind GGMs is to use partial correlations as a measure of conditional (in) dependence between any two variables. "partial correlation measures the degree of association between two random variables, with the effect of a set of controlling random variables removed"[3]. This makes it straightforward to distinguish direct from indirect interactions [57]. Partial correlation coefficient provide a strong measure of dependence and correspondingly, offers only a weak criterion of independence. In addition, some very weakly correlated genes, but highly related in terms of partial correlation in the context of the other genes may not be found by correlation based methods [16].

GGMs contain only undirected edges; this makes them on the one hand, conceptually simpler and on the other hand more widely applicable [57]. In their framework, the strength of direct pair wise correlation is characterized by the partial correlation matrix. Each entry of this matrix describes the correlation between any two genes i and j, given all other genes [21].

Standard graphical modeling theory indicates that partial correlation is related to the inverse of correlation matrix. This leads to a simple procedure for computing the partial correlation matrix ρ_{ij} using the following formulas:

$$\omega = {p_{ij}}^{-1} \tag{2.1}$$

$$\rho_{ij} = \frac{\omega_{ij}}{\sqrt{\omega_{ii}\omega_{jj}}} \tag{2.2}$$

In the inverse step 2.2, it is valid to use the covariance matrix σ_{ij} instead of the correlation matrix p_{ij} [56].

It should be noted that, classical GGM theory [57] may only be applied when n (the number of samples) $\gg$p (the number of genes), because otherwise the sample covariance and correlation matrices are not well conditioned, which in turn prevents the computation of partial correlations [19].

One of the simplest approaches to deal with the problem of high dimensionality is to assess the gene interactions among either a small subset of genes or a small number of clusters of genes (the number of selected genes or clusters has to be less than the number of samples). This strategy has the following limitations:

- Choosing a small subset of genes poses the risks of excluding important genes from the analysis.

- Selecting genes for inclusion in the dataset is challenging.
- The conditional dependence properties among clusters are somewhat meaningless, because not all the genes of one cluster interact with all the genes of other clusters.
- The strength of the association on the gene level is lost, because only clusters of genes are considered.

One way to deal with the problem of high dimensionality is to compute partial correlation coefficients of limited order [1]. This results in something between a full GGM that computes the correlation between every 2 gene given all the remaining genes and a relevance network model with unconditioned correlation. De la Fuente et al (2004) propose to calculate partial correlation coefficients up to second order only [57], i.e., the correlation of each pair of genes is computed with considering the effects of two other genes at most. Kishino and Waddell [42] address this problem by proposing a method for gene selection; In the proposed method, very small partial correlation coefficients are set to zero. Several strategies based on first order conditional dependencies are also employed by Wille et al. (2004), Wille and Buhlmanm(2005), and Magwenn and Kim(2004)[57].

Graphical Chain models

Graphical chain models are probabilistic graphical models in which the independence of the structure is represented by a graph [2]. In this model, the set of variables (genes)

[1] Order of a partial correlation indicates the number of control variables. For instance, A "second order partial correlation" is one with two control variables and a "zero-order correlation" is one with no control variable which is a simple correlation coefficient.

are partitioned into disjoint subsets from prior biological knowledge or numerically. These partitions are known as blocks. The edges within blocks are undirected, representing non-causal associations, and the edges between blocks are arrows pointing from blocks with lower index numbers to those with higher indices. These directed associations are assumed to be potentially causal (Figure 2.6).

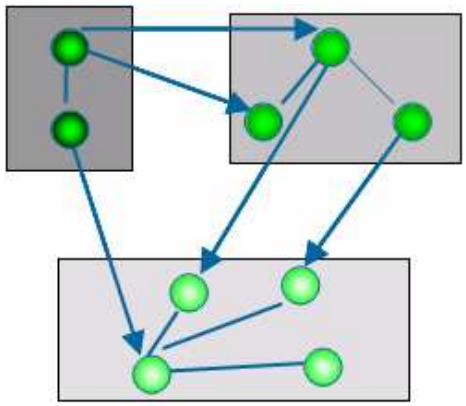

Figure 2.6: Graphical Chain models. In this figure, rectangular is an indicative of block and every circle represents a gene.

The absence of a line or arrow between two variables in the graph indicates that there is no direct association between those variables, i.e., the variables are independent after controlling for all of the other variables in the same and previous blocks [2]. To fit a graphical chain model, one needs to (1) partition the variables into a number of ordered blocks and (2) determine the significant direct associations between every pair of variables in each block given all of the other variables in the same and previous blocks [2]. The associations of the genes between and within blocks are inferred by partial correlation measure as in GGMs.

Aburatania et al. [2] develop a procedure for graphical chain modeling to analyze expression profiles of a number of cell cycle regulated genes in yeast which can be partitioned into several blocks in a natural order.

Bayesian Networks

BNs are directed acyclic graphical models representing the joint distribution over a set of random variables. These models consist of two components, a directed acyclic graph and a set of parameters of conditional distribution of each variable given the rest of variables. In the graphical structure of Bayesian Networks, nodes are representative of random variables and edges correspond to possible dependencies between variables (Figure 2.7). The absence of an edge between two genes means that those genes are conditionally independent given the rest of genes.

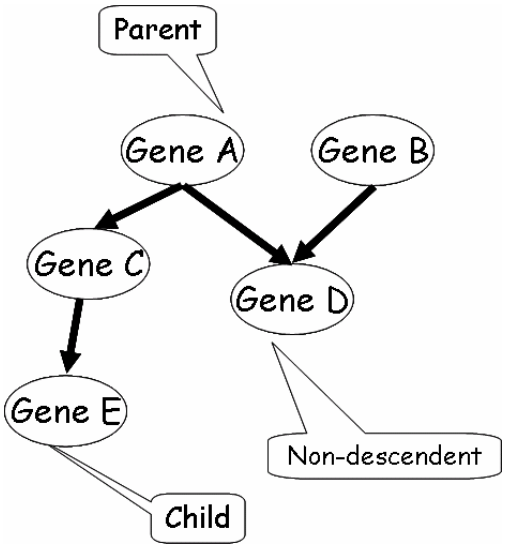

Figure 2.7: Bayesian Network Model; In this figure, gene C is independent of gene D given gene A.

The BNs follow the markov assumption that each variable is independent of its non-descendants given its parents. Therefore the joint distribution over the set of variables can be decomposed into the product form of probability of each variable given its parents. The first step in constructing the BNs, using score-based approaches, is determining a scoring function based on the posterior probability of BN given the data. The scoring function is then used as a criterion for finding an optimal set of regulators for each variable. There are several heuristic search algorithms such as greedy hill climbing and simulated annealing for finding the best set of regulators for each variable [10, 60]. Leray et al. [47] present an approach for learning BN based on the Expectation Maximization algorithm known as EM-algorithm [68]. The proposed approach improves the computation time of EM-algorithm and is able to infer BNs from incomplete data. In this study, expected statistics, computed by a parameter EM algorithm, used to complete the incomplete data. Nagele et al. [52] present an iterative algorithm called "Substructure learning" to estimate the structure of BNs by learning small sub networks. "This method reduces the complexity of learning large networks by splitting the task into several small subtasks" [52].

Dynamic Bayesian Networks (DBNs)

DBNs unlike BNs, use time series of gene expression data for constructing causal relationships among genes. Similar to BNs, these networks satisfy the first order Markov assumption. Based on this assumption, the parents (regulators) of each gene are chosen using information derived from gene expression at the same or previous time points. In addition, a gene is assumed to be independent of its non-descendants

given its parents. As a result, the graphical structure of DBNs only represents direct associations between genes. Furthermore, the Markov assumption prevents the network from getting unnecessarily complex.

Current methods for learning of DBN can be categorized into two major groups, constraint based methods [22] and score based methods [46, 27, 11]. The former, determine conditional independencies and dependencies between genes based on a number of statistical tests such as Pearson, Chi-square, and Mutual Information [9, 22]. Constraint-based methods are shown to create satisfactory results with sparse networks but are not suitable for large datasets and dense networks [9, 22]. The latter, score based methods, consider learning of DBN as an optimization problem. These methods devise a scoring function for a candidate network structure based on the probability of the structure given the gene expression data. They search through the space of all possible network structures that minimize the scoring function [67, 46, 27, 11].

Zou et al. [69] proposed a DBN-based approach with improved computational time and accuracy. The computational time of the algorithm was improved by reducing the search space; this is achieved by limiting the number of potential regulators to those genes whose transcription level changes earlier or at the same time as the transcription level of their target genes. The transcriptional time lag between every target gene and its regulators was estimated based on the time difference between the initial change in their expression profile. Berkman et al. [4] introduce DBNs inferred by weak learning approaches. These DBNs were referred to as WDBNs. The goal of building WDBNs is to increase the probability of converging to different local optima. Weak learning approaches produce a set of independent DBNs. DBNs are then combined into one

result. In order to build WDBNs, different learning algorithms or a learning algorithm with different parameters can be used. These networks can also be produced from data that is manipulated in different ways.

DBNs are known to have several applications in network reconstruction [69, 54]. Sometimes, they are considered to include hidden nodes which can describe transcription factor activity or any other kind of environmental effects in the cell [51]. In summary, DBNs provide a flexible frameworks for modeling gene regulatory networks compared to other methods [49].

2.5 Heuristic Search Strategies for Network Structure Learning

Learning the structure of a BN or DBN which requires searching for an optimal structure, is an NP-hard problem. This is mainly due to the fact that the number of possible structures for these models in a given problem grows exponentially with respect to the number of genes [36]. As such, exhaustive approaches for learning BN or DBN structure are computationally expensive or even not practical. Alternative heuristic approaches such as greedy hill climbing [29] and Simulated Annealing [29, 35] have been reported in the literature to reduce the computational time of learning these structures [52].

greedy hill climbing [29, 52] is a relatively fast local search strategy that extensively used in the literature for learning the structure of both BNs and DBNs. The drawback of this technique is that the search process might get trapped in a local minimum and the algorithm mostly is not able to find the optimum solution[29]. Janura et al.

[35] used a stochastic optimization algorithm, simulated annealing, which is based on the Markov Chain Monte Carlo approach to learn BNs from statistical data. In order to use this algorithm, the problem of learning BNs was reformulated as a discrete optimization problem.

Previous heuristic methods for structure learning of DBN have mostly used gradient-based approaches (eg. Hill Climbing) or Markov Chain Monte Carlo based approaches (eg. Simulated Annealing). Gradient based approaches convert to different local optima depending on the starting search point, while Markov Chain Monte Carlo methods require a relatively high computational time to find a solution close to the optimum. The efficiency of methods with simplifying assumptions depends on how these assumptions affect the accuracy of identified solution.

2.5.1 Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

CMA evolution strategy (ES) is a non-linear optimization method that was first proposed by Hansen and Ostermeier [25]. It aims to optimize (minimize) an objective function through an iterative process. In the initialization step, a set of candidate solutions are sampled from a predefined multivariate normal distribution. At each iteration, the algorithm updates the mean and covariance matrix of the multivariate normal distribution. This distribution is then used to sample a population of new candidate solutions.

The mean of the multivariate normal distribution is updated by a combination of the generated candidate solutions with small (good) objective function values. The adaptation of the covariance matrix of the multivariate normal distribution is

based on the evolution path. "The evolution path can be considered as a sum of the consecutive steps of the distribution mean" [24]. CMA-ES continues until a convergence (termination) criterion is satisfied.

CMA-ES is preferred to many other heuristic optimization algorithms such as hill climbing, particularly, when dealing with an "ill-conditioned problem" where the search landscape is rugged due to outliers, noise, or local optima. These conditions will not affect CMA-ES, while many other heuristic optimization algorithms fail and are not able to find the global optima. Among the interesting attributes of CMA-ES are its invariance properties. This algorithm is invariant with respect to the "angle preserving transformations of the search space" such as rotation, and "order preserving transformations of objective function value" [23].

However, CMA-ES requires a relatively high computational time to converge to a sample solution close to optimum, compared to other heuristic search algorithms. Due to this requirement, the application of CMA-ES is limited to problems with small search space dimensions usually between three and a hundred [24].

In this work, the high computational time requirement is addressed by proposing a new evolutionary strategy, M-CMA-ES for finding the best structure of a DBN model. In this method, an explicit memory is added to CMA-ES to keep a fixed number of good samples from previous iteration. This modification causes the algorithm to converge faster and makes it more applicable to reverse engineering of gene networks.

Chapter 3

Datasets And Preprocessing

In this thesis, in order to evaluate the developed approaches for reverse engineering of gene networks, three different datasets are used; two temporal artificial datasets and one biological dataset. First, a temporal synthetic dataset is developed and used for comparing the DBN learning strategy, M-CMA-ES, with greedy hill climbing. The network structure of the synthetic dataset is devised to enable the investigation of the efficiency of the methods for inferring both linear and non-linear interactions. A few genes with more than one regulator are also included. For further validation, the proposed approach is applied to Brainsim dataset which is a simulated temporal gene expression dataset, representing gene regulatory interactions in response to the singing behavior in a song bird. This dataset is a good benchmark for validating the methodology as the inferred interactions can be compared to those that actually exist in the data as well as results reported in the literature. Finally a biological dataset from yeast *Saccharomyces Cerevisiae* Cell Cycle is used. This dataset has two major advantages over many other publicly available datasets. First, the pathway of yeast *Saccharomyces Cerevisiae* cell cycle is well known, and second, this dataset has been

extensively studied using several reverse engineering approaches. These advantages make it possible to evaluate the performance of the proposed reverse engineering approach and compare it with other approaches as well.

In the the following chapter, the techniques applied to develop the above-mentioned datasets are described as well as the preprocessing employed to prepare them for later analysis.

3.1 Datasets

3.1.1 Temporal Synthetic Dataset

In order to assess how well and efficient M-CMA-ES can be used for learning of a DBN structure, a time-delayed gene regulatory pathway of arbitrary structure is modeled. Hereafter, this dataset is refereed to as 'synthetic data'. The network structure of the considered model is as shown in Figure 3.1.

The values of gene expression levels of 15 genes at discrete time points, *t*, are generated based on expression values at previous time points, *t-1*. The gene expressions are calculated by the following stochastic formula:

$$X_{t+1} = A.X_t + E \tag{3.1}$$

Where X_t and X_{t+1} are vectors containing the expression levels of all genes at time point t and $t+1$, respectively; A is a two-dimensional matrix that represents the association between genes in the underlying pathways, and E is a vector of noise values with uniform distribution. The stochastic nature of the above formula originates from this noise.

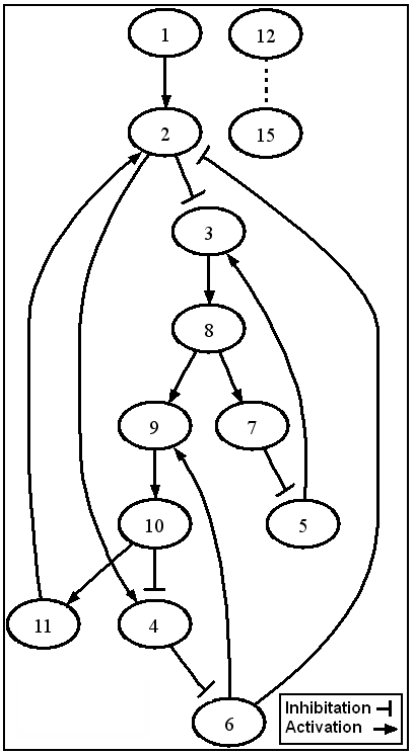

Figure 3.1: Time-delayed network structure of the synthetic dataset

Examining the range of the gene expression measurements shows that most expression levels X_t, lie within the range of -150 to 150. The value of each entry (i,j) of matrix A is a non zero constant if there exists some dependency between the regulator gene i and target gene j; zero otherwise. The magnitude and sign of the entries of matrix A indicate the strength and type of gene regulations (up or down regulations), respectively [66].

In Equation 3.1, X_t is initialized to zero. The expression values of all genes are generated recursively at 150 time points. The gene expression values of a portion of time points from the beginning are ignored as they are greatly affected by noise.

The rest of the time series containing 100 time points are used for reverse engineering of the gene regulatory network. The process, explained above, is repeated 50 times to generate 50 datasets where each dataset contains 100 observations of 15 genes. The process could have been repeated more than 50 times, however due to time constraints, this number is limited to 50.

If the effect of the noise element, E, in Equation 3.1 is ignored, the expression seems like a simple linear equation. However the model can also have a nonlinear behavior. The non-linear behavior exist among the genes and their regulators provided that they are affected by more than one regulator and that at least one of the regulators acts as an activator while the other one in an inhibitor. The changes in the expression values of such genes are caused by at least two regulators with opposite effects and therefore can not be predicted linearly by only one of the regulators. As such, the relationships between these genes and their regulators are considered as non-linear relationships. For instance, in this model, the associations between genes 2, 3, and 4 and their regulators follow a non-linear relationship.

3.1.2 Brainsim Simulated Dataset

The Brainsim Dataset employed in this study was produced by Smith et al. [58] using the Brainsim simulator. Brainsim models the singing behavior of a songbird, in five distinct regions of the brain, with expression levels of 100 simulated genes and the activity level in each of these regions of the brain. During simulation, two states of '0' or '1' map to 'singing ' or 'silence' which are two observable behavior of a songbird. The absolute values of gene expressions are measured in the range of 0 to 50 and the brain activity levels have values between 0 and 400 Hz. The singing behavior of a

songbird indirectly affects the gene expression levels in the artificial gene network by changing the activity level. In the first four regions of the brain, behavior is correlated with the activity level. As such, these regions are called related regions. In two of these four regions, there is a direct association between behavior and activity level as the 'singing' and 'silence' behavior correspond to activity levels in the range of (300-400 Hz) and (0-100 Hz) respectively. In the other two regions, this association is reversed. In the fifth region, the behavior and activity are not correlated. The gene regulatory network in every region contains 100 genes; however only 10 of these genes are connected to each other and the rest are irrelevant or independent. Two of the related genes are directly affected by the activity level and the remaining eight genes associate to the activity level through these two genes (Figure 3.2). The expression levels of irrelevant genes randomly fluctuate within the afore-mentioned range of 0 to 50. However, the expression level of the relevant genes at each time point are determined by the expression level of their regulators, a noise factor and a degradation factor.

In this work, 400 different datasets are generated from 400 different BRAINSIM, where each dataset is composed of 25 genes at 20 time points, sampled at simulated 5 minute intervals. In order to exclude songbird specific biases, the 400 datasets are generated from 2 different simulated songbirds (200 datasets each). The data from each songbird, is equally taken from 4 simulated brain regions where the regions differed slightly from one another through the weight values associated with their regulatory connections. In order to avoid the unnecessary complexities of the regulatory networks, the number of genes are limited to 25 composed of 10 related and 15 irrelevant genes. The regulatory network underlying all of the datasets are the same,

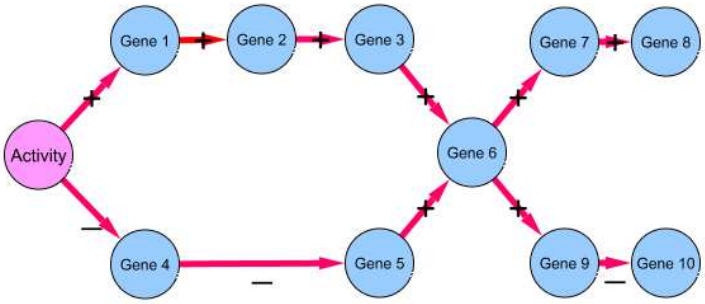

Figure 3.2: The network structure of gene regulatory interactions simulated in the Brainsim dataset. The upregulating interactions are indicated by '+' and down regulating interactions are represented by '-'.

however the associated weight values are different.

In the rest of the thesis, this dataset is referred to as 'artificial dataset'.

3.1.3 Yeast *Saccharomyces Cerevisiae* Dataset

Within the last few years, many reverse engineering approaches have been applied to budding yeast, *Saccharomyces cerevisiae*, to capture the mechanism of gene interactions during cell cycle in this organism. This is due to the fact that details of cell cycle control are well studied in yeast. Moreover well established time series data of yeast *Saccharomyces cerevisiae* are available [59]. In this work, our approach is applied to 14 selected yeast cell-cycle genes *CLN1*, *CLN2*, *CLN3*, *CLB5*, *CLB6*, *CDC28*, *MBP1*, *SWI4*, *SWI6*, *FAR1*, *SIC1*, and *FUS3*, *CDC6*, and *CDC20* which are known to have key functions in the regulation of the early cell-cycle. Most of the genes also oscillate through one peak per cell cycle [59].

Figure 3.3 illustrates the KEGG pathway of 14 genes involved in the early cell cycle of the yeast *Saccharomyces cerevisiae* (budding yeast) [1].

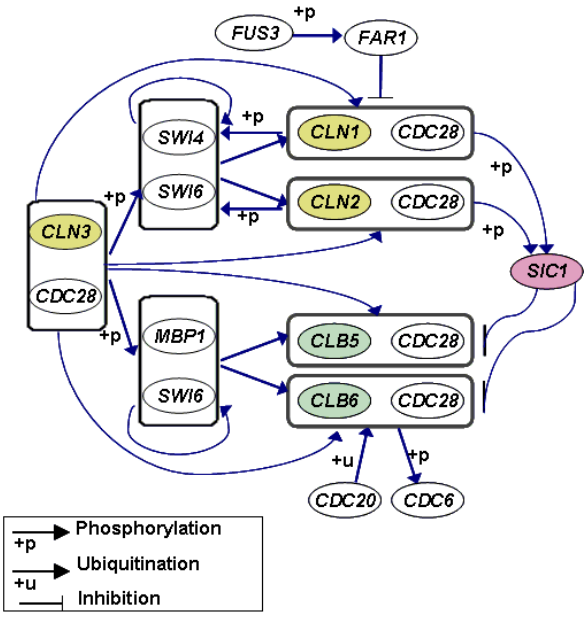

Figure 3.3: The yeast cell cycle pathway of 14 genes in KEGG

KEGG, Kyoto Encyclopedia of Genes and Genomes, is a project initiated in 1995 at Kyoto University to the goal of organizing all current knowledge of molecular and genetic pathways from experimental observations, "The KEGG Pathway database is a collection of manually drawn pathways consists of both metabolic pathways and regulatory pathways." KEGG regulatory pathway represents the current knowledge on the protein and gene interaction networks. [53].

Figure 3.4 includes some associations at phosphorylation, ubiquitination, and inhibition level in addition to regulations. The exact yeast cell cycle pathway that is used as a target network to evaluate and compare our resulting interactions are illustrated

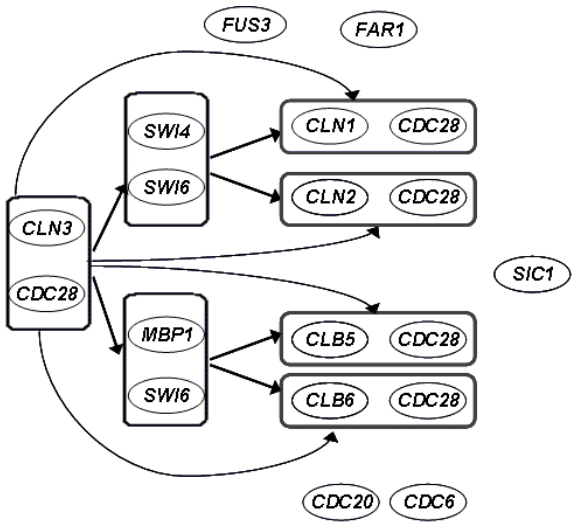

Figure 3.4: The yeast cell cycle pathway of 14 genes used as our target network

in Figure 3.4. This Figure is a modification of Figure 3.3. The associations represented in Figure 3.4 are mostly at the level of gene regulations. The only exceptions to this rule are the phosphorylation associations between the CLN3/CDC28 complex and the two transcription factor complexes SWI4/SWI6 and MBP1/SWI6. These associations are included as each of the transcription factor complexes contain a feed forward loop. Increasing levels of CLN3/CDC28 phosphorylation activity directed at the transcription factor complexes leads to the increase in the transcription level of the component genes and therefore there is an indirect association at the level of gene regulation between the CLN3/CDC28 complex and the two transcription factor complexes SWI4/SWI6 and MBP1/SWI6. All inferred interactions within the complexes are ignored as they do not have external effects independent of each other.

The yeast cell-cycle dataset generated by Spellman [59] includes three time series referred to “alpha”, “cdc15”, and “elu” synchronized via three different approaches:

alpha factor arrest, elutriation, and arrest of a cdc15 temperature-sensitive mutant. Datasets alpha, cdc15, and elu contain 18, 24, and 14 time points measured over a period of 119, 290, and 390 minutes, respectively. In this work, the reverse engineering method is employed to all three time series; Results from the cdc15 dataset are then studied in more detail as they were more accurate.

3.2 Preprocessing

In order to remove systematic bias in datasets and prepare them for later analysis, some manipulation to the data is needed as below:

- outlier removal
- missing value estimation
- scaling
- interpolation

3.2.1 Outlier removal

Prior to preprocessing the yeast cell-cycle and the synthetic datasets, outliers are identified. Outliers are defined as the expression values which do not seem to be correctly measured compared to most of the other values. There are no outliers in the Brainsim dataset.

To determine the outliers, it is assumed that the expression values of a gene over time follows a normal distribution. Mean and standard deviation of every gene is computed over time. Any expression value less than $Mean - 2 \times Std$ or more than

$Mean + 2 \times Std$ is considered as outlier. Outliers are replaced with the Median of each gene expression value over time.

3.2.2 Missing value estimation

Among the three datasets used for evaluation our method, only the yeast cell cycle data contains missing values. Both the synthetic and artificial datasets are devised with a full matrix of expression values. Similar to outliers, the missing gene expression values are replaced with the median of each gene expression value over time.

3.2.3 Scaling

In addition to outlier removal and missing value estimation, all datasets are scaled to be in the range of -1 and +1. For this purpose, the minimum and maximum expression values of every gene, j, over time are calculated and the gene is scaled according to the following formula:

$y_{ij}[i = 1..n] = \frac{x_{ij}[i=1..n]-Min(X_j)}{Max(X_j)-Min(X_j)} \times 2 - 1$

Where x_{ij} and y_{ij} are the expression value of gene j at time point i before and after scaling the vector. X_j is the expression profile of gene j over time.

After scaling, the expression values of all genes are in the range of -1 to 1.

3.2.4 Interpolation

Finally, interpolation is used to add time points (at 10 minutes intervals) between the original gene expression measurements in the 'elu' yeast cell cycle dataset. Interpolation is only used for the 'elu' dataset as its gene expression values are measured

every 30 minutes. This interval is 7 and 10 minutes for the two other yeast datasets 'alpha' and 'cdc15'.

In this work, Piecewise Cubic Interpolation method (PCHIP)[1] is used to construct new data points between the original measured time points. PCHIP receives a set of data points (e.g. t_i, y_i , [i=1..n], with $t_1 < t_2 < ... < t_n$) and constructs a piecewise cubic interpolant composed of a different cubic polynomial in each subinterval $[t_i, t_{i+1}]$. PCHIP produces a continuous piecewise cubic polynomial where its first derivation is also continuous.

[1] Piecewise Cubic Hermite Interpolating Polynomial

Chapter 4

Network Reverse Engineering

In this thesis, DBNs are used for reverse engineering of gene regulatory networks. The proposed method for learning of DBN, M-CMA-ES, is a score-based method. To construct a DBN model using score-based methods, one needs to (i) derive a criterion for evaluating the goodness of a network based on the given gene expression data; and (ii) search through the space of all possible network structures and parameters for the optimum structure that best fits the expression data.

Score-based learning of DBNs was separately conducted into two steps, estimating the values of the parameters of the model and calculating the scoring function based on the estimated parameter values. To learn a DBN model, for each gene, one needs to search for the optimum set of regulators. To this end, local BNRC scoring function is used as described in [34, 41] for assessing how well the network structure of a variable and its regulators fits the data. A heuristic algorithm, M-CMA-ES, is developed to search through the space of all possible regulators of a target variable. The developed approach is also compared with greedy hill climbing, a simple heuristic search strategy commonly used in the literature to find the best structure, one that minimizes the

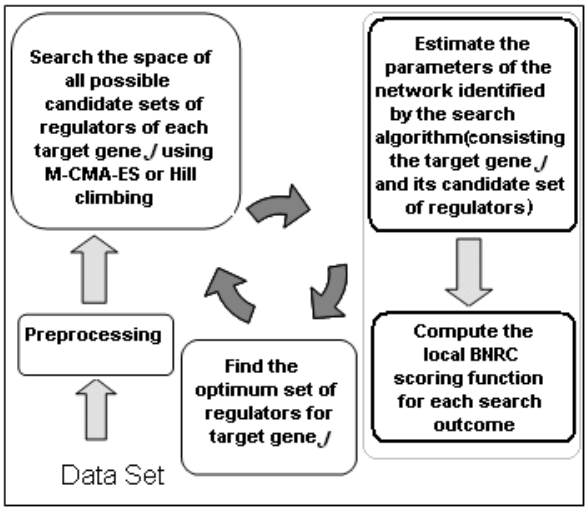

Figure 4.1: Illustration of the proposed score-base approach for reverse engineering of gene regulatory networks.

scoring function. Figure 4.1 summarizes the proposed approach, in this thesis, for reverse engineering of gene networks.

In our reverse engineering approach, time series of expression values of a set of genes are the input values and the goal is to capture a connectivity structure that best describes the data. To this end, for each gene, the best substructure consisting of the gene and its regulators is inferred. All these substructures are then merged to construct the optimum structure. In order to find the best substructure for a given gene that is equivalent to determining its best set of regulators, an evolutionary search strategy, M-CMA-ES, is employed. M-CMA-ES iteratively searches through the space of all possible candidate sets of regulators for a gene of interest to find the optimum set of regulators. The ones that along with the target gene compose a substructure with the minimum Bayesian (BNRC) score. M-CMA-ES randomly

generates each candidate set of regulators by sampling from an N-variate normal distribution. Each generated sample is an N-dimensional vector that maps to the space of all possible regulators and uniquely determines a set of regulators with the maximum size of three; gene regulators corresponding to the positive elements of the sample vector are included in the candidate set of regulators. M-CMA-ES, at each iteration, generates a population of candidate set of regulators and then a subset of the generated samples with the optimum (minimum) BNRC score are used to update the normal distribution which will be employed for sampling the next generation of the candidate solutions. The algorithm continues until a convergence criterion is satisfied (the algorithm will be described in detail in section 4.1.1).

The BNRC score is computed for every given gene and their candidate set of regulators by assuming that the association between each gene and its regulators follows a non-parametric regression model. More precisely, in this model, the expression value of a given gene at a given time point t is estimated by a linear combination of a set of polynomial regression function such that the i-th regression function receives the expression values of the i-th regulator at the previous time point, t-1, as an input argument. The BNRC score is determined in a way that the network with lower score results in a better structure and more fits the data. The details of calculating the BNRC score is explained in section 4.2.

4.1 Network Structure Search Strategies

4.1.1 Covariance Matrix Adaptation Evolutionary Strategy with Explicit Memory

In this work, an evolutionary strategy M-CMA-ES (Covariance Matrix Adaptation Evolution Strategy with explicit Memory) is proposed to search for the best DBN structure given the gene expression data.

M-CMA-ES is based on CMA-ES (Covariance Matrix Adaptation Evolution Strategy) first proposed in [25] as a heuristic approach for parameter optimization of non-linear objective functions [55]. CMA-ES aims to optimize (minimize) an objective function through an iterative process. In the initialization step, a set of candidate solutions are sampled from a predefined multivariate normal distribution. At each iteration, the algorithm updates the mean and covariance matrix of the multivariate normal distribution. This distribution is then used to sample a population of new candidate solutions. The algorithm continues until a convergence (termination) criterion is satisfied.

Our purposed approach improves the convergence time of CMA-ES by guaranteing that the best generated sample at each iteration of the algorithm is at least as optimum as that of previous iteration while the size of the generated population does not change through iterations. This is achieved by adding an explicit memory to CMA-ES to store a fixed number of good samples from the generated population at the previous iteration of the algorithm. These samples are then added to the list of the newly generated population (Figure 4.2).

M-CMA-ES iterates until at least one of the following termination conditions is

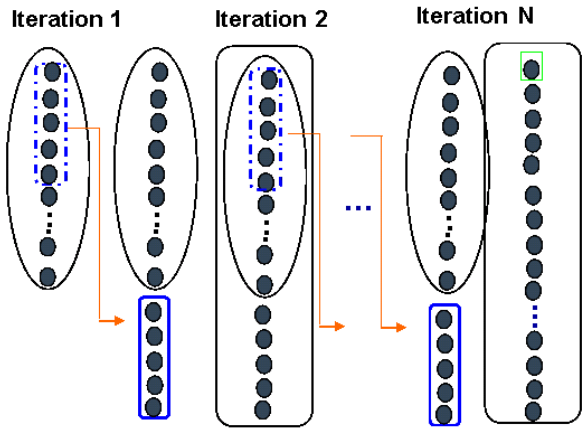

Figure 4.2: Illustration of iterations in M-CMA-ES; The circles represent the samples generated at every iteration. The samples are sorted in an ascending order according to their BNRC score. The explicit memory is shown with a small rectangular.

satisfied:

- The maximum difference between the value of the objective function, for the best sample, in three successive generations is less than a predefined threshold value. This value is set to a very small number close to zero.
- The total number of objective functions evaluated during iterations is more than a set maximum. The value of 200 is set for the maximum number of functions being evaluated during iterations.
- The value of the objective function for the best sample in an iteration is less than a specific threshold (close to the global optima). For this purpose, the threshold value of -1000 is chosen which is very rarely reached. Therefore this condition is usually met less often than the other two conditions.

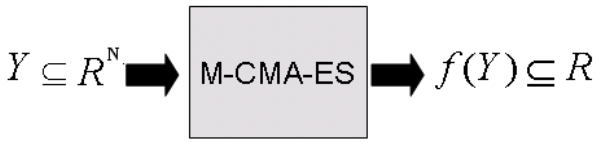

Figure 4.3: M-CMA-ES as a black box function

M-CMA-ES can be considered as a black box search algorithm (Figure 4.3) to optimize the fitness function $f : Y \subset R^N -> f(Y) \subset R$ in the space of N genes, where the N-dimensional vector Y is generated by a N-variate normal distribution (Figure 4.3). In order to use M-CMA-ES for finding the best structure of DBN, the fitness function f as well as the N-dimensional vector Y should be defined in the problem. The fitting function is defined to be the Bayesian scoring function. The N-dimensional vector Y, assuming N to be the number of all genes in a given dataset, is mapped to the search space of our problem such that each element of the vector Y corresponds to a specific gene (i.e the first element of the vector maps to the first gene, the second element of the vector maps to the second gene,...). Each element of the vector Y is sampled from a normal distribution and may hold either a positive or a negative value. It is conditioned that only those genes that map to a positive element of vector Y are counted as potential gene regulators. Therefore, vector Y represents a candidate set of regulators for a given target gene. To avoid the unnecessary complexities of network connections, an upper bound constraint is set on the number of gene regulators for a given target gene. For this purpose, an instance of Y with at most 3 positive values is considered as a valid input to the fitness function . The output value of the fitness function for an input vector Y indicates how well

the network structure composed of the set of regulators corresponding to the positive values of vector Y, and a specific target gene fits the data.

To implement M-CMA-ES, it is needed to initialize the mean vector and the covariance matrix of a multivariate normal distribution for sampling the first generation of the search points. The covariance matrix of the normal distribution is set to an identity matrix[1]. A suitable initialization value is assigned for the mean vector of the distribution by taking into account the upper bound constraint for the number of regulators. The mean vector is computed so that the probability of generating a valid N-dimensional sample, Y, from the multivariate normal distribution is maximum. For this purpose, the probability function of having a vector Y with exactly l positive values is computed as follows:

$$f(p) = \frac{n!}{(n-l)!l!} p^l (1-p)^{n-l} \tag{4.1}$$

where p is the probability of getting a positive value for each element of Y and n is the total number of genes. Next, *f(p)* is plotted for *p=0, 0.01,...,1,*. The goal is to find the peak value for p, p'. Setting p' as the probability of getting a positive value for each element of sample vector Y, there is the maximum chance of getting a sample of interest (with exactly l positive values). By examining the standard normal *(z)* distribution, the z-value corresponding to p' is determined. Finally, the mean of the assumed standard normal distribution of each element of Y is shifted from 0 to -z.

Figure 4.4 illustrates the steps of the proposed M-CMA-ES approach.

[1]The identity matrix or unit matrix of size n is the n-by-n square matrix with ones on the main diagonal and zeros elsewhere [Wikipedia].

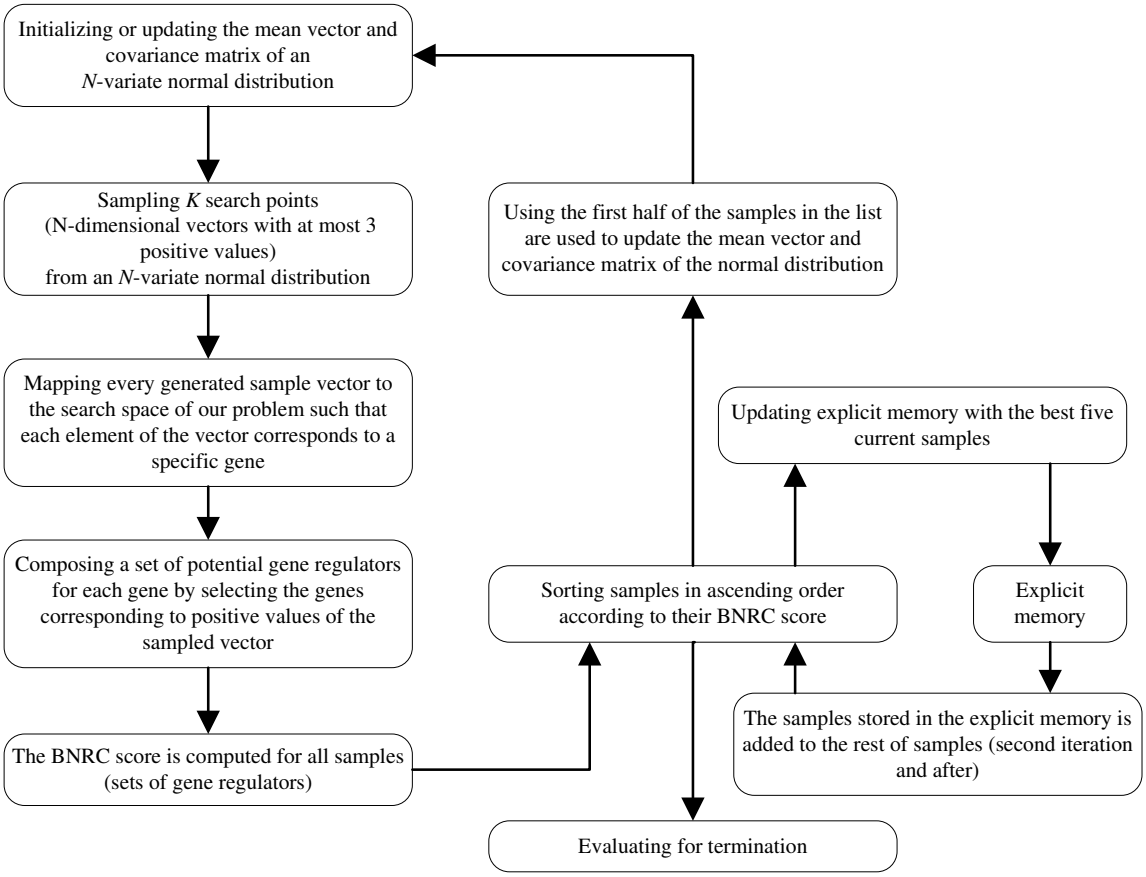

Figure 4.4: The steps of M-CMA-ES for selecting the best set of regulators for a given gene.

4.1.2 Greedy Hill Climbing

For comparison purposes, a modified version of greedy hill climbing is also employed to search for the optimum set of regulators for each gene in the space of candidate sets.

In the modified greedy hill climbing, similar to M-CMA-ES, a specific upper-bound constraint is imposed on the number of regulators for each gene. In the rest of the thesis, the modified greedy hill climbing is referred to as simply 'hill climbing'. Prior to applying hill climbing, for each gene, all other genes are sorted as potential regulators based on value of the scoring criterion (BNRC, section 4.2) in ascending order. For each gene, hill climbing starts from an empty candidate set of regulators and follows as:

1. For each target gene, j, add the first gene in the sorted list of its potential regulators to the empty candidate set.

2. If the local BNRC score improves, add the second gene in the sorted list to the current candidate set and compute the score; otherwise, remove the recently added gene from the candidate set and add the next gene in the sorted list and compute the score.

3. Repeat step 2 until all the potential gene regulators in the sorted list have been tested or the size of the candidate set reaches its upper bound.

4. Repeat the three previous steps for all of the genes in order to find the optimum network.

Computational time and memory required for hill climbing are less than M-CMA-ES, however hill climbing is more probable to get trapped in the local optimum [29].

4.2 Network Fitting

In order to calculate the local BNRC scoring function for a network consisting of a target gene j and its regulators (parents), one needs to estimate the network structure parameters . These two steps, estimating the network parameters and calculating the score, are conducted separately.

The local BNRC scoring function for gene j , $BNRC_j$, is based on the posterior probability of a known network structure given the gene expression data, and is defined as

$$BNRC_j = -2Log(P(M_j|D)) \tag{4.2}$$

where M_j is the network structure of gene j and its parents and D represents the gene expression dataset. According to Bayes theorem, Equation 4.2 can be rewritten as

$$BNRC_j = -2Log\frac{(P(D|M_j)P(M_j))}{P(D)} \tag{4.3}$$

Since the denominator of Equation 4.3 is the same for all possible network structures, M_j, inferred from the same Data, D, it can be ignored and only the nominator of Equation 4.3 is used as the network structure evaluation criterion.

P(M_j), the prior probability of the network structure M_j, is calculated as $P(M_j) = \exp(-(r_j + 1))$ where r_j is the number of parents of gene j [34]. P($D \mid M_j$), The probability of data D given the Network Structure M_j, is computed using the following equation which is a marginal likelihood that averages the probability of the data over all possible parameter assignments to M_j [20]:

$$P(D|M_j) = \int P(D|M_j, \theta_j)P(\theta_j|M_j)d\theta_j \tag{4.4}$$

In Equation 4.4 , θ_j represents the parameter vector of the conditional distribution of gene j given its parents. It is assumed that the association between each gene and its parents follows a non-parametric regression model in the form of:

$$\begin{aligned} x_{ij} = m_1^{(j)}(p_{(i-1)1}^{(j)}) + m_2^{(j)}(p_{(i-1)2}^{(j)}) + \ldots + \\ m_{q_j}^{(j)}(p_{(i-1)q_j}^{(j)}) + e_{ij} \end{aligned} \tag{4.5}$$

Where x_{ij} is the expression value of gene j at time point i and $p_{(i-1)k,\,(k=1\ldots q_j)}$ is the expression value of the k-th parent of the gene j while q_j stands for the number of parents of gene j. The error of e_{ij} is assumed to have a Normal distribution with mean of 0 and standard deviation of σ .

In Equation 4.5, $m_{k,\,(k=1\ldots q_j)}$ are non parametric regression functions from $R \mapsto R$ and are defined as:

$$m_k^{(j)}(p_{(i-1)k}) = \sum_{m=1}^{\mu_{jk}} \gamma_{mk}^{(j)} B_{mk}^{(j)}(p_{(i-1)k}^{(j)}) \; k = 1 \ldots q_j \tag{4.6}$$

where $B_{mk}^{(j)}$ is the m^{th} B-spline Basis function for k^{th} parent of gene j. The same number of B-spline functions is considered for all of the genes ($\mu_{jk} = 20$). The coefficients $\gamma_{mk}^{(j)}$ ($m = 1 \ldots 20,\; k = 1 \ldots q_j$) and σ_j are unknown parameters of the network structure M_j that should be estimated.

Equation 4.6 is substituted in Equation 4.5 and the non parametric regression of Equation 4.5 is written in the form of a probability density function as follows:

$$\begin{aligned} f_j(x_{ij}|p_{i-1k}^{(j)}, \gamma_{mk}^{(j)}, \sigma_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \times \\ e^{-\frac{(x_{ij} - \sum_{k=1}^{q_j} \sum_{m=1}^{\mu_{jk}} \gamma_{mk}^{(j)} B_{mk}^{(j)}(p_{(i-1)k}^{(j)}))^2}{2\sigma_j^2}} \end{aligned} \tag{4.7}$$

for $i = 2 \ldots n$ where n is the number of time points. The right hand side of the

above equation can substitute $P(D|M_j, \theta_j)$ in Equation 4.4 . $P(\theta_j|M_j)$, in Equation 4.4 is computed as described in [41, 40]. In order to calculate the high dimensional integration in Equation 4.4, Laplace approximation is used. For more details of implementation of Laplace approximation, the reader is referred to [41].

Thus far, all required elements to calculate the local BNRC scoring function are complete. It is only requires to approximate the parameters of the model. The Backfitting algorithm is used for estimating the parameters of the network structure of a target gene j and its parents $(\gamma_{mk}^{(j)}(k = 1 \dots q_j, m = 1 \dots \mu_{jk}), \sigma_j)$.

4.2.1 Backfitting

Backfitting is a general iterative procedure for fitting additive models (i.e., approximating the additive components) using any regression-type fitting mechanisms. In this method, at each step, one component is estimated while other components are kept fixed. The algorithm iterates until the value of mean square error of the model converges [26, 7].

In order to be more precise, consider the additive model: $f(X) = S_0 + \sum_{i=1}^{n} S_i(X_i)$, Backfitting aims to estimate the smoothing terms $S_0, S_1(.), S_2(.), \dots, S_n(.)$ where $E[S_i(X_i)] = 0$ for every i . For this purpose, the i^{th} set of partial residuals is defined as below:

$$R_i = f(X) - S_0 - \sum_{k \neq i} S_k(X_k) \tag{4.8}$$

And therefore, $E(R_i|X_i) = S_i(X_i)$. Based on this observation, the value of each smoothing function S_i can be iteratively approximated provided that the values of all other $(S_k, k \neq i)$ are determined. The above iterative procedure is known as

Backfitting algorithm [26, 7].

4.3 Validation And Benchmarking

In order to evaluate the performance of the proposed approach for inferring the actual connectivity networks from the datasets, the evaluation criteria 'Sensitivity' and 'Precision' are used which are extracted from a matrix, named the 'confusion matrix'.

4.3.1 Confusion Matrix

In predictive analysis, a confusion matrix is a table with two rows and two columns also called a 'table of confusion'. Generally speaking, each column of the matrix corresponds to the instances in a predicted class, while each row represents the instances in an actual class. The table of confusion, shown in Table 4.1, represents the number of 'True Positives', 'True Negatives', 'False Positives', and 'False Negatives'.

Table 4.1: Confusion Matrix. P and P' are the total number number of interactions in the actual and the predicted networks and N and $N'$are the total number of missing interactions in the actual and the predicted networks.

*	Predicted Outcome		Total
Actual Values	True Positives False Positives	False Negatives True Negatives	P N
Total	P'	N'	

The entries in the confusion matrix have the following meaning in the context of our study:

- True Positives: the number of actual interactions which are truly predicted; or

the interactions which exists in the actual connectivity network and also are inferred by the network inference approach.

- True Negatives: the number of interactions which are also truly not predicted; or the interactions which do not exist in the actual connectivity network and are also not inferred by the network inference approach.

- False Positives: the number of missing interactions which are falsely predicted; or the interactions which do not exist in the actual connectivity network but are inferred by the network inference approach.

- False Negatives: the number of actual interactions which are falsely not predicted; or the interactions which exist in the actual connectivity network but are not inferred by the network inference approach.

4.3.2 Evaluation Criteria

Several evaluation criteria can be defined based on the confusion matrix. According to the nature of our work, two of these criteria, 'Precision' and 'Sensitivity' are used. They are defined as bellow:

Precision

"Precision" refers to the ratio of the number of correctly estimated interactions to the total number of estimated links.

$$Precision = \frac{NumberofTruePositives}{NumberofTruePositives + NumberofFalsePositives} \tag{4.9}$$

Sensitivity

"Sensitivity" is defined as the ratio of the number of correctly estimated interactions to the number of actual interactions.

$$Sensitivity = \frac{NumberofTruePositives}{NumberofTruePositives + NumberofFalseNegatives} \tag{4.10}$$

Chapter 5

Implementation and Results

The network inference approach, proposed in this thesis, is implemented using MATLAB (MathWorks). In this chapter, the implementation details of the reverse engineering approach as well as the inferred networks for each dataset are presented. The network obtained from the temporal synthetic dataset using DBN with M-CMA-ES are evaluated and compared with the networks inferred by DBN using hill climbing search strategy. The networks inferred from the yeast cell cycle data are compared to the KEGG pathway of the yeast. The efficiency of the predicted transcriptional networks are compared with those of other similar yeast network inference studies.

5.1 Analysis of Temporal Synthetic Dataset

In order to evaluate the performance of the proposed method, M-CMA-ES, for learning the structure of DBN, it is applied to 50 synthesized datasets. These datasets, are very similar except in the amount of noise, E, according to Equation 3.1. It is expected that the reverse engineering method captures the underlying network structure, as

shown in Figure 3.1, similarly in all datasets.

The accuracy and sensitivity of the proposed method is also compared with hill climbing search strategy. The next challenge is to quantify the accuracy of the inferred networks, averaged over the 50 datasets. To determine which structural connections found by each method (M-CMA-ES and hill climbing) are reliable, and which are artifacts of the noise in the network, a bootstrap approach is used. In this approach, the distribution of incorrect connections inferred by both methods is estimated. Those distributions are then used to set thresholds for determining which links are significant, and which are likely spurious.

To derive a distribution for spurious connections found by each DBN method, 50 synthetic datasets are generated with the known structure. These datasets are then randomized by shuffling their values over time in all genes (in 25 datasets) or by shuffling their values over genes (in the other 25 datasets). Next, DBNs with M-CMA-ES and hill climbing are employed to reverse engineer the 50 random datasets. The above steps are repeated 10 times. The histograms are then plotted for the number of times that every pairwise interaction is inferred from the 50 random datasets by reverse engineering for M-CMA-ES and hill climbing separately. The histograms, presented in Figure 5.1-a and 5.1-b, show jumps at the thresholds of 18 and 14. As a result, the thresholds are set to conclude that an interaction between two genes is deemed significant if it appears at least 14 times and 18 times in the 50 networks inferred by hill climbing and M-CMA-ES, respectively.

By comparing the histograms shown in Figure 5.1-a and 5.1-b, it is observed that on average, the number of times that an interaction is inferred from 50 bootstrapped datasets by M-CMA-ES is higher than hill climbing. This observation is predictable

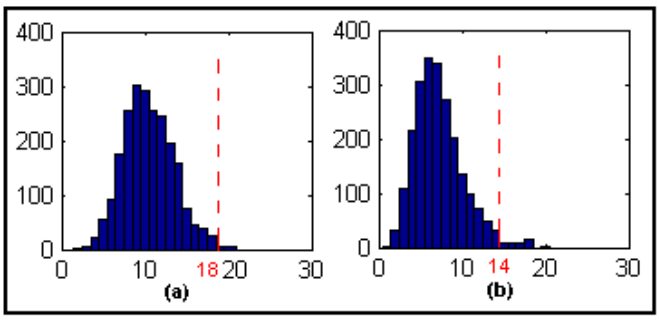

Figure 5.1: The histogram of the number of times that each pairwise gene interaction in the simulated dataset is inferred from 50 randomized datasets in 10 runs by DBN with (a) M-CMA-ES and (b) hill climbing

as M-CMA-ES and hill climbing explore the search space for the candidate set of regulators of a specific gene in different ways. Hill climbing starts from an empty set of regulators (candidate solution) and adds potential regulators to the empty set through exploring the search space. Meanwhile, M-CMA-ES generates a candidate solution by sampling from a multivariate normal distribution. Due to this difference, it is expected that under similar upper bound constraint on the number of regulators, M-CMA-ES will capture a higher number of interactions than hill climbing.

After applying M-CMA-ES and hill climbing to 50 synthesized datasets, the elements of 50 connectivity matrices obtained from these datasets, by M-CMA-ES and hill climbing are summed up separately to determine significant interactions for each model. The results are summarized in the Tables 5.1 and 5.2, respectively. In these tables, only values greater than the thresholds 18 for M-CMA-ES and 14 for hill climbing are shown. Each entry *(i,j)* of these tables holds the total number of times that the interaction between regulator gene i and target gene j is captured in the 50

Table 5.1: Connectivity matrix inferred by DBN using M-CMA-ES from 50 synthesized datasets.

*	1	2	3	4	5	6	7	8	9	10	11	12-15
1	.	.	.	.	.	.	.	.	.	.	.	...
2	.	.	31	22	.	.	.	.	.	.	.	...
3	.	.	.	.	.	.	.	37	.	.	.	...
4	.	.	.	.	.	34	.	.	.	.	.	...
5	.	.	×	.	.	.	.	.	.	.	.	...
6	.	24	.	.	.	.	.	.	21	.	.	...
7	.	.	.	.	25	.	.	.	.	.	.	...
8	.	.	.	.	.	.	24	.	20	.	.	...
9	.	.	.	.	.	.	.	.	.	26	.	...
10	.	.	.	19	.	.	.	.	.	.	27	...
11	.	×	.	.	.	.	.	.	.	.	.	...
12	.	.	.	.	.	.	.	.	.	.	.	...
13	.	.	.	.	.	.	.	.	.	.	.	...
14	.	.	.	.	.	.	.	.	.	.	.	...
15	.	.	.	.	.	.	.	.	.	.	.	...

Each entry in row i, column j of the matrix represents the total number of times that the relationship between gene regulator i and target gene j is inferred from 50 synthesized datasets using DBN with M-CMA-ES. The elements of the table filled with dots, are less than threshold of 18. The elements marked by $\times$ are actual interactions not captured by the method.

synthesized datasets. The elements of the tables filled with dots, are less than the threshold of 14. The elements marked by $\times$ are actual interactions not captured by the method.

The inferred networks corresponding to tables 5.1 and 5.2 are shown in the Figures 5.2-a and 5.2-b, respectively.

For plotting the graphs, the open source graph visualization software Graphviz[1] from AT&T Research Labs is used. In these figures, the dashed lines are the actual

[1]Graphviz Website, http://www.graphviz.org

Table 5.2: Connectivity matrix inferred by DBN using hill climbing from 50 synthesized datasets.

*	1	2	3	4	5	6	7	8	9	10	11	12-15
1	.	.	.	.	.	.	.	.	.	.	.	...
2	.	.	22	×	.	.	.	.	.	.	.	...
3	.	.	.	.	.	.	.	35	.	.	.	...
4	.	.	.	.	.	33	.	.	.	.	.	...
5	.	.	×	18	.	.	.	.	.	.	.	...
6	.	15	.	.	.	.	.	.	28	.	.	...
7	.	.	.	.	32	.	.	.	.	.	.	...
8	.	.	.	.	.	.	34	.	×	.	.	...
9	.	.	.	.	.	.	.	.	.	32	.	...
10	.	.	.	×	.	.	.	.	.	.	29	...
11	.	×	.	.	.	.	.	.	.	.	.	...
12	.	.	.	.	.	.	.	.	.	.	.	...
13	.	.	.	.	.	.	.	.	.	.	.	...
14	.	.	.	.	.	.	.	.	.	.	.	...
15	.	.	.	.	.	.	.	.	.	.	.	...

Each entry in row i, column j of the matrix represents the total number of times that the relationship between gene regulator i and target gene j is inferred from 50 synthesized datasets using DBN with hill climbing. The elements of the table filled with dots, are less than the threshold of 14. The elements marked by × are actual interactions not captured by the method.

interactions which were not found by the corresponding network inference method. The only extra interaction in the network obtained by hill climbing is highlighted by a bold line (between genes 5 and 4).

By comparing the inferred networks with the original network structure, it is observed that:

- All of the relationships between genes in which the target gene is regulated by only one gene are captured by DBN and both M-CMA-ES and hill climbing.

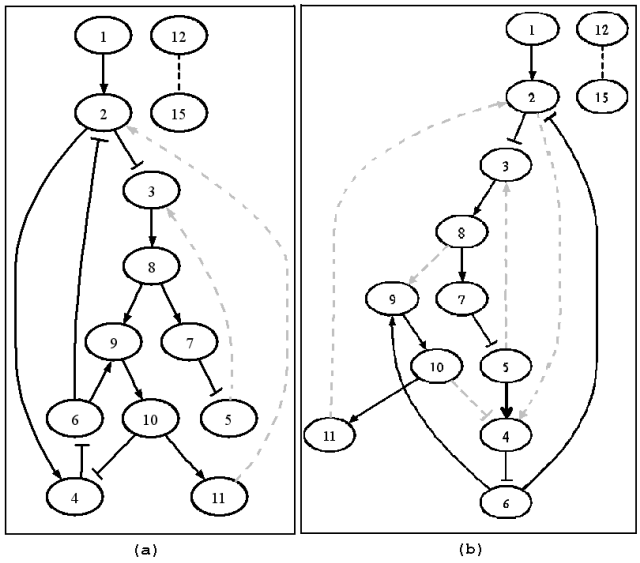

Figure 5.2: The interaction network inferred by DBN with (a) M-CMA-ES and (b) hill climbing.

- None of M-CMA-ES or hill climbing algorithms infer unnecessary interactions among independent genes 1, 12, 13, 14, and 15.

- The effect of gene 5 on gene 3 and gene 11 on gene 2 is not identified by either M-CMA-ES or hill climbing. This may be justified as follows: both genes 3 and 2 depend on two other genes; gene 3 depends on gene 2 and 5 and gene 2 depends on gene 6 and 11. The effect of gene 1 on gene 2 is ignored, since the role of gene 1 is to act as a random noise source and is not regulated by any other genes. When a gene is co-regulated by multiple genes with opposite regulatory effects (down-regulation, up-regulation), the relationship between the target gene and its regulators is a nonlinear relationship. In this experiment, gene 11 acts as an activator to gene 2 where gene 6 acts as its inhibitor. Due to observed small values of gene 11 in comparison to gene 6, the effect of gene 11 is less than that

of gene 6, which prevents the DBN with both M-CMA-ES and hill climbing from inferring the interaction between gene 11 and gene 2. In a similar way, the relationship between gene 3 and its regulators, genes 2 and 5, is a nonlinear relationship, where the effect of gene 2 is more pronounced than gene 5. To compare the magnitude of the expression levels of regulators 2, 5, 6, and 11, the histogram of absolute expression values of these genes in 1000 simulated datasets are plotted. The resulted histograms, show a higher average level for regulator 6 than 11. However, the absolute expression values for regulators 2 and 5 are almost at the same level.

- Several linear and nonlinear associations are missed by hill climbing. The nonlinear association between gene 4 and its regulators, gene 10 and 2, as well as the linear relationship between target gene 9 and its regulator, gene 8, is only inferred by M-CMA-ES not hill climbing (gene 9 is co-regulated by genes 6 and 8 in the same direction, that is why it is considered as linear relationship). An extra interaction from 5 to 4 is also inferred by hill climbing. The missing or extra associations found by hill climbing can be attributed to the algorithms getting trapped in local minimum and not being able to find the best solution.

To ensure the robustness of the obtained results, M-CMA-ES and hill climbing are further compared based on their ability to regenerate the simulated data. Gene expression levels at time t are predicted given expression levels at time *t-1* and using M-CMA-ES and hill climbing. The Mean Square Error (MSE) between the predicted and the simulated expression data is calculated for each method. The MSE of time series of all gene expression values are plotted using M-CMA-ES and hill climbing on 10 datasets in all of the plots, similar to what is shown in Figure 5.3.

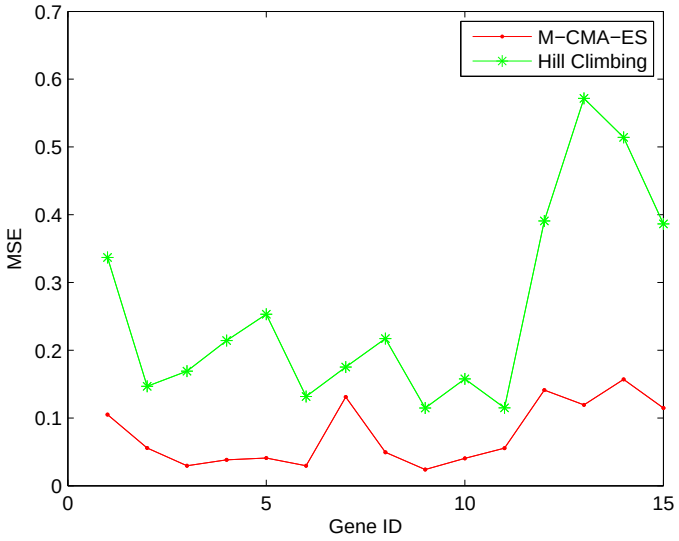

Figure 5.3: The plots of Mean Square Error of time series of genes for one dataset.

The error in time series of all gene expression predictions obtained by M-CMA-ES is less than hill climbing. To have a more precise picture of the difference between error in gene expressions prediction using M-CMA-ES and hill climbing, the histogram of the MSE is plotted over 10 datasets, shown in Figure 5.4.

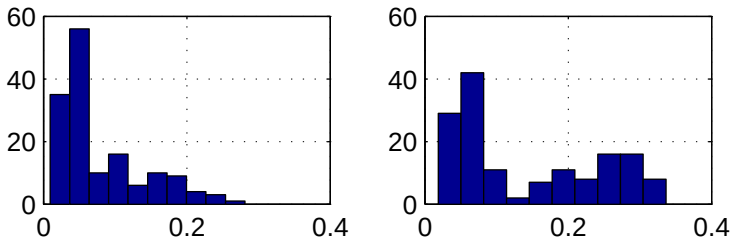

Figure 5.4: Histograms of Mean Square Error prediction of time series of genes for 10 simulated datasets using (a) M-CMA-ES and (b) hill climbing for DBN learning structure

It can be seen from the histograms that the majority of MSE for predictions for M-CMA-ES are less than 0.1 where this number is close to 0.3 for hill climbing.

The goodness of the obtained networks based on M-CMA-ES and hill climbing are

evaluated and compared using the criteria precision and sensitivity. The result, shown in Table 5.3 emphasizes the higher performance of M-CMA-ES than hill climbing by illustrating higher evaluation criteria precision and sensitivity for M-CMA-ES than hill climbing.

Table 5.3: Comparisons of network structures inferred by DBN with the heuristic search algorithms Hill Climbing and M-CMA-ES using the evaluation criteria sensitivity and precision

*	DBN and Hill Climbing	DBN and M-CMA-ES
Sensitivity	$\frac{9}{14} = 64\%$	$\frac{12}{14} = 86\%$
Precision	$\frac{9}{10} = 90\%$	$\frac{12}{12} = 100\%$

5.2 Analysis of Brainsim Dataset

As mentioned earlier, the proposed reverse engineering approach is also applied to Brainsim dataset which is a temporal dataset simulating the gene regulatory interactions underlying the singing behavior of a song bird. This dataset is used as a benchmark to validate the inference approach. The proposed method is applied to 400 datasets composed of 50 datasets for every combination of two songbirds and four regions of their brain. In order to eliminate both songbird and region specific biases, the datasets are taken from two songbirds and four regions. All the datasets have the same underlying network structure. However the associated weight values are different between different regions.

The network structure for 25 genes in each dataset is reverse engineered. The

network is then averaged over 400 datasets. For this purpose, a matrix is generated where each *(i,j)*-th element of the matrix is composed of the summation value of *(i,j)*-th element of all 400 interaction matrices. To distinguish significant interactions from irrelevant interactions, bootstrap strategy is employed. In this approach, 400 generated datasets are randomized by shuffling their values over time in all genes (in 200 datasets) or by shuffling their values over genes (in the other 200 datasets). The DBN and M-CMA-ES is then applied to reverse engineer 400 random datasets. A histogram is plotted for the number of times that every pairwise interaction is inferred from the 400 randomized datasets (Figure 5.5). The threshold is then put where the histogram had an obvious jump to infer significant interactions.

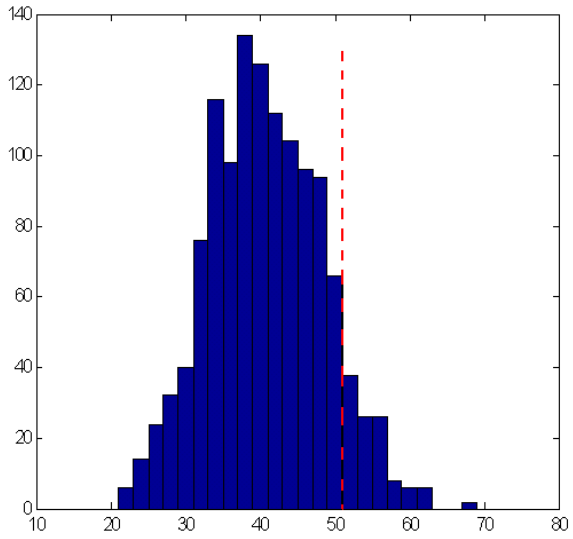

Figure 5.5: The histogram of the number of times that each pairwise gene interaction is inferred from 400 randomized Breainsim datasets by DBN and M-CMA-ES.

Figure 5.6 represents the network inferred using DBN and M-CMA-ES averaged

over 400 datsets. In Figure 5.6-b gray lines represent interactions which are incorrectly captured (extra interactions) and dashed lines show actual interactions not inferred using the proposed approach (missing interactions).

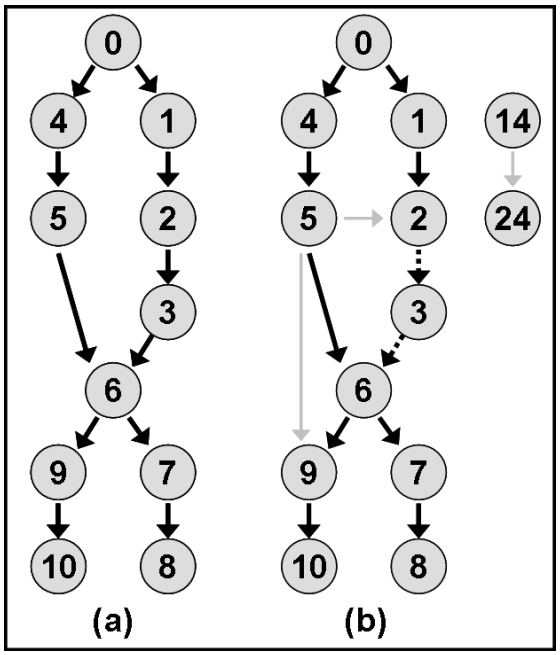

Figure 5.6: (a) The original network structure underlying the Brainsim dataset (b) The network inferred using proposed DBN with M-CMA-ES from 400 Brainsim datasets. Gray lines represent interactions which are incorrectly captured (extra interactions) and dashed lines show actual interactions not inferred (missing interactions).

By comparing the inferred network with the original network structure (Figure 5.6-a), It is observed that 9 out of all 12 inferred interactions are truly captured and 3 extra interactions are inferred; gene‘5’→ gene‘2’, gene‘5’→gene‘9’, and gene‘14’→gene‘24’. Among these interactions, gene‘5’ indirectly regulates gene‘9’ through gene‘6’ and gene ‘5’ and ‘2’ are indirectly co-regulated by the activity level.

The two interactions '2'→'3' and '3'→'6' are also missed. The interaction '3'→'6' was not also inferred by previous studies of Brainsim dataset [58].

Similar to the synthetic dataset, the accuracy of the obtained network is evaluated by DBN with M-CMA-ES using the criteria precision and sensitivity. The result are shown in Table 5.4.

Table 5.4: Evaluation of the inferred network from Brainsim dataset using the evaluation criteria sensitivity and precision

*	DBN and M-CMA-ES
Sensitivity	$\frac{9}{11} = 82\%$
Precision	$\frac{9}{12} = 75\%$

5.3 Analysis of Yeast *Saccharomyces Cerevisiae* Dataset

To evaluate the efficiency of the proposed reverse engineering approach using a biological dataset, it is applied to time series microarray expression data of 14 genes in yeast *Saccharomyces cerevisiae*(budding yeast) cell-cycle data reported in [59].

5.3.1 KEGG Pathway

Figure 3.3 illustrates the pathway of 14 genes involved in the early cell cycle of the yeast *Saccharomyces cerevisiae*[1](please note that the actual gene interaction network

[1]Cell cycle - yeast - Saccharomyces cerevisiae:
http://www.genome.ad.jp/kegg/pathway/sce/sce04111.html

which is reverse engineered is represented in Figure 5.7-a). A simplified minimal module to control the early cell cycle in budding yeast consists of three G1 cyclins, CLN1, CLN2, and CLN3, two B-type cyclins, CLB5 and CLB6, and a protein inhibitor of the complexes of CLB5/CDC28 and CLB6/CDC28, the SIC1 gene [61]. The transcription level of CLN3 is relatively constant in the cell cycle. However, a moderate increase is detected early after cell division and in late mitosis. In the beginning of cell cycle, the protein product of CLN3 binds to CDC28 protein kinase and activates it. When the concentration level of the CLN3/CDC28 complex crosses a threshold value, the cascade of cell cycle events is initiated.

The CLN3/CDC28 complex induces the expression of CLN1 and CLN2 by phosphorylating (activating) the SBF transcription factor comprising two subunits, SWI4 and SWI6. In a similar way, the CLN3/CDC28 complex regulates the transcription of CLB5 and CLB6 by phosphorylating MBF complex comprised of SWI6 and MBP1. In addition, The SBF and MBF transcription factor complexes contain a feed forward self regulation loop circuit [12]. Therefore, CLN3 protein as an activator, indirectly leads to increase in the transcription of SBF and MBF transcription factors.

SIC1 binds to the CLB-CDC28 complexes and inactivates them. SIC1 is then targeted for destruction by phosphorylation from CLN1/2 CDC28. CLB-CDC28 complexes are in turn responsible for proper timing of DNA replication. They also phosphorylate CDC6 which is required for initiation of DNA replication. CDC20 participates in targeting CLB5 and CLB6 proteins for ubiquination at the metaphase to anaphase transition of the cell cycle. Furthermore, to arrest cells in G1 for mating, FUS3 phosphorylates FAR1 which binds to and inhibits CDC28/CLN1 and CDC28/CLN2 complexes [61].

5.3.2 Prediction of Transcriptional Cell-Cycle Subnetwork in Yeast *Saccharomyces Cerevisiae*

In order to infer the causal interactions among the key cell cycle regulated genes in yeast *Saccharomyces cerevisiae*, DBNs with the M-CMA-ES is applied to three time series expression data of the 14 specific genes in the yeast cell cycle data [59]. The pathway of these genes in KEGG, shown in Figure 5.7-a, is regarded as the target to evaluate the networks obtained using the proposed reverse engineering approach.

The task of capturing meaningful causal gene associations requires one to consider suitable time delays that approximate the underlying biology in predicting the transcription level of a given target gene by its potential regulators. In reality, different parts of the pathway need different times for completion. However, to prevent the model from being too complex, one lag time that best fits the pathway is determined. To this end, the time series expression values of all genes in all datasets are plotted. By visually inspecting the plots of all genes and also by considering the fact that the majority of these genes show a single expression peak during a cell cycle, it is observed that there usually are 0, 1, or 2 units of time delay between the expression peak of target genes and that of their regulators in the KEGG pathway. Therefore, three different units of time delay, 0, 1, and 2 is considered for each dataset where a unit of time delay is defined as the interval between two successive time points in the dataset. Reverse engineering is then applied to the datasets alpha, cdc15, and elu for the three above-mentioned lag times to infer three separate networks from each dataset, a total of 9 networks for all datasets (Table 5.5).

Structure learning of DBNs requires sampling from a normal distribution; as such

Table 5.5: Evaluation of the networks obtained from the datasets alpha, cdc15, and elu for the three different units of time delay, 0, 1, and 2 using the evaluation criteria sensitivity and precision

	Alpha			CDC15			ELU		
Lag Times(min)	0	1	2	0	1	2	0	1	2
Sensitivity	30%	40%	30%	50%	80%	20%	30%	0%	0%
Precision	33%	31%	21%	50%	50%	12%	21%	0%	0%

it has an element of randomness associated with it. Network inference is repeated 50 times for each dataset and each lag time. Here forth, a particular dataset with a specific lagtime is referred to as a"case". The next challenge is to quantify the accuracy of the inferred networks, averaged over the 50 runs. The elements of 50 connectivity matrices obtained for each case are summed up to determine the significant interactions.

To determine which structural connections, inferred using different datasets, are reliable and which are artifacts of the noise in the network, a bootstrap approach is used. In this approach, the distribution of incorrect connections inferred from all datasets are estimated for three lagtimes. Those distributions are then used to set thresholds for determining significant links from those that are likely spurious.

To derive a distribution for spurious connections found by the proposed method, for each case, 50 random datasets are created such that the distribution of data remains unchanged. In other words, the values are either shuffled across time for all genes (in 25 datasets) or are shuffled over genes (in the other 25 datasets). DBN with M-CMA-ES is then employed to reverse engineer the 50 random datasets for each case. The above steps are repeated twice. The histograms are then plotted for the number of times that every pairwise interaction is inferred from the 50 random

datasets through reveres engineering for all three datasets and all three lag times (9 histograms).

The thresholds are set to the values at which the histograms show a jump. An interaction between two genes is deemed significant if it appears at least equal to the threshold value out of the 50 inferred networks.

The inferred network structure for each case, after thresholding, is then compared with the KEGG pathway, as the target network, using the evaluation criteria sensitivity and precision. Finally, the best network is chosen according to precision and sensitivity criteria and is compared with two previous studies of yeast cell cycle.

5.3.3 Results and Evaluation

In summary, DBN with M-CMA-ES is applied to the time series of 14 genes in three yeast cell-cycle datasets alpha, cdc15, and elu, for three different lag times 0, 1, and 2. The obtained networks are then compared with the KEGG pathway using the criteria sensitivity and precision.

Among all 9 inferred network structures, the network inferred from cdc15 dataset with a lagtime of 1 has the highest sensitivity and precision. This network is then selected to further test the stability of the proposed technique. The cdc15 dataset also has the fewest missing values among other datasets making it more reliable.

To ensure the robustness of the inferred interactions from the cdc15 dataset, the proposed approach is reapplied to this dataset under the same conditions. These two runs of network inference on the cdc15 dataset are referred to as Exp.1 and Exp.2. It is expected that the result of the new experiment would match that of the previous experiment. The network structures inferred in two experiments are summarized and

compared in Table 5.6. It is observed that almost 90 percent of the interactions of the networks are identical. This is a good indication of the stability of the proposed methodology.

The interactions found in common between two identical experiments on the cdc15 dataset, are then considered as the predicted transcription network model for the yeast cell cycle pathway of 14 specific genes in the Spellman dataset (Figure 5.7-b).

Table 5.6: The comparison between the obtained gene regulatory interactions from cdc15 dataset in two identical experiments.

	Exp. 1	Exp. 2	Common interactions between Exp. 1 & Exp. 2
Correctly Estimated	8	8	8
Misdirected	1	1	1
Incorrectly Estimated	7	6	5

The predicted network model using the proposed DBN with M-CMA-ES is compared with models obtained using DBN with hill climbing [41] and DBN with Structural Expectation Maximization (EM) [68]. The networks obtained from these DBN structure learning methods are represented in Figure 5.7. In this figure, a circle on a connection is used to represent a correctly estimated interaction, a cross indicates an incorrect interaction, and a triangle marks a misdirected edge.

Figure 5.7-a illustrates the yeast cell cycle pathway at the level of gene regulation. As mentioned before, all inferred interactions within the complexes are ignored as they do not have external effects independent of each other. However, due to their similar time series transcription values, they might be captured by computational methods.

By comparing the inferred networks using the M-CMA-ES, Structural EM, and

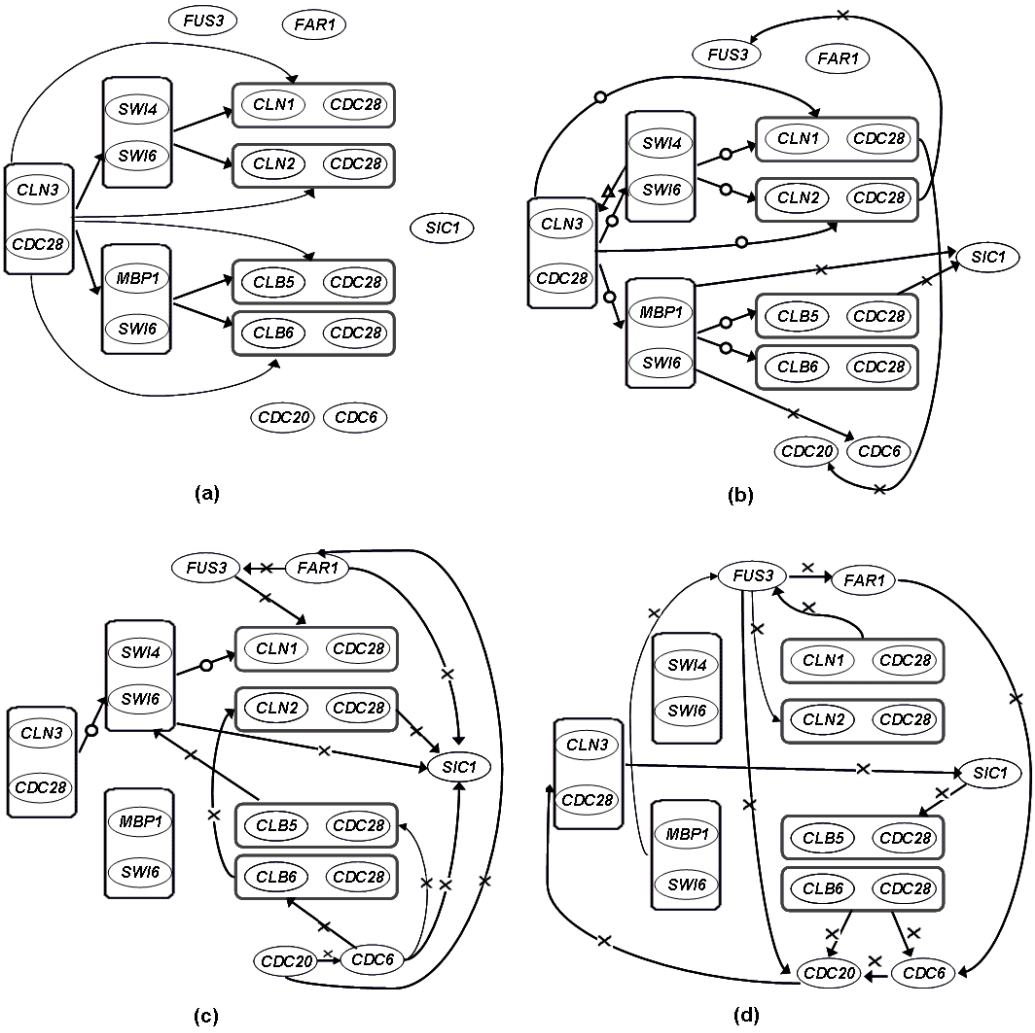

Figure 5.7: The yeast cell cycle pathway inferred from Spellman data. (a) Target pathway in KEGG. The pathway inferred by (b) the proposed DBN with M-CMA-ES (c) DBN with hill climbing (Kim et al., 2003), and (d) DBN with Structural EM (Yu et al., 2007). In this figure, A cross indicates an incorrect interaction, a circle is used to represent a correctly estimated interaction, and a triangle marks a misdirected edge.

hill climbing with the KEGG pathway, It is observed that eighty percent of the interactions presented in the target network are inferred by the proposed DBN and M-CMA-ES reverse engineering approach, while only two interactions are captured by DBN and structural EM. The network obtained using DBN and hill climbing [41] does not contain any of the true interactions. Also, M-CMA-ES approach outperforms hill climbing and structural EM in terms of predicting fewer incorrect and misdirected interactions.

Finally, the results using DBN with M-CMA-ES and other structure learning approaches, in predicting the true connectivity network, were compared using sensitivity and precision. Results, presented in Table 5.7, are a summary of Figure 5.7 and show

Table 5.7: Comparison of the proposed structure learning method, M-CMA-ES, with hill climbing and Structural EM summarized from Figure 5.7.

	M-CMA-ES	Structural EM	hill climbing
Correctly estimated	8	2	0
Incorrectly estimated	5	12	12
Misdirected	1	0	0
Precision= $\frac{Correctly\ est.\ Interactions}{All\ est.\ Interactions}$	$\frac{8}{14} = 57\%$	$\frac{2}{14} = 14\%$	$\frac{0}{15} = 0\%$
Sensitivity= $\frac{Correctly\ est.\ Interactions}{All\ correct\ Interactions}$	$\frac{8}{10} = 80\%$	$\frac{2}{10} = 20\%$	$\frac{0}{10} = 0\%$

that both evaluation criteria, precision and sensitivity, are distinctively higher for M-CMA-ES compared to previous methods indicating the efficiency of this approach.

Chapter 6

Conclusions and Future Work

6.1 Summary and Conclusions

Reverse engineering gene regulatory interactions from expression profiles of a set of genes is an important but challenging area of research in systems biology. In this thesis, a reverse engineering approach is proposed based on DBNs and a covariance-based evolutionary search strategy to model an optimal DBN from temporal gene expression data. The convergence time of the proposed algorithm for structure learning of DBN, M-CMA-ES, is improved compared to the previously reported covariance-based evolutionary search approaches. This extends the applicability of M-CMA-ES to problems with relatively high dimensional search space. The improved convergence of M-CMA-ES is achieved by keeping a fixed number of good sample solutions in each iteration of the algorithm and using it in the next iteration. This guarantees that the best generated sample at each iteration of the algorithm is at least as optimum as that of previous iteration while the size of the generated population does not change through iterations. Furthermore, the proposed approach, M-CMA-ES, compared to

gradient-based methods, is less probable to get trapped in local minima.

To evaluate the reliability and efficiency of M-CMA-ES for infering causal regulatory interactions from temporal gene expression data, DBN built using this approach is compared to DBN with hill climbing on a temporal synthetic dataset. Hill climbing is selected as it is the most frequently used algorithm for DBN structure learning.

As expected and emphasized by the results obtained from the temporal synthetic dataset, the proposed method M-CMA-ES is more efficient than hill climbing for identification of the best set of regulators for a gene of interest using DBN. Unlike hill climbing, M-CMA-ES successfully finds all linear relationships between genes in the synthetic dataset. In addition, more nonlinear relationships are inferred by M-CMA-ES than by hill climbing. Finally, no extra interactions are inferred by M-CMA-ES as opposed to the extra interaction identified by hill climbing.

The accuracy of the inferred structure from this dataset is quantified via mean-square error as well as two statistical criteria 'sensitivity' and 'precision'. M-CMA-ES, yields a lower mean-square error for prediction of expression values at time t given the expression values at time *t-1*. Furthermore, both evaluation criteria precision and sensitivity yield higher values for M-CMA-ES than hill climbing indicating more efficiency of M-CMA-ES compared to hill climbing in terms of predicted structure accuracy.

For more evaluation, DBN with M-CMA-ES is also applied to the Brainsim dataset, a temporal simulated dataset with known structure that models the singing behavior in a songbird. The inferred structure quantified in via the criteria precision and sensitivity, indicates a good performance of the network inference approach; 9 out of all 12 inferred interactions are true network connections and only two of the

actual interactions are missed by the DBN model.

Finally, the efficiency of M-CMA-ES for learning the structure of DBN, is evaluated using a biological dataset, the temporal expression values of 14 genes in yeast *Saccharomyces cerevisiae* cell-cycle data as reported in [59]. The networks inferred from the yeast cell cycle data are compared to the KEGG pathway of the yeast as the target network and those of other similar yeast network inference studies using evaluation criteria sensitivity and precision. The results indicate markedly improved scores for M-CMA-ES approach compared to previous methods reported in the literature; 80% of the actual interactions are inferred by the M-CMA-ES approach, while only 20% are captured by DBN and structural EM reported in [68]. The network obtained using DBN and hill climbing [41] does not contain any of the true interactions. In addition, M-CMA-ES outperforms hill climbing and structural EM in terms of predicting fewer incorrect and misdirected interactions.

In conclusion, the results demonstrate a good performance for the proposed DBN structure learning approach, M-CMA-ES, in terms of both predicted structure accuracy and mean square error for prediction of time series of gene expression values. Moreover, the explicit memory incorporated in the algorithm speeds up the convergence and makes it more applicable to the problem of reverse engineering of gene networks.

6.2 Future Work

Design of a reverse engineering method, similar to most other computational algorithms is concerned with finding an appropriate trade-off between accuracy of the obtained model and its required computational time. An extensive amount of future

work is considerable in this work which are mentioned below:

- Studying alternative basis functions to B-spline used in this work, to approximate the coefficients of the regression function modeling the association between expression value of a given gene and that of its potential regulators in DBN. Application of a simpler alternative might decrease the computation time of DBN model.

- Developing an efficient optimization algorithm for parameter estimation of the DBN model; in this work, Backfitting algorithm was used. An efficient optimization procedure might decrease the convergence time or enhance the accuracy of the model.

- Considering biological knowledge to determine transcription factors or potential gene regulators and assigning a higher prior probability to the genes which are more probable to be potential regulators. The biological knowledge can also be used for determining the maximum number of potential regulators for a given gene or selecting a meaningful time-lag to predict the gene expression values over time.

- Investigating the relationship between the size of the generated population at each iteration of M-CMA-ES, and size of the explicit memory added to keep a number of good sample solutions from previous iterations. In this work, a fixed-size memory is used for this purpose.

- Generalizing the proposed model to a model that allows different regulators of a gene regulate their target gene with different time-lags. This results in a more flexible network inference model with higher accuracy; as in reality, a target

gene is not usually regulated by different transcription factors with the same time delay.

- Investigating the effects of changing the upper bound constraint on the number of potential regulators of a given gene.

- Defining the Bayesian score for an alternative DBN local structure. In this work, the Bayesian score is defined for individual genes and their potential regulators. Alternative approaches could include more complex local structures.

Bibliography

[1] Cell cycle - yeast - saccharomyces cerevisiae http://www.genome.ad.jp/kegg/pathway/sce/sce04111.html.

[2] S. Aburatania, S. Saitob, H. Toh, and K. Horimotoa. A graphical chain model for inferring regulatory system networks from gene expression profiles. *Statistical Methodology*, 3(1):17–28, 2006.

[3] K. Baba, R. Shibata, and M. Sibuya. Partial correlation and conditional correlation as measures of conditional independence. *Australian and New Zealand Journal of Statistics*, 46(4):657–664, December 2004.

[4] O. Berkman and N. Intrator. Robust inference in bayesian networks with application to gene expression temporal data. *Lecture Notes in Computer Science*, 4472:479–489, 2007.

[5] D. R. Bickel. probabilities of spurious connections in gene networks:application to expression time series. *Bioinformatics*, 21(7):1121–1128, April 2005.

[6] A. Brazma, H. Parkinson, T. Schlitt, and M. R. Shojatalab. Introduction to elements of biology: Cells - molecules - genes - functional genomics - microarrays. Technical report, EMBL- European Bioinformatics Institute, 2001.

[7] L. Breiman and J. H. Friedman. Estimating optimal transformations for multiple regression and correlations (with discussion). *Journal of the American Statistical Association*, 80(391):580–619, 1985.

[8] A. Butte and I. Kohane. Mutual information relevance networks:functional genomic clustering using pairwise entropy measurements. *Pacific Symposium on Biocomputing*, 5:415–426, 2000.

[9] L. Campos and J. Huete. On the use of independence relationships for learning simplified belief networks. *International Journal of Intelligent Systems*, 12(7):495–522, Dec 1998.

[10] X. Chen, G. Anantha, and X. Wang. An effective structure learning method for constructing gene networks. *Bioinformatics*, 22(11):1367–1374, 2006.

[11] G. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Mahcine Learning*, 9(4):309–347, 1992.

[12] O. X. Cordero and P. Hogeweg. Feed-forward loop circuits as a side effect of genome evolution. *Molecular Biology and Evolution*, 23(10):1931–1936, 2006.

[13] X. Cui and G. A. Churchill. Statistical tests for differential expression in cdna microarray experiments. *Genome Biology*, 4(210), March 2003.

[14] E. H. Davidson and D. H. Erwin. Gene regulatory networks and the evolution of animal body plans. *Sceince*, 311(5762):796–800, 2006.

[15] R. C. Deonier, M. S. Waterman, and S. Tavar. *Computational Genome Analysis-an introduction, Biology in a Nutshell*. Springer-New York, 2005.

[16] A. Dobra, C. Hans, B. Jones, J. R. Nevins, G. Yao, M. West, and J. Multiv. Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis*, 90(1):196–212, 2004.

[17] J. Dopazo, E. Zanders, I. Dragoni, G. Amphlett, and F. Falciani. Methods and approaches in the analysis of gene expression data. *Journal of Immunological Methods*, 250(1-2):93–112, 2001.

[18] S. Draghici. *Data Analysis tools for DNA microarrays*. Chapman and Hall-CRC, 2003.

[19] D. Edwards. *Introduction to Graphical Modelling*. Springer, New York, 2nd edition, 2000.

[20] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using bayesian networks to analyze expression data. *Journal of Computational Biology*, 7(3-4):601–620, 2000.

[21] T. S. Gardner and J. J. Faith. Reverse-engineering transcription control networks. *Physics of Life Reviews*, 2(1):65–88, 2005.

[22] P. Spirtes C. Glymour and R. Scheines. *Causation, Prediction and Search*. MIT Press, first edition, 1993.

[23] N. Hansen. The cma evolution strategy: A comparing review. *Studies in Fuzziness and Soft Computing*, 192:75–102, 2006.

[24] N. Hansen. The cma evolution strategy: A tutorial. Technical report, 2007.

[25] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distribution in evolutionary strategies: The covariance matrix adaptation. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 312–317, Nagoya, 1996.

[26] T. Hastie and R. Tibshirani. Bayesian backfitting. *Statistical Science*, 15(3):196–223, 2000.

[27] D. Heckerman, D. Geiger, and D. Chickering. Learning bayesian networsk: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

[28] G. F. Holness. A direct measure for the efficacy of bayesian network structures learned from data. *Lecture Notes in Computer Science*, 4571:601–615, 2007.

[29] E. Hopper and B. C. H. Turton. Reverse-engineering transcription control networks. *Physics of Life Review*, 2(1):65–88, 2005.

[30] I. Hovatta, K. Kimppa, A. Lehmussola, T. Pasanen, J. Saarela, I. Saarikko, J. Saharinen, P. Tiikkainen, T. Toivanen, M. Tolvanen, M. Vihinen, and G. Wong. *DNA Microarray Data Analysis*. CSC, Scientific Computing Ltd., 2005.

[31] L. Hunter. *Molecular biology for computer scientists, Artificial Intelligence and Molecular Biology*. MIT Press- Cambridge, MA, 1993.

[32] L. Hunter. Life and its molecules, a brief introduction. *American Association for Artificial Intelligence*, 25:9–22, 2004.

[33] D. Husmeier. *Probabilistic Modeling in Bioinformatics and Medical Informatics*. Springer-Verlag London Ltd., 2005.

[34] S. Imoto, T. Goto, and S. Miyano. Estimation of genetic networks and functional structures between genes by using bayesian networks and nonparametric regression. In *In The Proceedings of Pacific Symposium on Biocomputing, World Scientific*, pages 175–186, Lihue, January 2002.

[35] M. Janura and J. Nielsen. A simulated annealing-based method for learning bayesian networks from statistical data. *International Journal of Intelligent Systems*, 21(3):335–348, February 2004.

[36] R. Kabli, F. Herrmann, and J. McCall. A chain-model genetic algorithm for bayesian network structure learning. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1264–1271, London, England, July 2007.

[37] E. Keedwell and A. Narayanan. Genetic algorithms for gene expression analysis. *Proceeding of EvoBio2003 1st European Workshop on Evolutionary Bioinformatics*, pages 76–86, 2003.

[38] E. Keedwell, A. Narayanan, and D. Savic. Modeling gene regulatory data using artificial neural networks. *Proceedings of the 2002 International Joint Conference on Neural Network*, 1:183–188, May 2002.

[39] H. Kim, G. H. Golub, and H. Park. Missing value estimation for dna microarray gene expression data: Local least squares imputation. *Bioinformatics*, 21(2):187–198, 2005.

[40] 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*, 75(2602):57–65, 2004.

[41] S. Y. Kim, S. Imoto, and S. Miyano. Infering gene networks from time series microarray data using dynamic bayesian networks. *Briefings In Bioinformatics*, 4(3):228–235, 2003.

[42] H. Kishino and P. J. Waddell. Correspondence analysis of genes and tissue types and finding genetic links from microarray data. *Genome Informatics*, 11:83–95, 2000.

[43] S. Knott. Computational approaches for reverse engineering large scale gene expression datasets. Master's thesis, Queen's University, Kingston, August 2006.

[44] S. Knott, S. Mostafavi, P. Mousavi, and J. Glasgow. Genetic network inference via gene set stochastic sampling and sensitivity analysis. *Proceedings of the 2005 IEEE Conference on Control Applications*, pages 148–153, August 2005.

[45] S. Knott, P. Mousavi, and S. Baranzini. A systematic approach for identifying regulatory interactions in largetemporal gene expression datasets from peripheral blood. *Proceedings of IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–8, September 2006.

[46] W. Lam and F. Bacchus. Learning bayesian belief networks: An approach based on the mdl principle. *Computational Intelligence*, 10:269–293, 1994.

[47] P. Leray and O. Francois. Bayesian network structural learning and incomplete data. In *In Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, Espoo, July 2005.

[48] S. Liang, S. Fuhrman, and R. Somogyi. Reveal, a general reverse engineering algorithm for inference of genetic network architectures. *Proceedings of Pacific Symposium Biocomp*, pages 18–29, January 1998.

[49] F. Markowetz and R. Spang. Inferring cellular networks a review. *Nature Immunology*, 8(6):63–73, September 2007.

[50] David M. Mutch, Alvin Berger, Robert Mansourian, Andreas Rytz, and Matthew-Alan Roberts. The limit fold change model: A practical approach for selecting differentially expressed genes from microarray data. *BMC Bioinformatics*, 3(17), 2002.

[51] I. Nachman, A. Regev, and N. Friedman. Inferring quantitative models of regulatory networks from expression data. *Bioinformatics*, 20(1):248–256, August 2004.

[52] A. Ngele, M. Dejori, and M. Stetter. Bayesian substructure learning - approximate learning of very large network structures. *Proceedings of 18th European Confernce on Machine Learning*, pages 238–249, 2007.

[53] H. Ogata, S. Goto, W. Fujibuchi, and M. Kanehisa. Computation with the kegg pathway database. *Biosystems*, 47(1-2):119–128, June 1998.

[54] B. E. Perrin, L. Ralaivola, A. Mazurie, S. Bottani, J. Mallet, and d'Alche Buc F. Gene networks inference using dynamic bayesian networks. *Bioinformatics*, 19(2):138–148, October 2003.

[55] N. Radde and L. Kaderali. Bayesian inference of gene regulatory networks using gene expression time series data. *Lecture Notes in Bioinformatics (LNBI)*, 4414:1–15, 2007.

[56] J. Schfer and K. Strimmer. Learning large-scale graphical gaussian models from genomic data. *In Proceedings of "Science of Complex Networks: from Biology to the Internet and WWW" (CNET 2004)*, pages 263–276, August 2004.

[57] J. S. Schfer. *Small-Sample Analysis and Inference of Networked Dependency Structures from Complex Genomic Data.* PhD thesis, University of Munich, November 2006.

[58] V.A. Smith, E.D. Jarvis, and A.J. Hartemink. Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics*, 18(90001):216–224, 2002.

[59] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. Q. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Molecular Biology of the Cell*, 9(12):3273–3297, Decembre 1998.

[60] N. Sugimoto and H. Iba. Inference of gene regulatory networks by means of dynamic differential bayesian networks and nonparametric regression. *Genome Informatics*, 15(2):121–130, 2004.

[61] A. P. Taylor, Z. Darieva, B. A. Morgan, and A. D. Sharrocks. Regulation of cell cycle-specific gene expression through cyclin-dependent kinase-mediated phosphorylation of the forkhead transcription factor fkh2p. *Molecular and Cellular Biology*, 24(22):10036–10046, 2004.

[62] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R.B. Altman. Missing value estimation methods for dna microarray. *Bioinformatics*, 17(6):520–525, 2001.

[63] E. v. Someren, L. Wessels, E. Backer, and M. Reinders. Genetic network modeling. *Pharmacogenomics*, 3(4):507–525, Jul 2002.

[64] F. Valafar. Pattern recognition techniques in microarray data analysis: A survey. *Special Issue of annals of New York Academy of Sciences,Techniques in Bioinformatics and Medical Informatics*, 980:41–64, Decembre 2002.

[65] A. V. Werhli, M. Grzegorczyk, and D. Husmeier. Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical gaussian models and bayesian networks. *Bioinformatics*, 22(20):2523–2531, 2006.

[66] J. Yu, V. A. Smith, P. P. Wang, A. J. Hartemink, and E. D. Jarvis. Using bayesian network inference algorithms to recover molecular genetic regulatory networks. In *Proceedings of International Conference on Systems Biology (ICSB02)*, Stokholm, December 2002.

[67] K. Yu, H. Wang, and X. Wu. A parallel algorithm for learning bayesian networks. *Lecture Notes in Computer Science*, 4426:1055–1063, 2007.

[68] Y. Zhang, Z. Deng, H. Jiang, and p. Jia. Inferring gene regulatory networks from multiple data sources via a dynamic bayesian network with structural em. *Lecture Notes in Computer Science*, (4544):204–214, 2007.

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

Glossary

amino acids A set of molecular building blocks that are assembled according to blueprints encoded in DNA to form a protein. There are 20 different amino acids.

cell cycle The series of events that lead to the replication of a eukaryotic cell.

CMA-ES An acronym for Covariance Matrix Adaptation Evolution Strategy that is a non-linear iterative optimization method first proposed by Hansen in 1996; This method aims to optimize (minimize) an objective function through an iterative process.

DNA Acronym for deoxyribonucleic acid. A nucleic acid consisting of large molecules shaped like a double helix; associated with the transmission of genetic information.

eukaryotes Organisms whose cells are organized into complex structures enclosed within membranes.

expression profile The transcription level of a gene as it changes over time.

gene A segment of DNA that contains all the information necessary to code for a protein.

gene regulatory network A complex network composed of a group of genes and their regulatory interactions through transcription factors.

interphase A process by which a eukaryotic cell grows, duplicates its DNA and gets ready for mitosis (see mitosis).

KEGG An acronym for Kyoto Encyclopedia of Genes and Genomes.

KEGG pathway is a collection of manually drawn pathway maps representing our knowledge on the molecular interaction and reaction networks for 'metabolism','genetic information processing','environmental information processing','cellular processes',and 'human diseases'.

M-CMA-ES An acronym for Covariance Matrix Adaptation Evolution Strategy with explicit Memory defined based on CMA-ES (see CMA-ES); M-CMA-ES improves the convergence time of CMA-ES by keeping a fixed number of good samples from previous iteration.

microarray A collection of single-stranded DNA segments deposited or synthesized as spots on a solid surface in a grid like pattern. Used to measure the expression levels of large numbers of genes simultaneously.

mitosis A process by which a eukaryotic cell divides into two distinct cells called daughter cells.

mRNA An acronym for messenger ribonucleic acid. A gene specific single stranded piece of genetic material that is the product of its corresponding genes transcription.

prokaryotes Organisms that usually lack a cell nucleus or any other membrane-bound organelles and mostly are unicellular.

protein A fundamental structural and functional unit in cells which can act as structural elements, enzyme catalysts, and antibodies.

transcription The process where a gene is transcribed into a single stranded mRNA molecule. The process is catalyzed by RNA polymerase.

transcription factor A protein that regulates the transcription of particular genes by binding to specific sites of the DNA.

translation The assembly of amino acids on a ribosome. The order of assembly is dictated by the triplet codons contained in a corresponding mRNA strand.